Detecting Middlebox Interference on Applications

Shan Huang
Queen Mary University of London, UK
shan.huang@qmul.ac.uk

Giuseppe Aceto
University of Napoli 'Federico II', IT
giuseppe.aceto@unina.it

Félix Cuadrado
Queen Mary University of London, UK
felix.cuadrado@qmul.ac.uk

Steve Uhlig
Queen Mary University of London, UK
steve.uhlig@qmul.ac.uk

Antonio Pescapè
University of Napoli 'Federico II', IT
pescape@unina.it

ABSTRACT

Middleboxes are widely used in today's Internet, especially for security and performance. Recent studies have uncovered and studied middleboxes in different types of networks. Middleboxes classify, filter and shape traffic, therefore interfering with application performance.

In this paper, we propose a tool to detect and locate middlebox interference for specific applications, specifically DNS and HTTP. The tool sends probes with crafted IP TTL and application payloads from multiple vantage points, that provides the ability to identify any middlebox interference in both directions of the traffic flows. In addition, we capture and analyse all transmitted packets to figure out the hop range of middlebox location.

Categories and Subject Descriptors

C2.3 [**Network Operations**]: Network Monitoring

Keywords

Middleboxes, Internet censorship, Internet measurement

1. INTRODUCTION

Middleboxes such as firewalls, load balancing switches and Deep Packet Inspection boxes have been a major part in today's network infrastructure. Currently, the main goals driving the deployment of middleboxes are security (to enhance the visibility of network traffic and enable the enforcement of security policies), and performance enhancement (through traffic shaping, caching and transparent proxying). Middleboxes are widely used in various types of networks. Indeed, according to a survey of 57 enterprise network administrators, it was concluded that there are probably as many middleboxes as routers inside the network [1]. Since middleboxes have the ability to inspect and manipulate traffic flows, they may interfere with their end-to-end behaviour, and even affect the performance of end host applications [2]. However, application developers have limited knowledge about middlebox interference [3].

In comparison with other Internet devices, such as switches and routers, middleboxes are complex, as they operate on packets from network layer to application layer at line rate. In general, the middlebox interference can be categorized into three types. First, middleboxes intentionally drop or filter the packets as policies [3, 4]. For example, network administrators begin to filter P2P file sharing traffic to avoid the legal implications of copyrighted files [5]. Second, middleboxes modify the packet contents [2, 3, 6]. Finally, in order to interfere with the performance of some applications, middleboxes inject forged packets, e.g., for blocking purposes. For instance, the Great Firewall of China (GFC) blocks some certain sites by injecting spoofed DNS responses, and causes collateral damage in terms of Internet censorship [7]. Considering the complexity of middleboxes and applications, we argue that tools are needed to detect and analyse middlebox interference for different kinds of applications.

One example of use of middleboxes nowadays is for censorship. GFC is the most complex middlebox system that operates with numerous protocols ([8, 9, 10, 11, 12]). Vietnam uses a system of firewalls, access controls, and strenuously encouraged self-censorship ([13, 14]) to implement network censorship. The governments of Iran, Yemen, Tunisia and Sudan use commercial software in state-controlled servers, and censor all outgoing flows [13]. Existing studies on Internet censorship provided some basic methods to detect middlebox

CoNEXT Student Workshop'15 December 01-01 2015, Heidelberg, Germany
© 2015 ACM. ISBN 978-1-4503-4066-3/15/12…$15.00
DOI: http://dx.doi.org/10.1145/2842665.2843566

interference. However, Internet traffic is changing (e.g., HTTPS accounts for a significant portion of Internet traffic now [15]), and the corresponding censoring techniques are evolving. Therefore, we need a tool capable of studying these new ways middleboxes might interfere with the changing Internet.

Prior work has detected middleboxes by crafting TCP segments [2, 3, 6], and analysed middlebox interference by correlating packet modifications. Tracebox [6] sends test probes with increasing TTL and makes use of the returned ICMP Time-Exceed (TE) packets to locate middleboxes. However, most of the adopted probing packets are crafted with TCP/IP headers without any application layer contents, or they focus only on packet modification. On the other hand, Internet censorship measurement uses increasing TTL probes to detect and locate the middleboxes ([7, 9]), but the test probes are specific for one application, and the considered measurement paths are only one direction.

2. METHODOLOGY

Compared with other related work, our goal is to develop a tool that can detect and locate any middlebox interference (filtering, injection, modification) at the application layer. Our tool adopts a client-server architecture, allowing to generate probe packets from clients and servers and compare the expected traffic exchanged between client and server and the one actually received. Our probe packets are crafted with application layer payloads that attempt to expose middlebox interference. Relying on a Tracebox-like probing with increasing TTL values helps locate the approximate position of the interfering middleboxes.

Analysing the captured traces, the server should receive the original probe (whose TTL value is equal or more than the number of hops to the server). If not, the probe may have been lost or filtered by some middlebox. To take into account the effect of random packet loss, we send the probes several times, repeating the experiments during a full day. Also, we send probes that do not attempt to trigger middlebox interference to verify that the exposed middlebox behaviour is indeed caused by our specific payload.

Another possible interference is packet injection triggered by probes from a given TTL value. Comparing the traces on the client and server side, we can find the injected packets which are sent from the middleboxes. The destination port number of the injected packet allows our tool to match the given TTL of the original probe and find out the hop range of the middlebox position.

As mentioned in the Tracebox methodology, ICMP TE replies contain the offending packet. According to RFC792 [16], the returned ICMP TE reply should quote the IP header and the next eight bytes of the original packet. RFC1812 [17] suggested to quote the entire IP packet in the ICMP TE reply, but this recommendation has not been widely implemented. Our tool detects and locates the middlebox modification by comparing the original probe, the quoted packet in the returned ICMP TE reply and the received one in server side.

3. PRELIMINARY RESULTS

The application we have considered for detection of middlebox interference is DNS, as it is known as one of the services affected by middleboxes such as the GFC.

The measurement setup consisted in a client inside China and a server outside China. We conducted the experiments at the same time with two client-server pairs. In the measurements, we sequentially used a non-blacklisted domain, and two blacklisted ones. For each probe, the injection interference appears independent across different domain names. We did not detect evidence that would suggest that the middleboxes are stateful.

The tool detected interference of the *injection* type, as reported in [7]. We confirm the results from [18], our tool detected that the GFC injected two different types of fake responses to pollute DNS resolvers inside China.

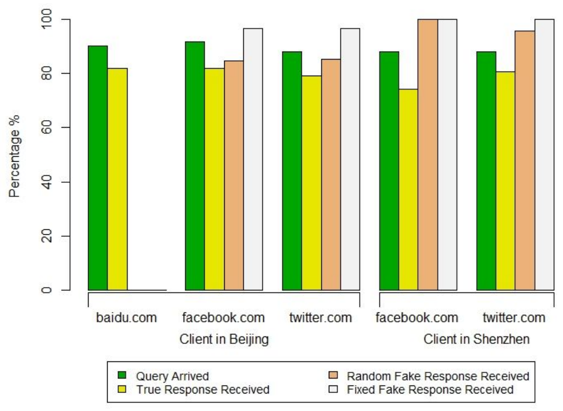

Figure 1: GFC DNS censorship with injected responses: the Fixed Fake Response had fixed IP ID and TTL in IP header, instead of random values as found in Random Fake Response.

Figure 1 shows how the non-blacklisted (baidu.com) query and the related responses are affected by a similar random packet loss compared to censored ones. By comparing censored and non-censored queries, we infer that no filtering is performed related to the domain in the request.

4. FUTURE WORK

In terms of future work, we plan to apply our methodology to other applications, such as Transport Layer Security Protocol (TLS), as well as to investigate the extent of censorship based on middleboxes across the Internet.

5. REFERENCES

[1] Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, and Vyas Sekar. Making middleboxes someone else's problem: network processing as a cloud service. In *Proc. ACM SIGCOMM*, 2012.

[2] Michio Honda, Yoshifumi Nishida, Costin Raiciu, Adam Greenhalgh, Mark Handley, and Hideyuki Tokuda. Is it still possible to extend tcp? In *Proc. ACM IMC*, 2011.

[3] Zhaoguang Wang, Zhiyun Qian, Qiang Xu, Zhuoqing Morley Mao, and Ming Zhang. An untold story of middleboxes in cellular networks. In *Proc. ACM SIGCOMM*, 2011.

[4] Giuseppe Aceto and Antonio Pescapè. Internet censorship detection: A survey. *Computer Networks*, 83:381–421, 2015.

[5] Marcel Dischinger, Alan Mislove, Andreas Haeberlen, and P. Krishna Gummadi. Detecting bittorrent blocking. In *Proc.ACM IMC*, 2008.

[6] Gregory Detal, Benjamin Hesmans, Olivier Bonaventure, Yves Vanaubel, and Benoit Donnet. Revealing middlebox interference with tracebox. In *Proc.ACM IMC*, 2013.

[7] The collateral damage of internet censorship by DNS injection. *SIGCOMM CCR*, 42(3), 2012.

[8] Philipp Winter and Stefan Lindskog. How the great firewall of china is blocking tor. In *Proc. USENIX FOCI*, 2012.

[9] Xueyang Xu, Zhuoqing Morley Mao, and J. Alex Halderman. Internet censorship in china: Where does the filtering occur? In *Proc.PAM*, 2011.

[10] Jedidiah R. Crandall, Daniel Zinn, Michael Byrd, Earl T. Barr, and Rich East. Conceptdoppler: a weather tracker for internet censorship. In *Proc.ACM CCS*, 2007.

[11] Xia Chu. Complete GFW rulebook for Wikipedia plus comprehensivve list for websites, IPs, IMDB and AppStore. https://docs.google.com/file/d/0B8ztBERe_FUwLWxUX0laeWF3aE0/edit, 2013. [Online; accessed 25-December-2013].

[12] Roya Ensafi, David Fifield, Philipp Winter, Nick Feamster, Nicholas Weaver, and Vern Paxson. Examining how the Great Firewall discovers hidden circumvention servers. In *Proc.ACM IMC*, 2015.

[13] Barney Warf. Geographies of global internet censorship. *GeoJournal*, 76:1–23, 2011.

[14] Phillipa Gill, Masashi Crete-Nishihata, Jakub Dalek, Sharon Goldberg, Adam Senft, and Greg Wiseman. Characterizing web censorship worldwide: Another look at the opennet initiative data. *ACM TWEB*, 9(1):4:1–4:29, 2015.

[15] David Naylor, Kyle Schomp, Matteo Varvello, Ilias Leontiadis, Jeremy Blackburn, Diego R. López, Konstantina Papagiannaki, Pablo Rodríguez Rodríguez, and Peter Steenkiste. Multi-context TLS (mctls): Enabling secure in-network functionality in TLS. In *Proc. ACM SIGCOMM*, 2015.

[16] J.Postel. Internet control message protocol. In *RFC792*, September. 1981.

[17] F.Baker. Requirements for IP version 4 routers. In *RFC1812*, June. 1995.

[18] Towards a comprehensive picture of the great firewall's DNS censorship. In *Proc. USENIX FOCI*, 2014.