A closer look at IP-ID behavior in the Wild

Flavia Salutari, Danilo Cicalese, and Dario J. Rossi

Telecom ParisTech, Paris, France – `first.last@telecom-paristech.fr`

Abstract. Originally used to assist network-layer fragmentation and reassembly, the IP identification field (IP-ID) has been used and abused for a range of tasks, from counting hosts behind NAT, to detect router aliases and, lately, to assist detection of censorship in the Internet at large. These inferences have been possible since, in the past, the IP-ID was mostly implemented as a simple packet counter: however, this behavior has been discouraged for security reasons and other policies, such as random values, have been suggested.
In this study, we propose a framework to classify the different IP-ID behaviors using active probing from a single host. Despite being only minimally intrusive, our technique is significantly accurate (99% true positive classification) robust against packet losses (up to 20%) and lightweight (few packets suffices to discriminate all IP-ID behaviors). We then apply our technique to an Internet-wide census, where we actively probe one alive target per each routable /24 subnet: we find that that the majority of hosts adopts a constant IP-IDs (39%) or local counter (34%), that the fraction of global counters (18%) significantly diminished, that a non marginal number of hosts have an odd behavior (7%) and that random IP-IDs are still an exception (2%).

1 Introduction

The IP identifier (IP-ID) is a 16 (32) bits field in the IPv4 (v6) header [24]. Originally, along with the fragment offset, IP-ID was used to assist packet segmentation and reassembly and it was unique per each combination of source, destination and protocol. Yet, with technology evolution and the adoption of the MTU path discovery [21], IP fragmentation is much less common nowadays, so that the last normative reference [26] allow IP-ID of atomic datagrams to be be non-unique.

As a consequence, IP-ID fields values are determined by the specific implementation on the Operating System [22]. Over time, different behaviors have been observed such as global and per-flow counters, pseudo-random sequences and constant values [2], as well as odd behaviors such as those due to load balancing [6] middleboxes, or host implementations using the wrong endianness [22]. Given that some of the above implementations maintain state at the IP level, the IP-ID has been widely studied [2,20,25], abused [6,14,15], and more recently used to assist host identification [4,19,22,23].

In particular, the majority of research work focus their attention on the *global* counter implementation, which used to the be the most common implementation about a decade ago [27]. However, due to recent evolution of the standards [10, 26], a wider range of behaviors can be expected nowadays. Given this context, we can summarize our main contributions in this work as:

- we design and implement a lightweight methodology to classify the full range of IP-ID behaviors, based on a handful of ICMP packets
- we carefully validate our method against a dataset comprising about 1,855 sample hosts, for which we built a ground-truth by manual inspection
- we apply the methodology to a Internet-wide campaign, where we classify one alive target per each routable /24 subnet, gathering an unbiased picture of the IP-ID adoption

Specifically, whereas the global counter (18% in our measurement) implementation was the most common a decade ago [27], we find that other behaviors (constant 34% and local counter 39%) are now prevalent. We also find that security recommendations expressed in 2011 [10] are rarely followed (random, 2%). Finally, our census quantifies a non marginal number of hosts (7%) showing evidence of a range of behaviors, that can be traced to poor or non-stand implementations (e.g., bogus endianness; non-standard increments) or other techniques (e.g., load balancing). To make our findings profitable to a larger extent, we make all our dataset and results available to the scientific community at [1].

2 Background and related work

Background. The IP-ID field identifies uniquely fragments of the packets and it is used to handle the re-assembling process. First documented in the early 80s by RFC791 [24] its use has been updated in several RFCs [5, 8, 10, 11, 26, 27]. Whereas [24] does not fully specify IP-ID behavior (i.e., it only states that each packet must have a unique IP-ID for the triplet of source, destination and protocol), different behaviors (namely Global, Local and Random, illustrated in Fig.1) are detailed in 2006 by RFC4413 [27].

In 2008, RFC5225 [8] observed that some hosts set the IP-ID to *zero*: at the time of [8], this was a not legal implementation as the field was supposed to be unique. Yet, in 2012 [22] observed that the actual IP-ID implementation depends on the specific Operating System (OS) and versions[1]. In 2013, RFC6864 [26] updated the specifications by affirming that the IPv4 ID uniqueness applies to only non-atomic datagrams: in other words, if the don't fragment (DF) bit is set, reassembly is not necessary and hence devices may set the IP-ID to zero.

At the same time, concern has been raised about security problems following the predictability of IP-ID sequences [9, 11, 13, 17]. In particular, in 2012 RFC6274 [10] discouraged the use of a global counter implementation for many security issues, such as stealth port scan to a third (victim) host, and in 2016 RFC7739 [11] addressed concerns concerning IPv6-specific implementations. In

[1] In particular [22] reports Windows and FreeBSD to use a global counter, Linux and MacOS to use local counters and OpenBSD to use pseudo-random IP-IDs.

light of these recent evolution of the standards, a re-assessment of IP-ID usage in the wild it thus highly relevant.

Related work. Additionally, the IP-ID has been exploited for numerous purposes in the literature. Notably, IP-ID side-channel information helped discovering load balancing server [6], count hosts behind NAT [2, 22], measure the traffic [6, 14] and detect router alias [3, 16, 25]. More recently, [19] leverages IP-ID to detect router aliases, or inferring router up time [4] and to reveal Internet censorship [23], refueling interest in the study of IP-ID behavior. Whereas the above work [2,6,14,23,25] mostly focus only on the global IP-ID behavior, in this work we not only consider all *expected* IP-ID behavior, but additionally quantify *non-standard* behaviors: in particular, we provide a methodology to accurately classify IP-ID behaviors, that we apply to the Internet at large, gathering an unbiased picture of the relative popularity of each IP-ID behavior.

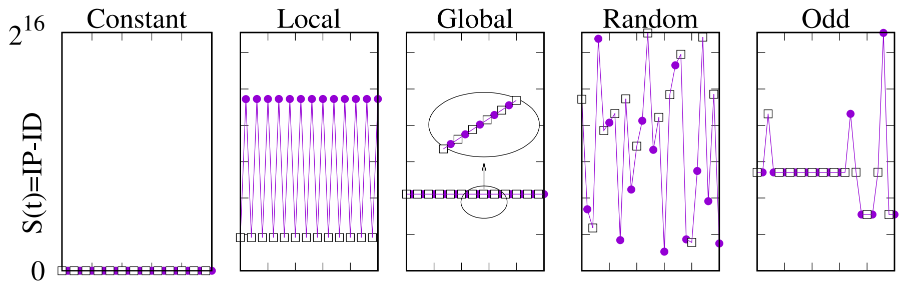
In terms of methodologies, authors in [20] use ICMP timestamp and IP-ID to diagnose paths from the source to arbitrary destinations and find reordering, loss, and queuing delay. In [15], the authors identify out-of-sequence packets in TCP connections that can be the result of different network events such as packet loss, reordering or duplication. In [6], they use HTTP requests from two different machines toward 150 target websites, to discover the number of load-balancing server. Authors in [23] use TCP SYN-ACK from multiple vantage points to identify connectivity disruptions by means of IP-ID fields, which then they use as a building block of a censorship detection framework. In this work, we leverage ICMP traffic (spoofing IPs to craft sequences of packets that precisely interleaved when they hit the target under observation) to build an accurate, robust and lightweight IP-ID classification technique.

3 Methodology

To provide an accurate and comprehensive account of IP-ID behavior in the wild, we need (i) a reliable classifier, able to discriminate among the different typical and anomalous IP-ID behaviors. At the same time, to enable Internet coverage, (ii) the classifier should rely on features with high discriminative power, extracted from an active probing technique that is as lightweight as possible. In this section we illustrate the practical building blocks and their theoretical foundations, that our classification framework builds upon.

3.1 IP-ID classes

From the host perspective, several IP-ID behaviors are possible as depicted in Fig.1. It shows sequences s of 25 IP-ID samples sent from 2 different host (dark and white) where the packets are sent alternatively to the target. The different behaviors depicted are, from left to right: (i) **constant** counters are never incremented (and for the most part are equal to `0x0000`); (ii) **local** or per-host counters that are incremented at each new packet arrival for that flow (mostly



Feature	Expected values				
$H(s)$	0	$\leq \log_2 N$	$\log_2 N$	$\leq \log_2 N$	-
$\sigma_x = \sigma_y$	0	$\sqrt{(N^2-4)/48}$	$\sqrt{(N^2-4)/12}$	$(2^{16}-1)/\sqrt{12}$	-
$\mathbb{E}[x'] = \mathbb{E}[y']$	0	1	2	$(2^{16}-1)/2$	-
σ_s	0	$\leq (2^{16}-1)/\sqrt{12}$	$\sqrt{(N^2-1)/12}$	$(2^{16}-1)/2$	-
$H(x') = H(y')$	0	0	0	$\leq \log_2 N/2$	-
σ'_s	0	$\lvert x_1 - y_1 - 1/2 \rvert$	0	$(2^{16}-1)/\sqrt{12}$	-

Fig. 1: Illustration of Constant, Local, Global, Random and Odd sequences (top) and tabulated expected values for selected features (bottom)

by 1 unit): as a consequence, while the black or white per-host sub-sequences are monotonically increasing, the aggregate sequence alternates between the two; (iii) **global** counters are incremented by 1 unit (rarely by 2 units) at each new packet arrival for any flow: thus, the sequence s is monotonically increasing (by 1 unit), and the black or white per-host sub-sequences are monotonically increasing but at a faster rate (by 2 units); (iv) **random** IP-IDs are extracted according to a pseudo-random number generator. Finally, a special mention is worth for the class of (v) **odd** IP-ID behaviors, that are not systematically documented in the literature and that arise for several reasons (including bugs, misconfiguration, non-standard increments, unforeseen interaction with other network apparatuses, etc.).

3.2 Active probing

To gather the above described sequences, our measurement technique relies on active probing. We craft a tool able to send and receive ICMP packets, running at two vantage points (VP) with public IP addresses in our campus network.

Specifically, we send a stream of N ICMP echo requests packets in a *back-to-back* fashion, which forces the target machine to generate consecutive ICMP echo replies: thus, assuming for the time being that no packet were lost, we gather a stream of N IP-IDs samples for that target. Sending packets back-to-back is necessary to reduce the noise in the IP-IDs stream sequence: if probe packets were spaced over time, the sequence could be altered by exogenous traffic hitting the target (e.g., in case of global counter). As a result, the sequence would depend on the (unknown) packet arrival rate in between two consecutive probe packets, making the classification more complex.

A second observation is that, whereas a single vantage point may be sufficient to distinguish among constant, random and global counters, it would fail to discriminate between global vs local counters. However, send packets from two different VPs is not advisable, due to the difficulty in precisely synchronizing the sending patterns so that packets from different hosts alternate in the sequence.

Therefore, a better alternative is to receive packets on two different VPs, x and y, but use only one of them, x, as sender: by letting x spoof the address IP_y of the colluding receiver y, it is possible to generate a sequence of back-to-back packets that is also perfectly interleaved as in Fig.1. To identify reordering, packet loss and duplication, we additionally control the sequence number in the stream of generated probe packets.

3.3 Features

We need to define features able to discriminate among IP-IDs implementations. We send N packets to a given target t, whose reply are sent back to our two VPs x and y: we indicate with s the aggregated sequence comprising N IP-IDs sent back by t, as we receive it at the edge of our network[2]. By abuse of language, we indicate with x and y the sub-sequences (each of length $N/2$) of IP-IDs, received by the homonyms host. From these sequences x, y and s we further construct derivative series x', y' and s' by computing the discrete differences between the consecutive IP-IDs (i.e., $x'_i = x_i - x_{i-1}$). We summarize these series with few scalar features by computing the first $\mathbb{E}[\cdot]$ and second moment $\sigma_{\cdot}$ of the IP-ID series, as well as their entropy $H(\cdot)$.

Intuitively, we expect the mean of the constant sequence to be unknown, but its derivative to be null. Similarly, derivative of a global counter would have a value of 1 (2) for the aggregate sequence s (subsequences x and y). Entropy of the sequence is expected to increase from a minimum of the constant sequence, to a global counter, to local counters, and to be maximum for random sequences. Actually, for each feature we can derive an *expected value* in the ideal[3] case (so that no expected values is reported for the odd class): for lack of space, we do not report the full mathematical details in this paper, that the reader can find in [1], but we summarize the main takeaway in the bottom part of Fig.1. Specifically, for each of the observed behavior depicted on the top plots, the figure tabulates the expected values for 6 relevant features (namely $H(s), \sigma_x, \mathbb{E}[x'], \sigma_s, H(x'), \sigma'_s$). The specific choice is motivated by the fact that these features happens to have the highest discriminative power, as later shown.

[2] Notice that packet losses and reordering may let us receive less than N packets, or receive packets in a slight different order than what sent by the target. We come back to this issue later on

[3] Sequences from well behaving hosts that have no software bug or malicious behavior, and that are neither affected by losses nor reordering

4 IP-ID Classification

From the values tabulated in Fig.1, we expect classifiers that use this set of features to fully discriminate the set of IP-ID well-defined behaviors under ideal conditions. However, as we shall see, unexpected behavior may arise in the Internet, due to a variety of reasons, which are hard to capture in general. We thus opt for a *supervised classification* approach, to learn a predictive model with decision trees (DTs), based on the above features. Specifically, we resort to the Classification And Regression Trees (CART) [18], that builds trees having the largest information gain at each node. DTs are part of the *supervised* machine learning algorithms, and infer a classification function from a (i) *labeled* training dataset, that we need to build and that is useful for training and validation purposes. Additionally, we investigate (ii) to what extent the classifier is robust against losses, and finally (iii) assess the minimum number of samples N needed to achieve a reliable classification.

4.1 Validation

We first train and validate our classifier using a real dataset $\mathcal{G}$ of IP-ID sequences for which we construct a ground truth. For this purpose, we perform a small-scale measurement campaign where we select 2,000 Internet targets and send sequences of $N = 100$ ICMP probes. We include in this dataset only the 1,855 hosts from which we receive 100% of the replies, and perform manual inspection of each of the sequences. Interestingly, we find a small but non marginal fraction (about 7%) of sequences that are hard to classify: a deeper investigation reveals these odd behaviors to be due to a variety of reasons – including per-packet IP-level load balancing, wrong endianness, non standard increments in the global counter, etc. These samples provide a useful description of the unexpected behavior class, that would otherwise have been difficult to define.

We assess the classification accuracy over $\mathcal{G}$ with a 20-fold cross-validation, whose results are reported in Fig. 2-(a) as a confusion matrix: we can observe that the classifier is extremely accurate, with 100% true positive in the constant and local classes, 99% for the random and 98% for the global class. The worst case is represented by 95% true positive for the odd class (that represent only 7% of the samples): these very few misclassifications are erroneously attributed to local, global or random classes, and additional series definition (e.g., to compensate for wrong endianness) could help reducing if needed.

Additionally, Fig. 2-(b) depicts the importance for the most useful features of the classifier (that were early tabulated in bottom part of Fig. 1). Four main takeaways can be gathered from the picture: first, just four features are necessary for a full discrimination; second, as expected features that measure the dispersion (entropy and standard deviation) are prevalent; third, both original and derivative sequences are useful in the detection; fourth, subsequence metrics are highly redundant (i.e., $H(x) = H(y)$, $\sigma_x = \sigma_y$, etc.) and so only one of the two appears in the classification tree.

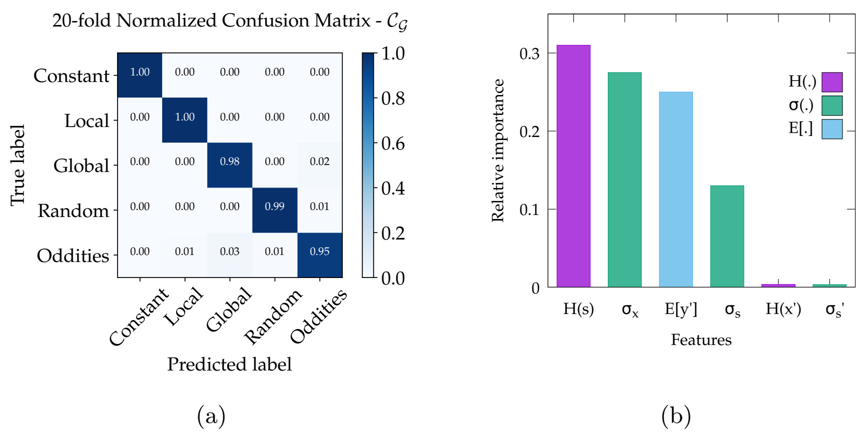
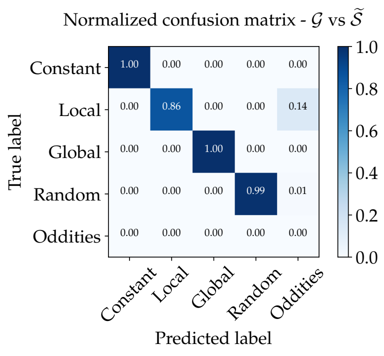


Fig. 2: Validation: (a) Confusion Matrix of 20-fold validation over $\mathcal{G}$ and (b) Relative importance for the most useful features of the classifier.

4.2 Robustness

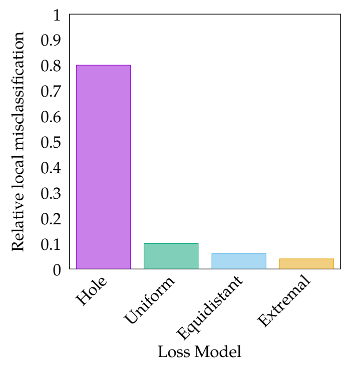
We next assess the robustness of our classifier against packet losses, which may introduce distortion in the measured features. Since the expected values in the ideal conditions are significantly apart, we expect the classifier to be resilient to a high degree of losses. For reason of space, we limitedly consider robustness to losses here and refer the reader to [1] for more details. Without loss of generality, we consider an extreme case where only 80 out of 100 samples are correctly received (i.e., a 20% loss rate). While for simple loss patterns (e.g., uniform i.i.d. losses) it is still possible to analytically derive expected values in closed form, for loss models where losses are correlated, this becomes significantly more difficult. As such, we opt for an experimental assessment of classification accuracy in presence of different synthetic loss models, that we apply to synthetic ideal sequences by purposely discarding a part of the sequences. Specifically, we consider: (i) a **uniform** i.i.d. loss model; (ii) a **hole** model where, starting from a random point in the sequence, 20% of consecutive samples are removed; (iii) an **extreme** model where we remove 20% of the initial values (or equivalently the final 20% of the sequence); and finally (iv) an **equidistant** model where losses start at a random point and are equally spaced over the sequence.

We apply these loss models to obtain a synthetic loss dataset $\widetilde{\mathcal{S}}$ and assess the accuracy of the previously validated model, i.e., the one trained on the real lossless dataset $\mathcal{G}$. Specifically, for each loss model we generate 5,000 loss sequence pattern, for an overall of 20,000 test cases. Results of these experiments are reported in Fig.3. In particular, the confusion matrix reported in Fig.3-(a) shows the aggregated results over all loss models: we can observe that most of the classes have a true positive classification of 99% or 100% even in presence of 20% packet losses, and irrespectively of the actual loss pattern.



	Constant	Local	Global	Random	Oddities
Constant	1.00	0.00	0.00	0.00	0.00
Local	0.00	0.86	0.00	0.00	0.14
Global	0.00	0.00	1.00	0.00	0.00
Random	0.00	0.00	0.00	0.99	0.01
Oddities	0.00	0.00	0.00	0.00	0.00

(a)



(b)

Fig. 3: Robustness: (a) Confusion Matrix of a classifier trained over a real lossless sequence $\mathcal{G}$ and tested over synthetic lossy sequences $\widetilde{\mathcal{S}}$ and (b) breakdown of the (local,odd) misclassification (14%) for the different loss models.

Additionally, we observe that in the case of the *local class*, only 86% of the sequences are correctly classified, whereas 14% of the local sequences in presence of heavy losses are erroneously classified as being part of the "odd" behavior class. Fig.3-(b) dig further the reasons of this discrepancy, showing that the misclassification mostly happens for the *hole* loss model, while in the other cases is a very rare event. Recalling the odd behavior early shown in the top right plot of Fig. 1, we notice that this model induces a gap in the sequence, which is possibly large enough to be statistically similar to cases such as load balancing, where the sequence alternates among multiple counters.

Overall, we find the classifier to be robust to very high loss rates and, with a single exceptions, also invariant to the actual loss pattern – which is a rather desirable property to operate the classifier into a real Internet environment.

4.3 Lightweight

We finally assess how large the number of samples N needs to be to have accurate classification results. In principle, features tabulated in Fig.1 are diverse enough so that we expect high accuracy even for very small values of N.

To assess this experimentally, we take the real loss less dataset $\mathcal{G}$ and only consider that we have at our disposal only $N' < N$ out of the $N = 100$ samples gathered in the experiment. For each value of N', we perform a 20-fold cross validation, training and validating with N' samples. We start from a minimum of $N' = 10$ (i.e., 5 packets per host) up to the maximum of $N = 100$ (i.e., 50

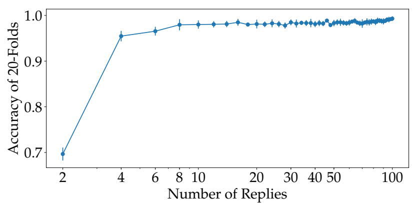


Fig. 4: Lightweight: Accuracy as a function of the sample set size

probes per host) samples. Fig.4 clearly shows that accuracy is already very high[4] at 0.95 when $N' = 4$ and exceeds 0.99 when $N = 100$.

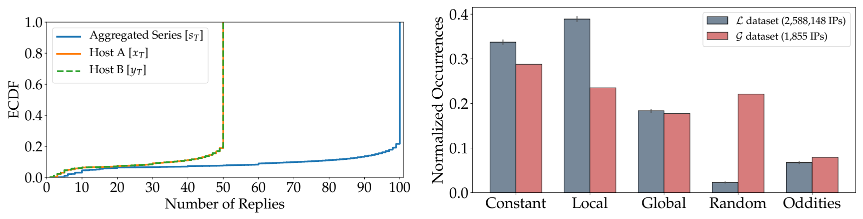
At the same time, these results are gathered in the context of an ideal sequence, whose replies are collected in order and without losses. It is intuitive that there is a trade-off between robustness against losses and lightweight: we expect the accuracy to degrade in presence of losses and reordering for short $N < 4$ probe sequences, whose detailed analysis we leave for future work.

5 Internet measurement campaign

Measurement. Finally, we apply our classifier in the wild, and perform a large scale Internet measurement campaign. We want to avoid putting stress on the infrastructure carrying a full Internet census: as we aim at providing an accurate picture of the *relative* popularity of IP-ID implementations on the Internet, it suffices to collect measurements for a large number of targets, namely 1 alive IP/32 host per each /24 prefix. We observe that, while our classifier is able to perform a very accurate classification even with few samples, we need to deal with loss rates, which is unknown a priori. Hence, we prefer for the time being use a simple and conservative approach and select $N = 100$ samples that is very accurate also in presence of very high loss rates. We instead leave the use of an adaptive sample set size (i.e., start with $N = 10$ and re-probe the same target with a larger N only in case of losses) for future work.

For the targets selection, we rely on the public available hitlist regularly published by [12], comprising 16 millions of targets IP/32. The hitlist contains targets for all /24, including those who have never been replied to the probing: excluding them from our target list, leaves us with approximately 6 millions of potential targets. To further reduce the amount of probe traffic, we then decide to be even more conservative: we preliminary probe the remaining targets sending two ICMP echo requests, and include in our final target list the approximately 3,2 millions responsive hosts (in line with [7, 28]).

[4] Notice that even in the extreme case with as few as $N' = 2$ packets, random and constant classification are correctly labeled, whereas the remaining global vs local cannot be discriminated, yielding to 0.70 accuracy in the $\mathcal{G}$ set.



(a) (b)

Fig. 5: Internet campaign: (a) ECDF of the number of packet replies and (b) Normalized classes occurrences for the training $\mathcal{G}$ and Internet-scale $\mathcal{L}$ dataset

We send a batch of N=100 back-to-back probe packets to each target, but otherwise probe at a low average rate, so that we complete a /24 census in about 3 days. Fig.5-(a) shows the empirical cumulative distribution function (ECDF) of received packets at our VPs. We observe that we receive almost all the replies from most of the targets: the 90% (80%) of the targets answer to more than 40 (all) packets per each host, corresponding to a 20% (0%) loss scenario. A large plateau in the CDF also indicate that the distribution is bi-modal, i.e., the remaining hosts generally reply with very few packets (e.g., 10 or less per each VP or over 90% loss rate). This suggests that future campaigns could be safely conducted with a smaller $N' < N$: indeed, while for hosts with over 90% loss rates there are few hopes of correct classification, we expect the classification for hosts with 20% or lower loss rates to be reliable with as few as $N' = 10$ packets.

To provide accurate classification results, in light of our robustness analysis, we limit our attention to the 2,588,148 hosts for which we have received at least $N = 80$ packets. We make this large-scale dataset, that we annotate with classification results and denote with $\mathcal{L}$, available for the community at [1].

Comparison with related work. We apply the classification to batches of 100,000 hosts, and for each class c, compute the relative breakdown of the class in that batch $\hat{n}_c = n_c / \sum_i n_i$, evaluating the confidence intervals of $\hat{n}_c$ over the different batches. Results are reported in Fig.5-(b), where we additionally report the breakdown in our $\mathcal{G}$ training set comprising just 1,855 population samples: it can be seen that while $\mathcal{G}$ has no statistical relevance for the census, it is not affected by class imbalance and thus proves to be a good training set.

Results are particularly interesting to put in perspective with current literature knowledge. Specifically, past work [6, 9, 20, 27] consistently reported the global counter to be more widespread: in 2003, [20] reported that 70% (over 2000 probed targets) were using an IP-ID counter (global or local implementation); in 2005, [6] reported that 38% (over 150 hosts) used a global IP-ID; in 2006, [27] affirms the global implementation to be the most common assignment policy (among 3 behaviors); in 2013, [9] asserts 57% (over 271 DNS TLD servers) to implement global counter. On the contrary, we find that only 18% (over 2,5 million targets) are still using global counter implementation: this in line with 2017 results that reports slightly more than 16% global IP-IDs [23] (whose main aim is to detect censorship in the Internet). While this decreasing trend is possibly affected by the comparably smaller population size of early studies, however we believe this trend to be rooted into OS-level changes in IP-ID policy implementations: e.g., Linux and Solaris, which previously adopted a global counter, for security reasons later moved to a local counter implementation [10].

The sole quantitative assessment of IP-ID behavior over multiple classes dates back to 2013, and is limited to 271 Top Level Domains TLDs probed by [9] (whose main aim is to propose practical poisoning and name-server blocking attacks on standard DNS resolvers, by off-path, spoofing adversaries). In particular, the 2013 study (our census) finds 57% (18%) global, 14% (39%) local and 9% (34%) constant IP-IDs, which testify of a significant evolution. Additionally, [9] suggests that 20% of DNS TLD exhibit evidence of "two or more sequential sequences mixed up, probably due to multiple machines behind load balancer", much larger than the 7% fraction of the larger "odd" class (including but not limited to load balance) that we find in this work. Finally, despite 2012 recommendations [10], the percentage of random IP-ID sequence was (and remains) limited 1% (2%).

6 Conclusions

This work presents, to the best of our knowledge, the first systematic study of all IP-ID behaviors prevalence in the current Internet. Our first contribution is to devise an accurate, lightweight and robust classifier: accuracy of the classifier follows from a principled definition of the statistical features used to succinctly describe the IP-ID sequence; robustness is a consequence of this choice, as features remains wide apart even under heavy losses.

Our second contribution is to carry on a manual investigation effort for a moderate size dataset coming from real Internet measurements: this valuable ground truth allow us to adopt a supervised classification techniques to train a model able not only to detect well-defined behaviors, but also to correctly recognize a wide range of odd behaviors.

Our final contribution is to apply this classification to an Internet-scale measurement campaign, obtaining a very accurate picture of nowadays IP-ID behavior prevalence, which we release as open dataset at [1]. This dataset is possibly instrumental to other relevant work in the measurement field [2, 6, 23, 25], and by updating and consolidating the scattered knowledge [6, 9, 20, 23, 27] of IP-ID prevalence, contributes in refining the current global Internet map.

Acknowledgments

This work has been carried out at LINCS (`http://www.lincs.fr`) and benefited from support of NewNet@Paris, Cisco Chair "Networks for the Future" at Telecom ParisTech (`http://newnet.telecom-paristech.fr`).

References

1. https://perso.telecom-paristech.fr/~drossi/ip-id/.
2. S. M. Bellovin. A technique for counting NATted hosts. In *Proc. IMW*, 2002.
3. A. Bender, R. Sherwood, and N. Spring. Fixing ally's growing pains with velocity modeling. In *Proc. ACM IMC*, 2008.
4. R. Beverly, M. Luckie, L. Mosley, and K. Claffy. Measuring and characterizing IPv6 router availability. In *PAM*, 2015.
5. R. Braden. *RFC 1122, Requirements for Internet Hosts – Communication Layers*, 1989.
6. W. Chen, Y. Huang, B. Ribeiro, et al. Exploiting the IPID field to infer network path and end-system characteristics. In *PAM*, 2005.
7. A. Dainotti, K. Benson, A. King, B. Huffaker, E. Glatz, X. Dimitropoulos, P. Richter, A. Finamore, and A. C. Snoeren. Lost in space: Improving inference of IPv4 address space utilization. *IEEE JSAC*, 2016.
8. K. S. G. Pelletier. *RFC 5225, RObust Header Compression Version 2 (ROHCv2):Profiles for RTP, UDP, IP, ESP and UDP-Lite* , 2008.
9. Y. Gilad and A. Herzberg. Fragmentation considered vulnerable. *ACM TISSEC*, 2013.
10. F. Gont. RFC 6274, Security assessment of the internet protocol version 4. 2011.
11. F. Gont. RFC 7739, Security implications of predictable fragment identification values. 2016.
12. J. Heidemann, Y. Pradkin, R. Govindan, C. Papadopoulos, G. Bartlett, and J. Bannister. Census and survey of the visible internet. In *Proc. ACM IMC*, 2008.
13. A. Herzberg and H. Shulman. Fragmentation considered poisonous, or: One-domain-to-rule-them-all. org. In *IEEE CCNS*, 2013.
14. Idle scanning and related IPID games. https://nmap.org/book/idlescan.html.
15. S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley. Measurement and classification of out-of-sequence packets in a tier-1 IP backbone. *IEEE/ACM TON*, 2007.
16. K. Keys, Y. Hyun, M. Luckie, and K. Claffy. Internet-scale IPv4 alias resolution with MIDAR. *IEEE/ACM TON*, 2013.
17. A. Klein. OpenBSD DNS cache poisoning and multiple O/S predictable IP ID vulnerability. Technical report, 2007.
18. W.-Y. Loh. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2011.
19. M. Luckie, R. Beverly, W. Brinkmeyer, et al. Speedtrap: internet-scale IPv6 alias resolution. In *Proc. ACM IMC*, 2013.
20. R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. User-level internet path diagnosis. *ACM SIGOPS Operating Systems Review*, 2003.
21. J. C. Mogul and S. E. Deering. *RFC 1191, Path MTU discovery*, 1990.
22. S. Mongkolluksamee, K. Fukuda, and P. Pongpaibool. Counting NATted hosts by observing TCP/IP field behaviors. In *Proc. IEEE ICC*, 2012.
23. P. Pearce, R. Ensafi, F. Li, N. Feamster, and V. Paxson. Augur: Internet-wide detection of connectivity disruptions. In *IEEE SP*, 2017.
24. J. Postel. *RFC 791, Internet protocol*, 1981.
25. N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies with rocketfuel. *IEEE/ACM TON*, 2004.
26. J. Touch. *RFC 6864, Updated Specification of the IPv4 ID Field*, 2013.
27. M. A. West and S. McCann. RFC 4413, TCP/IP field behavior. 2006.
28. S. Zander, L. L. Andrew, and G. Armitage. Capturing ghosts: Predicting the used IPv4 space by inferring unobserved addresses. In *Proc. ACM IMC*, 2014.