On a Middlebox Classification

Korian Edeline
Université de Liège
Belgium
korian.edeline@ulg.ac.be

Benoit Donnet
Université de Liège
Belgium
benoit.donnet@ulg.ac.be

ABSTRACT

Recent years have seen the rise of middleboxes, such as firewalls, NATs, proxies, or Deep Packet Inspectors. Those middleboxes play an important role in today's Internet, including enterprise networks and cellular networks. However, despite their huge success in modern network architecture, their actual impact on packets, traffic, or network performance (all in IPv4 and IPv6 networks) is not that much understood.

In this paper, we propose a path impairment oriented middlebox classification that aims at categorizing the initial purpose of a middlebox policy as well as its potential unexpected complications.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Network Topology

General Terms

Classification

Keywords

Middleboxes, Taxonomy, Tracebox, Traceroute, Path Impairment

1. 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* (i.e., "an intermediary box performing functions apart from normal, standard functions of an IP router on the data path between a source host and destination host" [1]) being part of the network architecture in addition to traditional network hardware [2].

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 where behind NATs. Further, it has been shown that middleboxes have a negative impact on the TCP protocol (and its extensions) evolution [5, 6].

There is a wide range of middleboxes, going from "simple" NAT to complex system that can potentially modify the whole packet header. There is thus a wide variety of middleboxes, as the wildlife and plant life are diversified. However, on the contrary to the biology, there is no rigorous behavioral taxonomy of the middleboxes, classifying them according to their effects on packets, on traffic, or on the network quality experienced by users. Furthermore, as middleboxes increasingly impact the Internet, protocol designers have to cope with a middlebox-full Internet. Each 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.

This is exactly what we want to tackle here. In this paper, we advocate for and propose a path-impairment oriented middlebox policy taxonomy, that aims at categorizing the initial purpose of a middlebox policy as well as its potential unexpected complications. We choose to classify middlebox *policies* (i.e., "Definite methods of action to guide and determine present and future decisions" [8]) rather than middleboxes themselves because the latter often combine multiple policies. Our taxonomy is based on three main policies: *Action* (i.e., the fate of a packet crossing a middlebox implementing this policy), *Function* (i.e., the policy purpose), and *Complication* (i.e., the potential path connectivity deterioration).

The remainder of this paper is organized as follows: Sec. 2 presents our middleboxes taxonomy; finally, Sec. 3 concludes this paper by summarizing its main achievements.

2. MIDDLEBOXES CLASSIFICATION

In this paper, we advocate for and propose a path-impairment oriented middlebox policy taxonomy, that aims at categorizing the initial purpose of a middlebox policy as well as its potential unexpected complications. We choose to classify middlebox policies (i.e., "A definite method of action to guide and determine present and future decisions" [8]) rather than middleboxes themselves because the latter often combine multiple policies. Our taxonomy focuses on packet mangling middleboxes aspects, the initial purpose of such an action, and the network interferences that may involuntarily result.

We describe each policy implemented in a single (not multi-hop) middlebox in three ways, each including several taxa (i.e., categories); (*i*) *Action*, the fate of a packet crossing a middlebox that implements this policy; (*ii*) *Function*, what the policy is intending to achieve, its purpose; (*iii*) *Complication*, the possible resulting path connectivity de-

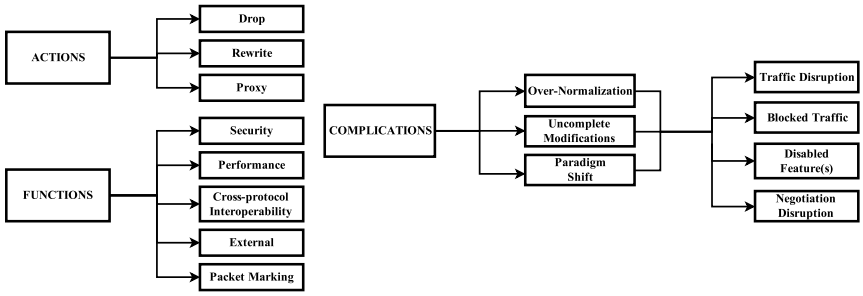

Figure 1: A path-impairment oriented middlebox policy classification

terioration. Fig. 1 illustrates our path-impairment oriented middleboxes policy taxonomy. Each middlebox policy has to fall in *at least* one taxon for each of the three angles in order to be consistent with the taxonomy. Furthermore, each category essentially describes path characteristics, deliberately avoiding a larger focus.

A middlebox policy can be classified differently if we consider the policy in itself or the functional entity it is part of. For instance, the purpose of a tunnel endpoint policy is to provide cross-protocol interoperability but, if we consider the whole tunnel, it may be security (e.g, VPNs), traffic engineering (e.g., MPLS), or another. As we aim at characterizing middlebox-related network interferences rather than establishing an exhaustive middlebox taxonomy [1], we focus on single-hop modifications and ignore composition aspects. Moreover, we intentionally leave aside the middlebox state-related features (e.g., statefull, stateless), robustness to failure, and its implicit or explicit nature.

While examining the possible complications involved by middlebox policies, we deliberately narrow our horizon to performance worsening and feature's inability of use, omitting the multiple and various reported security flaws created by middlebox policy implementations [9, 10, 11].

Fig. 1 displays the taxonomy and shows its three facets: *Actions*, *Functions*, and *Complications*. These families are described in the subsequent sections.

2.1 Actions

The *Action* category describes the actual action of a middlebox on a matched packet, defined by middlebox policies. We consider three basic kinds of middleboxes policy action: *Drop*, *Rewrite*, and *Proxy*. This aspect is decisive because middlebox policies applying different actions will more likely cause different types of network dysfunctions.

Drop policies are common features whose purposes vary from security to performance optimization concerns. Depending on how both ends react to this type of failure, the outcome may also vary from minor traffic disruptions, such as bandwidth reduction or the inability to use specific TCP options, to the inability to establish a TCP connection. An example of an ordinary middlebox policy that may apply dropping policies are TCP sequence number sanity checks for dropping out-of-window packets and invalid ACKs [5, 6].

Rewrite policies are also common among middleboxes. Their initial purposes are covering the entire Sec. 2.2. As they break the TCP/IP end-to-end principles [12], they may cause various problems to protocol end-to-end functions which assume unmodified layers over the third. An aging but still relevant example is the TCP Initial Sequence Number (ISN) re-randomization policy which aimed at mitigating ISN prediction attacks [13], but which may fail to ensure consistencies with TCP absolute sequence related numbers other than the TCP sequence number and the TCP acknowledgment number (e.g., Selective Acknowledgement) and cause traffic disruption [6, 14].

Proxy policies middleboxes are relay agents between clients and servers of an application. They vary from the rewritten policies middleboxes by not simply rewriting the forwarded packets but rather by receiving client data from a connection, then establishing a second connection to send data to the server and vice versa.

2.2 Functions

The *Function* category describes the purpose of a middlebox policy (e.g., what it is intending to achieve). As we already noticed earlier in this section, a policy can be classified differently if we consider the policy in itself or the middlebox set it is part of (e.g., a tunnel endpoint with respect to the whole tunnel). We consider five kinds of Functions middleboxes: *Security*, *Performance*, *Cross-Protocol Interoperability*, *External*, and *Packet Marking*.

Security-motivated middlebox policies are implemented in Security-oriented middleboxes, dedicated hardware deployed in enterprise, and home network for improving network security. They may serve purposes like providing an authentication mechanism (e.g., ALGs), defending against attacks such as DDoS, attacks on IP (e.g., IP spoofing, fragmentation attacks) or attacks on TCP (e.g., sequence number related attacks, TCP reset attacks), providing access control, normalizing the TCP features to prevent the use of features considered unknown and/or unsafe or separating similar networks into different security zones.

It has been shown that such a type of middleboxes is becoming more and more popular [2]. This increasing popularity raises concerns regarding overly restrictive middlebox policies which may either block benign traffic matched as unsafe, causing blackholes, or apply modifications to transport and network header fields, precluding the use of protocol features or hampering their proper functioning, indirectly reducing networks performance and/or functionalities (e.g., forbidding ECN or MultiPathTCP). Security-oriented middleboxes involves IP and application firewalls, IDS/IPSes, certain (ALGs), and NATs.

Performance-enhancing middlebox policies aim at improv-

ing the network performance regarding a link along the path, a connection between both path's ends, or the middlebox itself It may aim at improving pure effective bandwidth performance through transport layer engineering as well as solving box-relative performance issues.

This category of middlebox policies may apply semantically wrong legacy modifications which may cause traffic disruption and protocol inconsistencies. They might as well introduce TCP errors and unforeseen changes in TCP adaptive behavior, leading to various network interferences. TCP performance enhancing proxies [15], caches, and packet schedulers are examples of middleboxes implementing *Performance* policies.

Cross-protocol interoperability middlebox policies are performing protocol translation. They aim either at connecting dissimilar networks or at translating protocols over the network layer. The main problem with this kind of policies is to be consistent with every protocol versions and features. Indeed, the input protocol language may evolve, and middleboxes may then apply outdated translations to it or may fail to ensure completeness of its modifications. This can lead either to the inability to use certain features or to network performance degradation. Example of cross-protocol interoperability middleboxes policies are NAT 6to4, Interworking between LISP enabled sites and non-LISP sites [16], NAT-Protocol Translation (NAT-PT [17]), Stateless IP/ICMP Translation (SIIT [18]), tunnels endpoints and transcoders.

Packet marking middleboxes classify packets according to policies and select or mark them for differentiated services (via IP DSCP's field). Their action is limited to network layer modifications. They are packet classifiers, or ECN-capable routers. Their policies are not likely to degrade network performance.

External middlebox policies have purposes that are external to the paths to which they belong (e.g., IPv4 address exhaustion) or not related to the Internet at all (e.g., economic purposes). External middlebox policies are heterogeneous and have to be analyzed independently.

2.3 Complications

We describe here the potential *Complications* caused by middleboxes policies by examining them under two points of view: (*i*), their technical causes, which are directly related to their initial purposes and, (*ii*), the associated actions (respectively Sec. 2.2 and Sec. 2.1), and their unfortunate consequences, i.e., causes and consequences.

2.3.1 Causes

The *Causes* of the network interferences created by middleboxes aims at classifying the technical problems origin. It regroups manufacturers/policy designers fundamental errors or deliberated choices that, from a path-impairment perspective, lead to network interferences.

Over-normalization refers to a middlebox policy that limits protocol features and options, as a blacklist or whitelist filter, to a restricted subset of the protocol. The problem of this type of middlebox behavior constraining the design of new extensions has already been addressed [5]. It may limit protocol performance as well by preventing the usage of the entire protocols capabilities, or simply by taking drop decisions. The Cisco ASA firewall family includes a TCP normalizer feature [19] that limits TCP to a subset of the protocol considered safer either by removing and rewriting

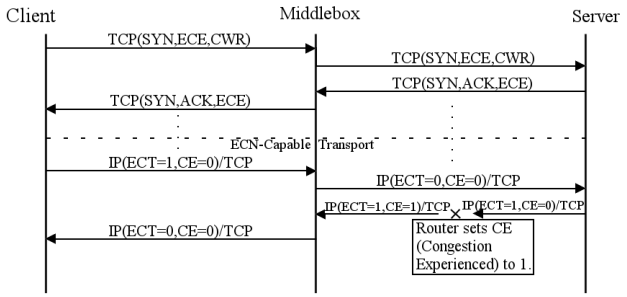

Figure 2: Clearing IP ECN bits: Over-Normalization

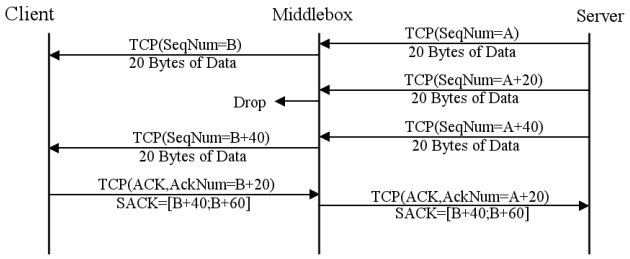

Figure 3: TCP Initial Sequence Number re-shuffling middlebox and Selective ACKnowledgement: Incomplete modifications

bytes related to unwanted features or by dropping packets containing those features.

An example of such a middlebox behavior is displayed in Fig. 2. In this example, the middlebox is allowing both ends to negotiate the use of ECN but is clearing the ECN bits in the IP header, rendering it unable to report any congestion. The client tries to establish an ECN-Capable connection with a remote server by sending a TCP `SYN` segment with the `ECE` (ECN Echo) and `CWR` (Congestion Window Reduced) flags set. The server sends back a TCP `SYN+ACK` segment with the `ECE` flag set. Both packets are forwarded unmodified by the middlebox. Following packets are marked as ECN-Capable Transport (ECT) by both ends but the middlebox is systematically clearing both the `ECT` and `CE` (Congestion Experienced) bits in the IP header. If an intermediate router sets the `CE` bit of a packet, it will be cleared by the middlebox afterward.

Incomplete modifications refers to middlebox policies that fail to ensure completeness of their modification(s). This type of network inconveniences is caused by middleboxes modifying a specific protocol field and not modifying semantically related fields, allowing translated/modified data alongside untranslated/unmodified data. They may fail to identify all related fields for legacy reasons or simply neglecting them for performance concerns (e.g., refusing to parse TCP options). Examples of such middlebox policies are TCP ISN shuffling boxes and NATs regarding respectively TCP Selective ACKnowledgement (`SACK`) and FTP.

Fig. 3 displays an example of a middlebox applying *Incomplete modifications*. Server is sending three TCP segments

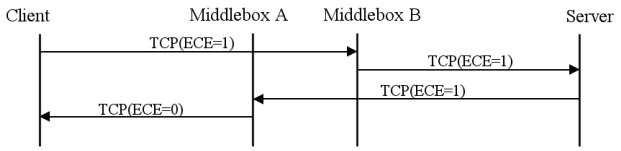

Figure 4: State announcement and asymmetric path/load balancing: Paradigm shift

with 20 bytes of data each through an already established TCP connection within which both ends have agreed to use the `SACK` option. In the path between the client and the server, there is a middlebox rewriting the Sequence Number and ACKnowledgement Number of any packet of this connection as it has re-shuffled the Initial Sequence Number to counter prediction attacks. When the second packet with Sequence Number $A + 20$ is dropped, the receiving side is notifying the sending side by acknowledging the first and the third data segment using the `ACK` number and the `SACK` option. The middlebox is modifying the sequence and `ACK` number of this packet, but not the sequence numbers of the `SACK` block. If those unmodified sequence numbers are out of window, Linux's TCP stack discards the whole packet [6].

A *Paradigm shift* (2-way to `n`-way) happens when both ends running a protocol are assuming 2-way peering relationships. Middleboxes, by applying modifications *in the middle of the path*, are breaking the TCP/IP end-to-end principles, and thus, are causing both ends to undergo a paradigm shift *de facto* to `n`-way peering relationships [12]. As many mechanisms are not designed to handle this new paradigm, errors may occur. When both ends are trying to share state related data or to negotiate capabilities, this phenomenon may, in certain scenarios, put both ends in inconsistent states or, combined with an unfortunate load balancing, distort protocol negotiations [6]. This paradigm shift also raises security concerns. We know that NATs are limiting the use of IPSEC [20], but it also invalidates most transport layer security solutions. Moreover, the trust models and key distribution models have to be definitively rethought as `n`-way relationships [1].

A consequence of the *Paradigm shift* implied by middleboxes is shown in Fig. 4. In this example, both ends are trying to share state related data (i.e., `ECE` bit). The path between them is asymmetrical and ingress and egress traffic are crossing different middleboxes, A and B respectively. Middlebox A is clearing the `ECE` bit and middlebox B is not. The result of the state share is inconsistent because the server has sent `ECE` $= 1$ and the client has received `ECE` $= 0$, the client thinks that the connection is not ECN-Capable while the server does. This shows that a consequence of the *Paradigm shift* is that end state announcements are becoming path state announcements, and if the path is asymmetrical, it may lead to inconsistencies.

2.3.2 Consequences

The *Consequences* are the network complications final outcome, i.e., what both ends actually experience. We focus exclusively on path performance related issues, leaving aside security and processing performance considerations.

Traffic disruption policies produce unwanted consequences such as interferences with control data rendering it useless, bandwidth reductions, or others path performance impairments.

Middlebox policies may cause *Blocked traffic* either explicitly (sending TCP `RST` or ICMP destination unreachable packets) or implicitly (dropping packets). It may not be the final outcome of a connection. If a specific option/feature is blocked by a middlebox, the client could be configured to retry establishing the connection without the undesirable options/features, but if it is not, no connection at all is possible.

Middlebox policies may aim at preventing the use of features considered unknown and/or unsafe by modifying, nopping them, or by preventing them from being negotiated. If it is achieved symmetrically, the consequences are limited to the inability to use the restricted features, i.e., it is a *Feature-disabling* policy. If the modifications are done asymmetrically and the negotiation is not resilient enough, the policy may fail to disable the feature and lead to inconsistent protocol states [6]. Policies resulting in the latter consequences are categorized as *Negotiation disruption* policies.

3. CONCLUSION

Middleboxes are becoming more and more popular in mostly every type of network. 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. Aside from this, there is no formal classification of middleboxes according to their effects on packets, traffic, or on the network quality experienced by users.

This is exactly what we have addressed in this paper. We advocated for and proposed a path-impairment oriented middlebox policy taxonomy, that aims at categorizing the initial purpose of a middlebox policy as well as its potential unexpected complications.

It is clear that the classification proposed in this paper is incomplete. In the near future, we plan to include some security aspects to our taxonomy. Some of them were briefly discussed in this paper, but we wish to extend our middlebox policy taxonomy to include security concerns and middlebox-related complications.

Acknowledgments

This work is funded by the European Commission funded mPlane ICT-318627 project.

4. REFERENCES

[1] B. Carpenter and S. Brim, "Middleboxes: Taxonomy and issues," Internet Engineering Task Force, RFC 3234, February 2002.

[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*, 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] D. Agrawal, S. Calo, K. Lee, and D. Lobo, J Verma, *Policy Definition and Usage Scenarios for Self-Managing Systems*. IBM Press, 2008.

[9] Z. Qian and Z. M. Mao, "Off-path TCP sequence number inference attack - how firewall middleboxes reduce security," in *Proc. IEEE Symposium on Security and Privacy (SP)*, May 2012.

[10] S. Kamara, S. Fahmy, E. Schultz, F. Kerschbaum, and M. Frantzen, "Analysis of vulnerabilities in Internet firewalls," *Computers & Security*, vol. 22, no. 3, pp. 214–232, April 2003.

[11] Z. Qian, Z. M. Mao, and Y. Xie, "Collaborative TCP sequence number inference attack: How to crack sequence number under a second," in *Proc. ACM Conference on Computer and Communications Security (CCS)*, October 2012.

[12] M. A. Lemley and L. Lessig, "The end of end-to-end: Preserving the architecture of the Internet in the broadband era," University of California at Los Angeles, Technical Report 2000-19, October 2000.

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

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

[15] J. Border, J. Griner, G. Montenegro, Z. Shelby, and M. Kojo, "Performance enhancing proxies intended to mitigate link-related degradations," Internet Engineering Task Force, RFC 3135, June 2001.

[16] D. Lewis, D. Meyer, D. Farinacci, and V. Fuller, "Interworking between locator/id separation protocol (LISP) and non-LISP sites," Internet Engineering Task Force, RFC 6832, January 2013.

[17] G. Tsirtsis and P. Srisuresh, "Network address translation-protocol translation," Internet Engineering Task Force, RFC 2766, February 2000.

[18] E. Normark, "Stateless IP/ICMP translation algorithm (SIIT)," Internet Engineering Task Force, RFC 2765, February 2000.

[19] J. Frahom, O. Santos, and A. Ossipov, *Cisco ASA: All-in-One Firewall, IPS, and VPN Adaptive Security Appliance*. Pearson Education, 2014.

[20] B. Aboba and W. Dixon, "IPsec-network address translation (NAT) compatibility requirements," Internet Engineering Task Force, RFC 3715, March 2004.