A First Look at the Prevalence and Persistence of Middleboxes in the Wild

Korian Edeline, Benoit Donnet
Université de Liège, Montefiore Institute – Belgium

***Abstract*—Recent years have seen an uprise in the development of middleboxes functionalities (CGNATs, proxies, accelerators, etc), participating so in the ossification of the Internet. In parallel, various solutions have been developed to detect or circumvent unwanted middleboxes interferences such as UDP-based middlebox-proof transports (Google's QUIC, PLUS), middlebox-proof extensions to TCP (HICCUPS, TCPcrypt), and middlebox traversal mechanisms (STUN, ICE, PLUS) [1].**

All those solutions make the assumption of ubiquitous middleboxes. However, a view of their actual deployment in the wild, in IPv4 wired networks, is missing. In particular, knowing how autonomous systems (ASes) deploy middleboxes in terms of prevalence and persistence would provide additional relevant information to Internet topology models. In this paper, we aim at filling this gap. Based on a large-scale measurement campaign, we highlight different characteristics of middlebox deployment within ASes to elicit middleboxes profiles.

I. Introduction

Nowadays, the standard and well-known description of the TCP/IP architecture (i.e., the end-to-end principle) is not anymore applicable in a wide range of network situations. Indeed, enterprise networks, WiFi hotspots, and cellular networks usually see the presence of *middleboxes* being part of the network architecture in addition to traditional network hardware [2]. A middlebox is a network device inspecting, filtering, or even modifying packets that traverse it. It performs actions on a packet that are different from standard functions of an IP router.

Recent papers have shed the light on the deployment of those middleboxes. For instance, Sherry et al. [2] obtained configurations from 57 enterprise networks and revealed that they can contain as many middleboxes as routers. Wang et al. [3] surveyed 107 cellular networks and found that 82 of them used NATs. D'Acunto et al. [4] analyzed P2P applications and found that 88% of the participants in the studied P2P network were behind NATs. Middleboxes may be deployed for several reasons, typically security (e.g., IDS, NATs, firewalls) and network performance (e.g., load balancer, WAN optimizer).

However, if there is a widespread usage of middleboxes, they come with important drawbacks. Indeed, it has been shown that middleboxes have a negative impact on the TCP protocol (and its extensions) evolution [5], [6]. Middleboxes may modify, filter, or drop packets that do not conform to expected behaviors. As a consequence, the Internet faces a kind of ossification due to the difficulties of proposing new transport protocols.

Researchers, when designing a new protocol, have thus to cope with a middlebox-full Internet. Each new mechanism has to be certified as middlebox-proof [6], [7]. For those researchers, a summary of the potential middlebox network interferences would be a valuable asset as they could easily confront their new protocol with potential issues caused by middleboxes.

Recent years have seen the emergence of new network measurement techniques that aim to achieve this objective (i.e., revealing the presence of middleboxes and characterizing their behavior). Medina et al. [8] report one of the first detailed analysis of the interactions between transport protocols and middleboxes. They rely on active probing with `tbit` and contact various web servers to detect whether Explicit Congestion Notification (ECN), IP options, and TCP options can be safely used. The `TCPExposure` software developed by Honda et al. [5] is closest to `tracebox` [9]. It also uses specially crafted packets to test for middlebox interference. Wang et al. [3] analyzed the impact of middleboxes in hundreds of cellular networks. This study revealed various types of packet modifications. More recently, Craven et al. [10] proposed TCP HICCUPS to reveal packet header manipulation to both endpoints of a TCP connection. HICCUPS works by hashing a packet header and by spreading the resulting hash into three fields (in case one is changed).

However, to the best of our knowledge, nothing has been really done to characterize the middleboxes deployment in terms of *prevalence* for ASes (e.g., if a firewall is setup, do all traffic goes through that firewall?)[1] and *persistence* over time (e.g., is a middlebox up and running all the time? Or do we observe any dynamics as for IP networks? [11], [12], [13]). This would be valuable in any effort in modeling middleboxes interference and in getting a first look at how ASes actually deploy middleboxes.

This is exactly what we want to tackle here. In this paper, based on a large dataset we collected with `tracebox` on IPv4 wired networks, we are able to investigate middleboxes prevalence and persistence. Analyzing the data, we highlight different characteristics of middlebox deployment within autonomous systems and elicit middlebox profiles. For instance, on the contrary to enterprise networks [2], our observations show that, in general, ASes do not deploy that much middleboxes compared to standard Layer-3 devices.

[1] It is worth noticing that prevalence has already been investigated by Sherry et al. [2], but at the enterprise level. In this paper, we propose a broader vision of middlebox deployment by considering ASes at large.

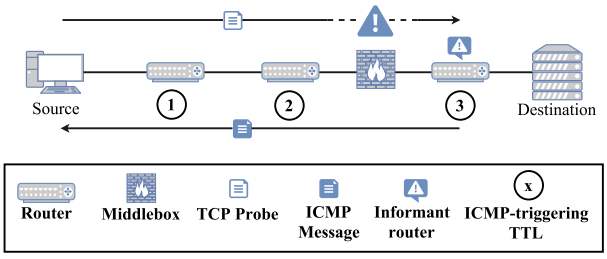


Fig. 1. Middlebox detection with `tracebox`.

Those middleboxes do not affect a large portion of the paths (in 20% of the cases, a middlebox may affect more than half of paths traversing an AS) but are, in the majority, deployed at the AS border. Finally, we also demonstrate that the majority of middleboxes are stable over time (i.e., do not exhibit dynamic features). Our dataset is available online.[2]

The remainder of this paper is organized as follows: Sec. II provides details on our measurement campaign; Sec. III explains how we preprocess the large amount of collected data; Sec. IV discusses results about middleboxes prevalence while Sec. V focuses on middleboxes persistence over time; finally, Sec. VI concludes this paper by summarizing its main achievements and discussing potential future research directions.

II. Dataset

This section describes the dataset we collected. In particular, Sec. II-A describes `tracebox`, the tool we used for collecting data, while Sec. II-B provides details about our measurement methodology and general statistics.

A. `tracebox`

To reveal the presence of middleboxes along a path, we use `tracebox` [9], an extension to the widely used `traceroute` [14].

`tracebox` mechanism is illustrated in Fig. 1. It relies on RFC1812 [15] and RFC792 [16] stating that the returned ICMP `time-exceeded` message should quote the IP header of the original packet and respectively the complete payload or the first 64 bits. `tracebox` uses the same incremental approach as `traceroute`, i.e., it sends packets with different IP, UDP, or TCP fields and options with increasing TTL values. By comparing the quoted packet to the one sent, one can highlight the modifications and the initial TTL value allows us to localize the two or more hops between which the change took place. In Fig. 1, the source sends a TCP packet, `a`, with IP-TTL of 3. The middlebox, located between hop 2 and hop 3, modifies (for instance) the TCP Initial Sequence Number (ISN) and forwards the rewritten packet, `b` to the next hop. When the router located at hop 3 receives the packet, the TTL has expired and it sends back to the packet source an ICMP `time-exceeded` packet `c` containing packet `b` as a payload. When the source receives it, it is able to compare packet `a` and the payload of packet `c` to detect any changes and the initial

[2]https://observatory.mami-project.eu/

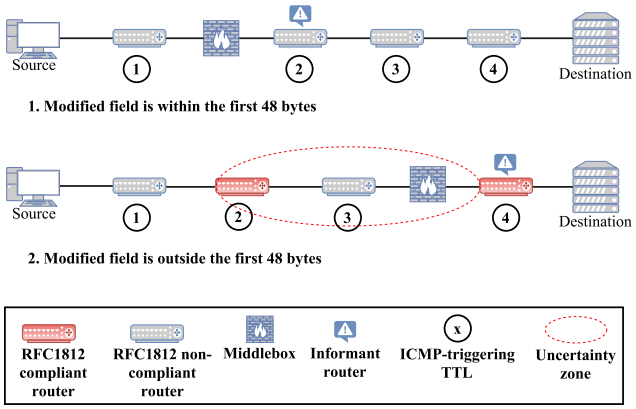


Fig. 2. Uncertainty zone when locating middleboxes.

TTL value, i.e., 3, allows `tracebox` to bound the middlebox location. The router that reveals packet modifications in the ICMP `time-exceeded` message is called the *informant router*.

Depending on whether or not routers along the path implement RFC1812 [15], that recommends to quote the entire IP packet in the returned ICMP, `tracebox` locates precisely the source of the modification or introduces a location uncertainty we called *uncertainty zone*. The two cases are shown in Fig. 2. In the first case, the informant router (at TTL distance of 2 from the vantage point in Fig. 2, part 1) sends an ICMP `time-exceeded` message that highlights a modification of a field that lies within the first 48 bytes (i.e., the IP header and the first 8 bytes of the transport header, e.g., the TCP ports and Sequence Number). Because all routers include at least these fields in the ICMP payload, we are able to locate the middlebox between the informant router, not included, and the previous router (at TTL distance of 1 from the source in Fig. 2). However, when the modification performed by the middlebox is outside the first 48 bytes (i.e., after the first 8 bytes of the transport header, e.g., from the Acknowledgment Number to the end of the TCP header), it can only be revealed by a RFC1812-compliant router. Therefore, an uncertainty zone may appear between the informant router, not included, and the previous RFC1812-compliant router. In Fig. 2, part 2, the uncertainty is located between TTL 2 (from the vantage point) and TTL 4 (not included). Sec. II-B will evaluate the probability of encountering a RFC1812-compliant router along the path, while Sec. III will explain how one could fix difficulties related to uncertainty zones.

B. Measurement Methodology

We deployed `tracebox` on wired IPv4 networks via PlanetLab. We selected the maximal number of nodes available for each campaign (between 108 and 129). Destinations have been selected using the top 1M Alexa website that we resolved once to 594,241 unique addresses beforehand. We conducted 14 campaigns over nine different ports (80, 8080, 8000, 8800, 443, 53, 12345, 1228, 34567) with TCP `SYN` probes

Campaign ID	Port	Raw Data #IPs	#ASes	#Probes	Middleboxes #Paths w/MB	#ASes w/MB
1	80	886,065	2,953	40,392,061	2,175,335	337
2	80	886,263	2,955	42,774,781	2,382,938	347
3	8000	816,656	2,891	24,860,295	1,773,409	252
4	80	887,171	2,950	41,767,341	2,346,832	334
5	8080	816,192	2,897	24,252,300	1,667,765	226
6	8800	820,653	2,889	38,750,758	1,673,998	241
7	443	856,918	2,938	41,590,151	3,415,643	364
8	12345	813,152	2,880	39,234,092	2,286,052	311
9	8080	813,955	2,895	22,466,692	1,609,383	220
10	80	884,808	2,955	42,866,154	2,369,670	342
11	1228	812,213	2,885	39,489,438	2,282,698	329
12	443	882,658	2,955	41,593,420	3,454,363	361
13	34567	820,305	2,893	39,225,840	1,806,605	269
14	53	820,698	2,887	39,202,907	2,784,658	260
HTTP		930,842	2,969	250,983,908	5,696,282	510
non-HTTP		888,596	2,939	267,482,322	3,011,418	368
Total		948,457	2,977	518,466,230	5,832,789	661

TABLE I
GENERAL STATISTICS ABOUT DATA COLLECTED WITH TRACEBOX, AFTER FILTERING, BEFORE PRE-PROCESSING.

including TCP options (MSS and SACKP) for a period of two months between March, 3rd and May, 8th 2016, each campaign lasting between three and seven days. The total amount of data collected corresponds to 1.3TB. This dataset is available online. [3]

Once the raw data collected, we selected the nodes that remained available during all campaigns (89 PlanetLab nodes). We also filtered out PlanetLab-related errors and kept only probes that reached the destination or that expired or triggered an ICMP message after at least 10 hops. Doing so, we obtained an exploitable dataset made of 518 millions probes. 34% of those probes reached the destination, 64% expired, and 2% triggered an ICMP message before reaching the destination. The high amount of timeout is due to probes with non-HTTP ports that are dropped by a firewall before reaching the destination. From this filtered dataset, we extracted 38 millions observations of middlebox behavior (i.e., a single modification, addition, or deletion of single field of a probe, on a single path, in the course of one campaign). We note that we did not witness any significant difference in middlebox behavior towards HTTP and non-HTTP probes, apart from the fact that HTTP probes are less likely to be blocked, and thus are able to highlight more middleboxes. For the remaining of this paper, we process HTTP and non-HTTP evenly, at the exception of the persistence analysis (Sec. V).

During the entire measurement campaign, we observed 948,457 different responsive hops (excluding vantage points and targets addresses), scattered over 2,977 different ASes. The most represented ASes are Cogent (35.7% of all addresses – Tier 1 network), CenturyLink (10.6% – Tier 1 network), Telia Carrier (6.3% – Tier 1 network), NTT (3.4% – Tier 1 network), Rackspace (1.8% – cloud services), Level3 (1.6% – Tier 1 network), and Chinanet (1.5% – Chinese ISP). The corresponding addresses are geographically distributed in North America (40.4%), Europe (37.5%), Asia (18.7%), Latin America and Caribbean (2.7%), and Africa (0.7%) according to the regional Internet registries. The same addresses were registered under 189 different country codes. Table I provides additional general statistics about data collected. In particular, the last two columns show the importance of middleboxes in the dataset.

[3]https://observatory.mami-project.eu/

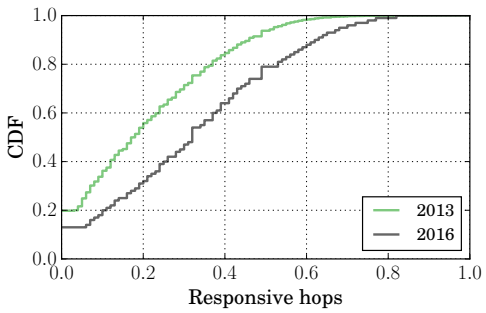


Fig. 3. Proportion of RFC1812-compliant routers on a path. The "2013" curve (green) is from Detal et al. [9].

Fig. 3 shows the proportion of RFC1812-compliant routers (the horizontal axis) as a CDF. A value of 0, on the horizontal axis, corresponds to paths that contained no RFC1812-compliant router. On the other hand, a value of 1 corresponds to paths made only of RFC1812-compliant routers. In particular, in 2013 (green curve), in 80% of the cases, a path contained at least one router that replies with a *Full ICMP*. Nowadays (2016 – black curve), the situation is clearly better. Roughly, in 90% of the cases, a path at least contain one RFC1812-compliant router. In addition, in half of the cases, 30% of routers along a path are RFC1812-compliant routers. This means that, nowadays, tracebox has the potential to reveal much more modifications by upstream middleboxes. It is also worth to notice that the more RFC1812-compliant routers, the less uncertainty zones.

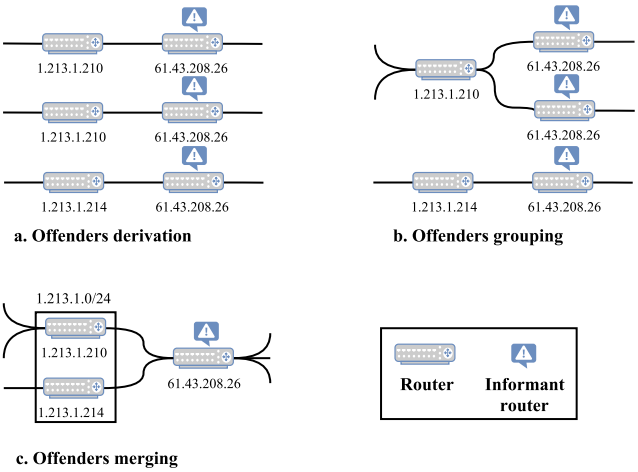


Fig. 4. An example of probes pre-processing.

III. Pre-processing

Before studying middleboxes prevalence and persistence over time, the dataset must be cleaned through a 3-step preprocessing, as illustrated in Fig. 4. The objective beyond the preprocessing is to merge multiple `tracebox` observations of a given middlebox into a single identifier. The first step (Sec. III-A) allows to identify a so-called *offender*, i.e., the router preceding the middlebox on a given path. Second, offenders are grouped together to obtain the profiles of observed middleboxes behavior (Sec. III-B). Finally, offenders are aggregated into unique middleboxes (Sec. III-C).

A. Offender derivation

First, we assign a label to each observation of middlebox behavior. As middleboxes can be located between two routers with consecutive ICMP-triggering TTLs (i.e., there is no uncertainty zone), we arbitrarily choose to label a middlebox with the IP address of the router preceding the middlebox on the path. This router is called *offender*. This is the perfect case as there is no uncertainty zone and the location of the middlebox is, therefore, certain. However, when there is an uncertainty zone, heuristics must be applied to narrow, as much as possible, the zone size so that we are able to identify an offender and, thus, label the middlebox. In addition to identifying the offender IP address, we also perform IP2AS mapping, using Team CYMRU [17], on the offender IP address. This process of identifying the offender is called *offender derivation*. Algorithm 1 provides a high level view of the offender derivation heuristics.

The trivial case is when there is no uncertainty zone (lines 9 → 11 on Algorithm 1). In this configuration, we pick the address of the router preceding the informant router as the offender IP address (in Fig. 2, part a, offenders are identified for three observations[4]). When there is an uncertainty zone, we attempt to apply different heuristics. First, if the router at the informant TTL minus one or two did not respond to the probe (i.e., three consecutive timeouts for a single TTL value), we pick the previous router, respectively at offender TTL minus two or three, as the offender (lines 15 → 23 in Algorithm 1). Second, we map the IP addresses of the routers in the uncertainty zone to AS numbers and select the most frequent one (lines 28 → 31 in Algorithm 1). We choose to pick the first router that belongs to the major AS as the offender, but only if it was also used for labeling via the first heuristic or the trivial case. Otherwise, we keep only the offender AS. Finally, if neither heuristics succeeded, we select the first router of the uncertainty zone as the offender. If this router was not used for labeling by any other heuristics, we disregard the offender IP address and AS number for this observation (lines 33 → 42 in Algorithm 1).

Note that our dataset contains 2.25% (21,330) of addresses from non publicly-routable address spaces (10.0.0.0/8,

[4]An "observation" here stands for the identification of a packet modification attributable to a middlebox

Algorithm 1 Offender derivation

```
1  def offender(probe):
2
3      u = probe.uncertainty
4      informant_ttl = probe.informant.ttl
5      previous_hop  = probe.hops[informant_ttl - 1]
6      offender_ip, offender_as = '*', '*'
7
8      # No uncertainty zone
9      if u == 1:
10         offender_ip = previous_hop.addr
11         offender_as = previous_hop.asn
12
13     # Heuristic 1: No answer at TTL-1
14     informant_minus2 = probe.hops[informant_ttl - 2]
15     elif u > 1 and previous_hop.addr == '*':
16         offender_ip = informant_minus2.addr
17         offender_as = informant_minus2.asn
18
19     # Heuristic 1bis: No answer at TTL-2
20     elif u > 2 and informant_minus2.addr == '*':
21         informant_minus3 = probe.hops[informant_ttl - 3]
22         offender_ip = informant_minus3.addr
23         offender_as = informant_minus3.asn
24
25     # Heuristic 2:
26     # First occurence of major AS in uncertainty zone
27     else:
28         uzone_start = informant_ttl - u
29         uzone_end   = informant_ttl
30         uzone       = probe.hops[uzone_start:uzone_end]
31         as_uzone    = [hop.asn for hop in uzone]
32
33         major_as = most_common(as_uzone)
34         if as_uzone.count(major_as) > 1:
35             as_index = as_uzone.index(major_as)
36             offender_ip = uzone[as_index].addr
37             offender_as = as_uzone[as_index].addr
38
39         # Heuristic 3: Start of uncertainty zone
40         else:
41             offender_ip = probe.hops[uzone_start]
42             offender_as = probe.ases[uzone_start]
43
44     return offender_ip, offender_as
```

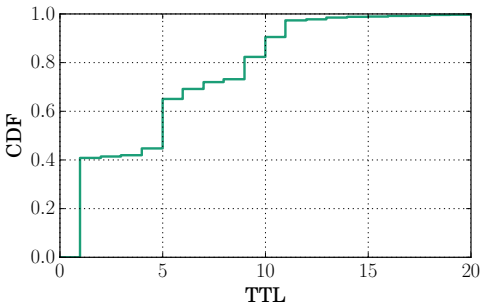


Fig. 5. Size of Uncertainty zones.

172.16.0.0/12, 192.168.0.0/16), 0.1% (905) from CGNAT shared address space (100.64.0.0/10), and 2.18% (20,669) that were not mapped to an AS. We keep such addresses as offenders only if the routers at the ends of the unresolved address zone are mapped to the same AS.

In parallel, we compute the position of the offender within the AS. If the routers that precedes and follows the offender are parts of the offender AS, we mark the middlebox position as internal to the AS. If either one of them is not part of the offender AS, we set the position to border of the AS.

Fig. 5 gives the distribution of the uncertainty zone, as a cumulative mass, in our dataset. We see that, in 41% of the cases (15.5 millions over the 38 millions of observations), there is no uncertainty zone (size of 1). Moreover, in 60% of the cases, the uncertainty zone has a size less or equal to five. In those cases, we expect the location heuristics to be reasonably accurate. Finally, as we have observed an increase in RFC1812-compliant router deployment (see Fig. 3) we expect, in the near future, that the uncertainty zone size will decrease more and more, leading to a better location of middleboxes.

At the end of this first step, we were able to derive offenders ASes for 99% of the observations, and to derive offenders IP addresses for 20 millions (52%) observations. The first set of observations (i.e., offenders with AS numbers) is kept for AS-level analysis, the second (i.e., offenders with IP addresses) serves as input to the second step of the preprocessing.

B. Offender grouping

Next, single path observations are grouped together based on the previously derived offenders. This is illustrated on Fig. 4, part *b*. In the first stage, offenders were derived for three observations. During this second stage, offenders grouped into two offender profiles (1.213.1.210 and 1.213.1.214 on Fig. 4, part *b*) based on the address, and regardless of the informant.

The goal of this step is to obtain a profile of observed middlebox behavior, to cross-check offender derivation heuristics and to rule out observations conflicts. Indeed, an offender profile contains several relevant pieces of information, such as the amount of paths it affects (see Sec. IV), its dynamics (see Sec. V), and the set of next hops.

As mentioned in the previous section, the observations for which offenders were obtained via heuristics 2 or 3 must correlate with at least one observation for which offenders were derived with heuristics 1 or the trivial case. To ensure this, the modification of fields and value in each observations must corresponds, i.e., it concerns the exact same fields and similar values. With the exception of modifications that are not observable on certain path because of the absence of any RFC1812-compliant router after the offender. We also make sure that an offender derived with any heuristic but the trivial case is backed up by at least ten single observations to avoid reporting inconsistent middlebox behavior.

Then, we solve the remaining conflicts (that can be related to errors in traces, punctual network conditions causing probe loss, a middlebox reconfigured during a campaign, or the inherent `tracebox` uncertainty) by applying a threshold of one twentieth on the variability of the observed positions and modified fields. For example, when we have different offender positions for the same offender, we pick the most widespread if it accounts for at least 95% of this offender's observations and that it remains constant during each campaign. We are able to obtain a constant position for 89% of offenders while 2% appear to have moved between campaigns. The remaining 9% of offenders position are either unresolved conflicts or inconclusive. For addresses whose ASes were resolved during the offender derivation step, we note the absence of conflict on the AS resolutions.

This second step allowed us to identify 8,322 offenders in our dataset.

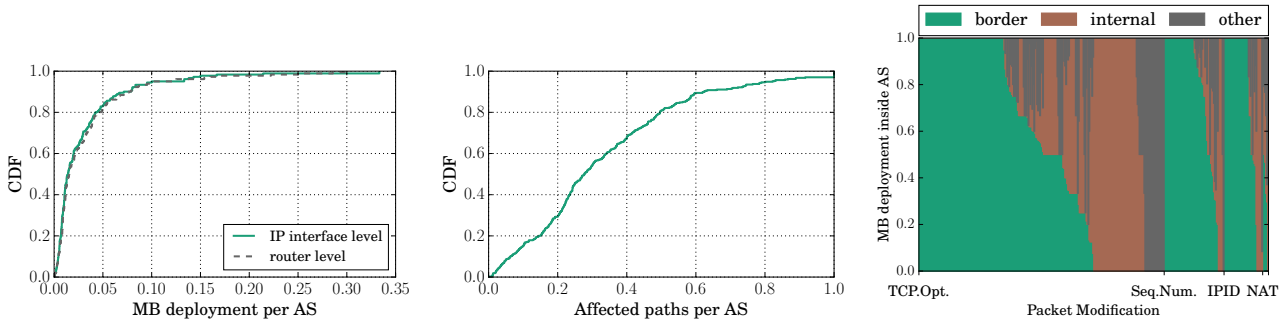
C. Offender merging

Finally, we attempt to merge offenders when we believe they refer to the same middlebox. The merging is done with IP prefixes, as illustrated in Fig. 4, part *c*, where offenders 1.213.1.210 and 1.213.1.214 are merged in prefix 1.213.1.0/24. The following criterions are considered for merging:

- All offenders IP addresses are part of the same subnetwork (/24).
- All offenders were observed applying the same set of modifications, or subsets of modifications in case certain were not observable (i.e., fields outside the first 48 bytes with all routers located after the offender being RFC1812 non-compliant).
- For all affected paths of each offender, the set of IP addresses located after the offender (at offender TTL plus one) are equal.

If all criterions are met for groups of two or more offenders, they are merged. During this step, 515 offenders were aggregated into 198 prefixes. We achieve to recompute the profile of the middlebox, considering all merged offenders as one. We obtain 8,005 aggregated offenders, that we will now use as different middleboxes, distributed in 343 different ASes.

Anecdotically, we observed 7 cases of Multi-Origin AS Conflicts (MOAS) [18] (i.e., addresses parts of the same /24 being mapped to different ASes), but none of them were candidate to offender merging.



(a) Infrastructural deployment of middleboxes (b) Middleboxes popularity (c) AS topological location of middleboxes

Fig. 6. Middlebox prevalence evaluation.

IV. Middleboxes Prevalence

Once the dataset has been cleaned by the 3-step preprocessing (see Sec. III), one can start looking at middleboxes prevalence. We define the *prevalence* of middleboxes for an AS as being the importance of the middleboxes at three levels. First, we envision an infrastructural deployment of the middleboxes, i.e., the proportion of middleboxes deployed by the AS compared to typical Layer-3 devices (IP interfaces or routers). This allows us to know whether ASes are deploying as many middleboxes as Layer-3 devices, as previously stated for enterprise networks [2].

Fig. 6(a) shows the proportion of middleboxes deployed by ASes as a fraction of IP interfaces discovered in our measurement campaign (plain line) and as a fraction of routers (dashed line – the alias resolution has been performed using CAIDA's ITDK dataset [19]). At the AS level, the deployment of middleboxes (compared to Layer-3 devices) is rather marginal[5] compared to Layer-3 devices. In general, less than 5% of the deployed infrastructure is dedicated to middleboxes. For instance, Cogent (the most observed AS in our measurement campaign) deploys between 1% and 1.5% of middleboxes compared to Layer-3 devices.

Second, we consider the middlebox popularity, i.e., the fraction of paths, inside an AS, that is harmed by at least one middlebox (this information is directly derived from step 2 in the preprocessing – see Sec. III-B). A value of 0, for the popularity, would mean that all observed paths traversing the AS do not encounter a middlebox (and, as such, the AS does not deploy any middlebox). On the contrary, a value of 1 means that all observed paths are harmed by a middlebox. In that extreme case, one can say that middleboxes are prevalent as they impact every packet traversing the AS. Fig. 6(b) shows the middlebox popularity (X-Axis) as a cumulative mass. We observe that in 20% of the cases, more than 50% of the AS paths are affected by a middlebox. This suggests thus that, in some cases, middleboxes are prevalent, even if, in general, an AS does not deploy that much middleboxes in its network. However, it is worth noticing that this metric has its inherent limits. For instance, more than 44 millions of paths are traversing Cogent, and we detect middleboxes presence on more than 2 millions of them. This gives a popularity of "only" 5%, while we believe middleboxes are quite prevalent across this AS.

Location	MB count	Percentage
border	4,210	52.6%
internal	2,931	36.6%
other	864	10.8%

TABLE II
Global overview of middleboxes location within an AS.

Finally, following the middlebox popularity, we evaluate where, in its topology, an AS is likely to deploy a middlebox. Two locations are envisioned: (i) at the border of the network (meaning that it is most likely that the middlebox will process every packet entering/leaving the AS network) or (ii) in the core of the AS network (meaning that middleboxes are deployed for very dedicated services and traffic). Table II provides a global overview of middleboxes over the whole dataset. The "other" category corresponds to cases where we were unable to derive the offender position (for instance, because all addresses were non publicly-routable), or where the offender appears as being at the border of the AS for some paths, and AS internal for others. Those two situations appeared in 9.1% of the cases. Finally, a few offenders appeared to have been moved from the border to the core, or vice versa, from one campaign to another but this is a rare case (1.7%). What we observe from Table II is that middleboxes are mainly located at the border of ASes. This is aligned with results presented in Fig. 6(b). Indeed, if the majority of paths, within an AS, are affected by (at least) one middlebox, it is expected to see this middlebox at the ingress (or egress) of the AS, creating so a kind of bottleneck in which the majority of the traffic must go through.

Fig. 6(c) shows the distribution of middleboxes location, split in four categories, per ASes on which they are deployed.

[5]Obviously, the amount of observed middleboxes in our dataset is strongly related to the way we performed the measurement campaign. And, in particular, results are biased due to the target being the Top 1M Alexa web sites. We, however, believe that results given in this paper are a lower bound, giving so a first insight on how ASes deploy and use middleboxes.

The categories are the following: (i) TCP options: This regroups traffic engineering middleboxes that modify, strip, or add TCP options (`MSS` and `SACKP`), that we highlighted based on the value of the actual TCP options, TCP offset, and IP Length, (ii) TCP sequence number modification, security-related middleboxes that sets the TCP initial sequence number to randomly chosen value, (iii) IP-ID modification, middleboxes that sets the IP-ID field to unique non-null value for each transmitted packet, and, (iv) NAT, middleboxes that re-mapped the source port of our probes, often combined with other previously described behavior.

We observed 254 ASes that deploy TCP options middleboxes, among which 88 deploy all of them at their border. 149 ASes deploy at least half of them at the border, 44 ASes deploy all TCP options middleboxes in their core while 20 ASes also deploy such middleboxes, but we cannot conclude on their positioning. 62 ASes are deploying Sequence Number shuffling middleboxes. 31 ASes deploy all those middleboxes at their border, while 48 ASes deploy at least half of their shuffling middleboxes also at the border. Finally, only 4 ASes deploy those middleboxes in their network core. 40 ASes deployed middleboxes that checks for IP-ID uniqueness, 25 of them put all such middleboxes at their border, 5 others in their core. 6 ASes were observed making use of NATs, but without privileged position.

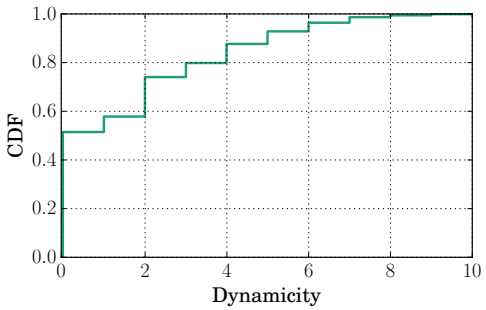
Overall, we found that ASes tend to deploy most of their middleboxes at their border, at the exception of 65 ASes (19% of the ASes with labelled middleboxes) that deploys at least half of their middleboxes in their core.

V. Middleboxes Persistence

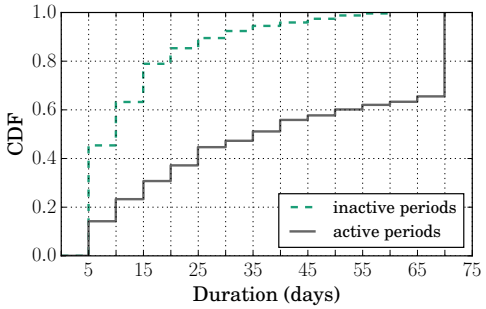
In this section, we analyze the persistence of middleboxes over time. We consider that a middlebox is active during a campaign if at least one of its offender IP address was responsive during this campaign, and that it was used for labeling. We consider a middlebox as inactive if at least one of its offender IP address was responsive and none of them was used for labeling. We take care to not count the cases were the middlebox is not observable because of the absence of any RFC1812-compliant informant router. Finally, a middlebox is considered offline if none of its offender IP addresses were responsive during a whole campaign.

From the set of labelled middlebox, we selected those that were responsive to probes with HTTP and non-HTTP ports, excluding middleboxes located on path portions invisible to certain non-HTTP probes because of port-based blocking, to be able to analyze their persistence through all the campaigns. From the 8,005 labelled middleboxes, we selected 5,888 in this manner.

We compute the dynamics by comparing the set of active campaigns to the set of inactive campaigns. The *middlebox dynamic* is thus the number of times the middlebox switched from active state to inactive and vice versa. We chose to ignore the offline states not to draw conclusions from the absence of observations, to make sure that the address was not simply unobservable (e.g., the probe took a different path).



(a) Global overview, state changes (per middlebox).



(b) Duration (per period, 8,788 active periods, 5,121 inactive ones).

Fig. 7. Middlebox Dynamics.

A value of zero means that a middlebox is constant (i.e., it is always up and running), while a value larger than one provides the number of times a middlebox switches from an active (respectively inactive) state to an inactive (respectively active) state.

Fig. 7 displays the distribution of middleboxes dynamic. In particular, Fig. 7(a) provides a general overview of how middleboxes are affected by dynamics. We see that 51% of the middleboxes are stable over time (i.e., a value of 0), the corollary being that the other half of the middleboxes exhibits a dynamic behavior. The second most frequent value is 2, meaning that a middlebox will switch its state twice over the measurement campaign. The maximum value, quite rare, is ten, suggesting so that some middleboxes are very unstable over time.

Fig. 7(b) gives the states duration in terms of consecutive days for both active (plain line) and inactive (dashed line) periods of middleboxes. To compute this figure, we normalized campaigns durations to five days. The maximum duration for an active period is 70 days (in 38% of the cases), i.e., the whole measurement campaign duration. This corresponds to middleboxes with a dynamic of 0. Moreover, we observe that 50% of the active periods are longer than 35 days (half of the campaigns). Inversely, we notice that 44% of inactive periods are short-lived, lasting only 5 days, while 20% of inactive periods were longer than 20 days.

Globally, we showed on the one hand that the largest part of middleboxes (more than 75%) tends to be constantly active,

or presenting few short periods of inactivity. On the other hand, a minority of middleboxes are more dynamic, alternating between active and inactive states 3 times or more.

Investigating the cause of middleboxes dynamic is left for future work.

VI. Conclusion

Middleboxes are becoming more and more popular in particular in enterprise networks. Those middleboxes are supposed to be transparent to users but it has been shown the contrary. In particular, they impact the TCP protocol and its extensions. However, little is actually known on how middleboxes are deployed by autonomous systems (ASes).

This is exactly what we tackled in this paper. Based on a large-scale measurement campaign targeting Top 1M Alexa Website, we studied how prevalent middleboxes are for ASes and how persistent they are over time. Our observations demonstrate that ASes do not deploy that much middleboxes compared to standard Layer-3 devices. Those middleboxes do not affect a large portion of the paths but are, in the majority, deployed at the AS border. Finally, we also demonstrate that the majority of middleboxes is stable over time.

In this paper, we focused on IPv4 wired networks. Future works should highlight whether our conclusions are still valid in emergent IPv6 networks. Further, with the rise of mobile devices and mobile connections, it would be interesting to push this study further by looking at middleboxes in mobile networks using TraceboxAndroid [20].

Acknowledgments

Thanks to Roman Müntener and Stephan Neuhaus from Zurich University of Applied Sciences (ZHAW) for fast assistance in using the MAMI Path Transparency Observatory.

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 688421. The opinions expressed and arguments employed reflect only the authors' views. The European Commission is not responsible for any use that may be made of that information.

References

[1] G. Papastergiou, G. Fairhurst, D. Ros, A. Brunstrom, K.-J. Grinnemo, P. Hurtig, N. Khademi, M. Tuxen, M. Welzl, D. Damjanovic, *et al.*, "De-ossifying the internet transport layer: A survey and future perspectives," *IEEE Communications Surveys & Tutorials*, 2016.

[2] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, "Making middleboxes someone else's problem: Network processing as a cloud service," in *Proc. ACM SIGCOMM*, August 2012.

[3] Z. Wang, Z. Qian, Q. Xu, Z. Mao, and M. Zhang, "An untold story of middleboxes in cellular networks," in *Proc. ACM SIGCOMM*, August 2011.

[4] L. D'Acunto, N. Chiluka, T. Vinò, and H. J. Sips, "Bittorrent-like P2P approaches for VoD: a comparative study," *Computer Networks (COMNET)*, vol. 57, no. 5, pp. 1253–1276, April 2013.

[5] M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda, "Is it still possible to extend TCP," in *Proc. ACM/USENIX Internet Measurement Conference (IMC)*, November 2011.

[6] B. Hesmans, F. Duchene, C. Paasch, G. Detal, and O. Bonaventure, "Are TCP extensions middlebox-proof?" in *Proc. Workshop on Hot Topics in Middleboxes and Network Function Virtualization (HotMiddlebox)*, December 2013.

[7] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "TCP extensions for multipath operation with multiple addresses," Internet Engineering Task Force, RFC 6824, January 2014.

[8] A. Medina, M. Allman, and S. Floyd, "Measuring interactions between transport protocols and middleboxes," in *Proc. ACM Internet Measurement Conference (IMC)*, November 2004.

[9] G. Detal, b. Hesmans, O. Bonaventure, Y. Vanaubel, and B. Donnet, "Revealing middlebox interference with tracebox," in *Proc. ACM Internet Measurement Conference (IMC)*, October 2013.

[10] R. Craven, R. Beverly, and M. Allman, "Middlebox-cooperative TCP for a non end-to-end Internet," in *Proc. ACM SIGCOMM*, August 2014.

[11] V. Paxson, "End-to-end routing behavior in the Internet," *IEEE/ACM Transactions on Networking*, vol. 5, no. 5, pp. 601–615, October 1997.

[12] D. G. Andersen, N. Feamster, S. Bauer, and H. Balakrishnan, "Topology inference from BGP routing dynamics," in *Proc. ACM SIGCOMM Internet Measurement Workshop (IMW)*, Nov. 2002.

[13] I. Cunha, R. Teixeira, and C. Diot, "Measuring and characterizing end-to-end route in the presence of load balancing," in *Proc. Passive and Active Measurement Conference (PAM)*, March 2011.

[14] V. Jacobson et al., "traceroute," UNIX," man page, 1989.

[15] F. Baker, "Requirements for IP version," Internet Engineering Task Force, RFC 1812, June 1995.

[16] J. Postel, "Internet control message protocol," Internet Engineering Task Force, RFC 792, September 1981.

[17] Team Cymru, "IP to ASN mapping," 2008, http://www.team-cymru.org/Services/ip-to-asn.html.

[18] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. F. Wu, and L. Zhang, "An analysis of BGP multiple origin AS (MOAS) conflicts," in *Proc. ACM SIGCOMM Internet Measurement Workshop (IMW)*, October 2001.

[19] Center for Applied Data Analysis, "The CAIDA UCSD internet topology data kit," March 2016, see http://www.caida.org/data/internet-topology-data-kit.

[20] V. Thirion, K. Edeline, and B. Donnet, "Tracking middleboxes in the mobile world with traceboxandroid," in *Proc. 7th International Workshop on Traffic Monitoring and Analysis (TMA)*, April 2015.