PMTUD is not Panacea: Revisiting IP Fragmentation Attacks against TCP

Xuewei Feng*, Qi Li†‡, Kun Sun§, Ke Xu*‡✉, Baojun Liu†, Xiaofeng Zheng†¶, Qiushi Yang¶
Haixin Duan†¶, Zhiyun Qian‖
*Department of Computer Science and Technology, Tsinghua University, China
†Institute for Network Sciences and Cyberspace, Tsinghua University, China
‡Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, China
§Department of Information Sciences and Technology, CSIS, George Mason University, USA
¶Qi An Xin Group, China, ‖UC Riverside, USA
{fengxw18@mails, qli01@, xuke@, lbj@, zxf19@mails, duanhx@}tsinghua.edu.cn, ksun3@gmu.edu
yangqiushi@qianxin.com, zhiyunq@cs.ucr.edu

***Abstract*—There is a widespread belief that TCP is not vulnerable to IP fragmentation attacks since TCP performs the Path Maximum Transmission Unit Discovery (PMTUD) mechanism by default, which can avoid IP fragmentation by dynamically matching the maximum size of TCP segments with the maximum transmission unit (MTU) of the path from the originator to the destination. However, this paper reveals that TCP is in fact vulnerable to IP fragmentation attacks, which is contrary to the common belief.**

We conduct a systematic study on the complex interactions between IP fragmentation and TCP, and we discover two key situations under which IP fragmentation can still be triggered on TCP segments even if the originator performs PMTUD. First, when the next-hop MTU of an intermediate router is smaller than the originator's acceptable minimum path MTU, TCP segments from the originator will be fragmented by the router. Second, when the originator's path MTU values between the IP layer and the TCP layer are desynchronized due to a maliciously crafted ICMP error message, the originator could be tricked into fragmenting TCP segments. Once IP fragmentation on TCP segments could be falsely triggered, attackers can inject forged fragments into the victim connection to poison the target TCP traffic after successfully addressing practical issues of predicting IPID and deceiving TCP checksum. Our case studies on both HTTP and BGP demonstrate the feasibility and effectiveness of poisoning TCP-based applications via IP fragmentation. We also conduct a comprehensive evaluation to show that our attacks can cause serious damages in the real world. Finally, we propose countermeasures to mitigate malicious IP fragmentation on TCP segments and defeat the attacks.

I. Introduction

As an Internet Protocol (IP) mechanism, IP fragmentation splits large packets into multiple smaller fragments for them to pass through a transmission link that enforces a maximum packet length limit called Maximum Transmission Unit (MTU) [70]. When the fragments arrive at the destination, they are reassembled into the original packets. Historically, IP fragmentation has been widely abused by attackers to evade intrusion detection system (IDS) [67], [5] and launch denial of service (DoS) attacks [54], [61], [29], [31]. Moreover, attackers can misuse IP fragmentation to attack UDP based applications and services, e.g., poisoning DNS cache [41], [11], [42], [86]. Since UDP is a transaction-oriented datagram protocol [68] and is not tightly coupled with IP, it lacks the ability to dynamically adjust the datagram size and thus avoid IP fragmentation. Therefore, when the MTU of the host or the transmission path is smaller than the datagram size given by the application, IP fragmentation will happen on UDP based applications[1].

In contrast, TCP is an unstructured stream protocol tightly coupled with IP, so that TCP can proactively avoid IP fragmentation by adjusting the maximum segment size (MSS), a construct unique to TCP, according to the path MTU value maintained in the IP layer. As a result, it has been widely believed that TCP protocol is not vulnerable to IP fragmentation attacks [41], [31], [64], [82], [46], [17], [30]. For instance, one study [84] explicitly states that the path MTU discovery (PMTUD) mechanism [57], [59] is an effective defense mechanism to prevent IP fragmentation attacks against TCP. Another reason to believe that TCP is invulnerable is due to the difficulty on disabling the TCP checksum mechanism during the attacks, though UDP's checksum mechanism can be easily disabled by zeroing the field [31], [68].

In this paper, we systematically study the complex interactions between IP fragmentation and TCP. Through extensive real-world evaluations, we demonstrate that TCP is still vulnerable to IP fragmentation attacks, which is contrary to the common belief. Particularly, we show that even off-path attackers can still trigger IP fragmentation on TCP and then poison the original TCP segments by injecting malicious IP fragments into the victim connection, which does not require guessing the sequence and acknowledgment numbers that are always carried in the first benign fragment.

Our study identifies two situations under which IP fragmentation on TCP segments can be triggered by off-path attackers,

Network and Distributed Systems Security (NDSS) Symposium 2022
27 February - 3 March 2022, San Diego, CA, USA
ISBN 1-891562-74-6
https://dx.doi.org/10.14722/ndss.2022.24381
www.ndss-symposium.org

[1]RFC8899, which is released in September 2020, supports UDP based applications with path MTU discovery; however, our measurement study shows that rare applications implement those complicated functions yet.

even if PMTUD has been enabled by the originator. First, along the path from an originator to a destination, when the next-hop MTU of an intermediate router is smaller than the originator's acceptable minimum path MTU, TCP segments from the originator will be fragmented by this intermediate router, though the originator may assemble the packets with the size of its acceptable minimum path MTU permanently [76]. By analyzing datasets collected from routers in backbone networks and conducting active measurements on the Internet, we identify a considerable number of problematic routers (i.e., with a small next-hop MTU size) in the real world that fragment TCP segments. For example, by analyzing datasets collected from a 10Gbps backbone IPv4 router on the Internet, we find more than 600 problematic routers on the Internet whose next-hop MTU values are less than 596 octets, the acceptable minimum path MTUs of Windows systems[2]. Thus, these routers will fragment TCP segments generated by the originators running Windows systems.

Second, we discover that an off-path attacker can carefully craft an ICMP error message with `type`=3 and `code`=4 (i.e., an ICMP "Fragmentation Needed" message) to trigger desynchronization on the path MTU value between the IP and TCP layers, thus incurring IP fragmentation on TCP. The ICMP "Fragmentation Needed" message is used to indicate the originator that the embedded packet in the message is oversized and needs to be fragmented [59]. When an attacker pretends to be a router and sends such a crafted message embedded with an ICMP echo reply packet to the originator, the originator defers the updated path MTU value from the originator's IP layer to TCP layer, so that TCP will write oversized segments to IP using the old path MTU value. As a result, it causes IP to fragment these oversized segments. This vulnerability is found in Linux kernel version 3.8.1 and beyond. We evaluate the prevalence of the false IP fragmentation on TCP due to the crafted ICMP error message in the real world, and we discover that more than 14% of the Alexa (www.alexa.com) top 10k websites are vulnerable to fragmenting TCP segments even if PMTUD is performed, and thus they may suffer from off-path poisoning attacks.

Once IP fragmentation could be falsely triggered on TCP segments, the off-path attacker is capable of injecting forged fragments to poison the target TCP traffic after addressing two remaining practical issues, i.e., predicting IPID and deceiving TCP checksum. To trick the victim receiver into mis-reassembling the forged fragments and the benign ones together, the attacker needs to predict the `Identification` field of IP (IPID) of the benign fragments and assign the same value to the forged ones. Our comprehensive measurement study shows that IPID is still predictable on a large number of hosts due to the vulnerable IPID assignment. Besides, the TCP checksum mechanism that checks transmission errors based on the one's complement sum can be easily evaded by crafting equivalents.

We perform two case studies to demonstrate the effectiveness of IP fragmentation attacks against TCP. First, we show that off-path attackers can manipulate HTTP traffic via IP fragmentation. Through injecting forged IP fragments into the target HTTP traffic, attackers can replace the web data that are cached by the victim client in the browser, i.e., poisoning the client's web cache. Moreover, when HTTP redirection happens at web servers, attackers can use forged IP fragments to replace the advertised legal URL with a malicious one, thus intercepting HTTP redirection. Second, we show an off-path attacker can manipulate BGP routing tables by forcing BGP routers to fragment TCP segments and then using forged fragments to poison the advertised BGP messages. We evaluate the impacts of our attacks on the Internet and show that our attacks can be performed to cause serious damages in the real world with a high success rate. Finally, we propose to eliminate the root cause of our attacks by enhancing the PMTUD mechanism to avoid IP Fragmentation at intermediate routers and at hosts.

Contributions. Our main contributions are as follows:

- We reveal that IP fragmentation attacks against TCP are still feasible, and we identify two situations under which an off-path attacker may trigger IP fragmentation on TCP.
- We discover that more than 14% of the Alexa top 10k websites and a considerable number of routers on the Internet suffer from incorrect IP fragmentation on TCP segments.
- We conduct experiments to demonstrate that IP fragmentation can be exploited to poison TCP-based applications, e.g., HTTP and BGP. Our comprehensive evaluation confirms the seriousness of these attacks in the real world.
- We propose to eliminate the root cause of IP fragmentation on TCP using an enhanced PMTUD mechanism.

II. Background

A. IP Fragmentation Attacks

As shown in Table I, due to the fundamental differences of MTUs in heterogeneous networks, IP fragmentation is designed to solve the problem of packet discarding by networks enforcing smaller MTUs. In recent years, IP fragmentation has been widely abused by attackers to cause serious damages in the real world, e.g., evading IDS [5], [67], launching DoS attacks [54], [61], [29], [31], and poisoning DNS caches [41], [11], [42], [86].

TABLE I. MTUs in different networks.

No.	Network	MTU (octet)
1	16 Mbps Token Ring	17914
2	SONET/SDH	4470
3	FDDI	4352
4	Ethernet	1500
5	IEEE 802.11	1452
6	X.25	576
7	NETBIOS	512
8	ARCNET	508
9	IEEE 802/Source-Rt Bridge	508
10	Point-to-Point (low delay)	296

Since different OSes and IDSes may reassemble IP fragments differently [79], attackers can craft fragments with the same IPID and overlapping offsets to the benign fragments. Due to the mis-reassemble by the IDSes, those attacks can defeat the signature analysis and bypassing the detection. However, once the fragments arrive at the end host, the host may handle the overlapped fragments differently and reassemble a malicious packet [5], [67].

[2]Our measurement shows the acceptable minimum path MTUs of Windows 7 Pro, Windows 10 Pro, Windows Server 2008 Standard, Windows Server 2012 Standard, and Windows Server 2016 Standard are all 596 octets.

By leveraging IP fragmentation, an attacker can also prevent a victim host from receiving the original benign packets to launch DoS attacks. For example, when the attacker sends crafted fragments to the victim host, since the crafted fragments may be mis-reassembled with the legitimate ones that have the same IPID, it can produce a corrupted packet. The corrupted packet containing legitimate fragments will be discarded by the victim host, and thus a DoS attack is constructed [29], [31]. Besides, when IP fragments are reassembled at a vulnerable host, they may overflow the host's buffer [54] or lead to a CPU saturation [61], resulting in host crashes.

Moreover, IP fragmentation can be exploited to poison DNS caches [41], [11], [42], [86]. The basic idea is straightforward. First, the attacker forces the DNS server to fragment its reply packets destined to DNS clients, since DNS servers are unable to learn the path MTU value due to lack of path MTU discovery. Second, the attacker crafts malicious fragments with fake DNS records and impersonates the server to send them to the victim client. Finally, the legitimate fragments from the DNS server and the malicious fragments from the attacker will be reassembled incorrectly together at the client and evade the UDP checksum. As a result, the DNS reply is poisoned and cached at the DNS client.

B. Path MTU Discovery (PMTUD)

Since IP fragmentation introduces performance loss and security weaknesses, the path MTU discovery (i.e., PMTUD) mechanism is proposed as a standard to help determine the MTU on the network path between two communication end hosts (i.e., an originator and a destination) [59], [57]. When the Path MTU (PMTU) size is set to the minimum of MTUs of all hops along the entire packet transmission path, IP fragmentation could be avoided.

To discover the PMTU size, the originator sets the `DF` (Don't Fragment) flag in the IP header as `True`, instructing routers along the path not to fragment the packets. When the packet size is larger than an intermediate router's next-hop MTU, the router will discard the packet and reflect its next-hop MTU size to the originator by either issuing an *ICMP Destination Unreachable message* with the code *Fragmentation Needed and DF set* in IPv4 networks or an *ICMPv6 Packet Too Big message* in IPv6 networks[3]. RFC 792 states that at least the first 8 octets besides the IP header of the discarded packet should be embedded in the ICMP error message, indicating the originator to match the message to the appropriate process [69]. According to the newer standard RFC 1812 [7], the ICMP error message may contain up to 576 octets of the discarded packet. After receiving the ICMP error message, the originator reduces the size of subsequent packets according to the router's next-hop MTU. The originator repeats the sending process until a packet with certain size can be forwarded to the destination successfully, and this MTU size will be set as PMTU and maintained at the IP layer.

Unlike UDP, TCP is tightly coupled with IP in the sense that it is aware of the discovered path MTU (maintained in the IP layer) at all times as it will adjust its MSS (a TCP layer construct) based on the path MTU value to actively avoid IP fragmentation. Hence, TCP is widely considered to be immune from IP fragmentation attacks. Instead, most recent attacks against TCP mainly focus on side channel attacks guessing the random ephemeral port numbers, sequence and acknowledgment numbers of the target connection in order to inject forged TCP segments [72], [12], [13], [16], [73]. Surprisingly, in this paper, we show that TCP protocol does suffer from IP fragmentation attacks, especially, an off-path attacker can use malicious fragments to poison TCP traffic without guessing the random sequence numbers and acknowledgment numbers.

[3]In this paper, we do not differentiate ICMP error messages and ICMP "Fragmentation Needed" messages and use them interchangeably.

III. Threat Model

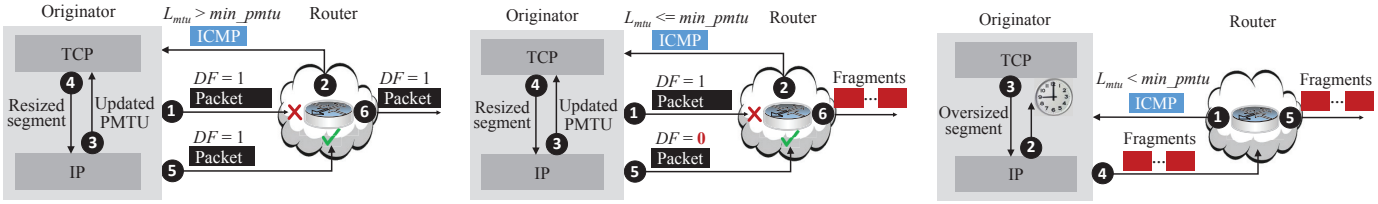
In this paper, we focus on studying the vulnerability of IP fragmentation, which can be exploited to construct off-path poisoning attacks against TCP. By leveraging IP fragmentation, an off-path attacker aims to poison TCP segment data originated from a victim server to a victim client. The off-path attacker is not located on the path between the server and the client and thus cannot eavesdrop their established TCP connections. Instead, the attacker can infer the presence of the victim TCP connection via two typical approaches, i.e., by leveraging an unprivileged application called puppet [30], [72], [16], [73], [32] or by leveraging side channels in TCP/IP implementations [30], [14], [3], [34].

Puppet-Assisted Threat Model. The puppet (e.g., sandboxed scripts at the victim client due to a spam) works at the application layer, which can observe requests from the victim client to the server and notify the attacker about the request stealthily. As a result, the attacker can infer the presence of a victim TCP connection and then perform the attack. Note that the application-layer puppet does not have permissions to access any information of the TCP layer such as sequence numbers. In addition, due to the security mechanisms deployed on the client, e.g., same-origin policy (SOP) [77], the puppet cannot directly tamper with the data sent from the server, it can only notify the attacker of the existence of a victim connection. Puppet-assisted threat model has been widely used by off-path TCP exploits [30], [72], [16], [73], [32], and we use this approach in poisoning HTTP traffic (see Section VI-A).

Side Channel-Assisted Threat Model. The attacker can also infer the presence of a victim TCP connection through side channels discovered in TCP/IP implementations, e.g., by probing `SYN` backlog buffer [14], by leveraging sequential assignment of source ports [30], [34], or by leveraging IPID hash collisions [3], [85]. We use this approach (i.e., IPID hash collisions [3], [85]) to detect the presence of a victim TCP connection when poisoning BGP routing tables (see Section VI-B).

Once the target TCP connection is identified, our attack still needs to fulfill the following requirements:

- **IP Address Spoofing.** The attacker is capable of sending spoofed packets with the IP addresses of the server [24], [12], [13], [55]. IP address spoofing can be easily constructed and is widely utilized in recent off-path attacks [24], [12], [13]. Nowadays, at least a quarter of the Autonomous Systems (ASes) on the Internet do not filter packets with spoofed source IP addresses that leave their networks [53],



(a) No IP fragmentation (b) IP fragmentation at intermediate routers (c) IP fragmentation at originators

Fig. 1. Three modes of handling an ICMP "Fragmentation Needed" message.

and in our experiments, we can easily rent such spoofable attack machines in AS7497.

- **Plaintext Communication.** The communication content originating from the server to the client is not encrypted, and the content that the attacker expects to replace is known, so the attacker can craft acceptable IP fragments to evade the mechanism of TCP checksum. At present, a considerable portion of traffic on the Internet is still not encrypted and can be predicted, e.g., 30% of page loads by Firefox do not use TLS [78]. Besides, network configuration messages (e.g., BGP messages, DNS messages over TCP) are also not full encrypted [56], [28], [36].
- **Predictable IPID.** The attacker can predict the server's IPID and assign the same value to the crafted fragments, thus forcing the benign fragments to be mis-reassembled with the crafted ones. Previous studies show that vulnerable IPID assignment methods are widely used [31], [43], [48], [4], even the mixed IPID assignment adopted by modern Linux systems can also be predicted [3], [85], [24]. We conduct a comprehensive measurement study on IPID prediction and confirm the predictability of IPID in our attacks.

IV. Residual IP Fragmentation on TCP

First of all, in this section, we conduct a systematic study on the handling modes of ICMP "Fragmentation Needed" messages in current PMTUD implementations to discover and measure residual IP fragmentation on TCP. According to PMTUD, after a TCP connection is established, the originator first generates TCP segments based on the initialized MSS value (e.g., 1460 octets in Ethernet), and sets the `DF` flag in IP header as `True`. During being forwarded, if the packet size exceeds the next-hop MTU of an intermediate router, the router will discard the packet and issue an ICMP "Fragmentation Needed" message to the originator. By performing measurement study from a 10Gbps backbone IPv4 router on the Internet, we identify three different modes for originators to handle the ICMP "Fragmentation Needed" messages, and we discover that two of the three modes may incur IP fragmentation on TCP segments.

Mode 1: No IP Fragmentation. When an oversized packet is discarded by an intermediate router, the router reflects an ICMP "Fragmentation Needed" message to the originator, as shown in Figure 1(a). According to the PMTUD specification, the issued ICMP error message usually carries the intermediate router's next-hop MTU value L_{mtu} by default. After the originator receives the ICMP error message, it first checks whether the TCP sequence number embedded in the message is located in its send window. If the ICMP error message passes this check, the originator then compares L_{mtu} with its min_pmtu which is a system variable in the PMTUD implementations to indicate the acceptable minimum PMTU value for the system (e.g., 552 octets in Linux by default). If L_{mtu} is greater than min_pmtu, the originator resizes the MSS of subsequent TCP segments to L_{mtu} - 40 and sets the `DF` flag of the packets as `True` before sending them out. This mode can effectively avoid IP fragmentation on TCP segments and has been widely adopted on almost all major OSes, including Linux kernel version 2.6.32 and beyond, FreeBSD version 8.2∼12.1, Mac OS 10.11∼10.15, and Windows.

Mode 2: IP Fragmentation at Intermediate Routers. As shown in Figure 1(b), different from the first mode, if the intermediate router's next-hop MTU value L_{mtu} is less than or equal to the originator's acceptable minimum PMTU value min_pmtu, the originator will resize the MSS of subsequent TCP segments to min_pmtu - 40, instead of L_{mtu} - 40. Christopher [76] pointed out that originators equipped with Linux systems resize subsequent packets in this way if an intermediate router reflects such an ICMP error message, i.e., ignoring the router's MTU value and always assembling IP packets with the size of 552 octets.

By investigating the entire interactions between the originator and the intermediate router, we discover that the originator will not only resize the MSS of subsequent TCP segments to min_pmtu - 40, but also set the `DF` flag as `False`. As a result, if one intermediate router's L_{mtu} is less than the originator's min_pmtu, IP fragmentation will occur at the router though the originator may assemble IP packets with the size of min_pmtu permanently [76]. Linux kernel version 2.6.32 and beyond, FreeBSD version 8.2∼12.1, Mac 10.11∼10.15, and Windows all adopt this mode[4]. In RFC 1191 [59], the min_pmtu is recommended as 576 octets. However, we observe that the detailed implementations may adopt different values in the real world, e.g., 256 octets in FreeBSD version 8.2∼12.1, 296 octets in Mac OS 10.11∼10.15, 552 octets in Linux 2.6.32 and beyond, and 596 octets in Windows.

This mode has been popularly employed on the Internet. According to the study on 121 CAIDA datasets collected from 2008 to 2016, Göhring *et al.* uncover 2.9 million ICMP "Fragmentation Needed" messages with the values of next-hop MTU ranging from 0 to 43205 octets [37]. It can be found that the next-hop MTU of a large number of routers on the Internet are very small, such as 576 octets, 296 octets, 100 octets, and even 68 octets. For instance, as exposed in the datasets, there are more than 900 routers whose next-hop MTU is 576 octets. These routers prefer to fragment TCP segments, even if PMTUD is performed by the originator. For example, routers with the next-hop MTU of 576 or smaller

[4]Windows and Mac OS clear the `DF` flag directly, and they do not resize the MSS of subsequent TCP segments.

octets will certainly fragment TCP segments originated from systems whose acceptable minimum path MTU is greater than 576 octets, e.g., Windows whose min_pmtu is 596 octets[5].

We also conduct two types of measurement studies, i.e., a passive measurement and an active measurement, to discover problematic routers on the Internet. In the passive measurement study, we filter ICMP "Fragmentation Needed" messages (i.e., with small next-hop MTU value of less than 596 octets) and then identify problematic routers based on the flowing-through IPv4 traffic of our 10 Gbps backbone IPv4 router[6]. From our vantage point, we totally capture 8,365 ICMP "Fragmentation Needed" messages with small next-hop MTU values between December 16th, 2020 and January 20th, 2021, as shown in Figure 2. These ICMP error messages are reflected by 602 problematic intermediate routers distributed in 80 ASes and 28 countries, which are prone to fragment the flowing-through TCP segments due to the extremely small next-hop MTU value (i.e., less than 596 octets). Our measurement results also show that the mechanism of PMTUD has been well implemented by intermediate routers on the Internet. When an oversized packet with the `DF` flag setting to `True` is received, the router will follow PMTUD and reflect an ICMP error message to the originator, instead of simply ignoring the `DF` flag [84]. However, due to the router's extremely small next-hop MTU, the originator will clear the `DF` flag directly and allow TCP segments to be fragmented by the router.

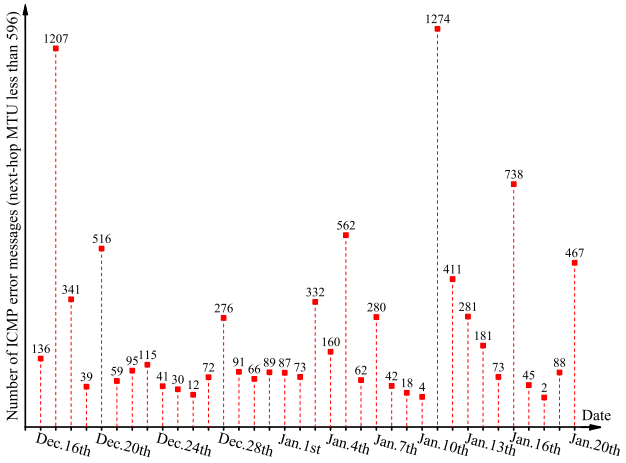


Fig. 2. ICMP error messages with small next-hop MTU.

Table II illustrates the distribution of the identified problematic routers in the top 20 ASes, and Table III presents the details of 30 problematic routers. Figure 3 shows the cumulative distribution function (CDF) of the identified 602 problematic routers' next-hop MTU value. Since the measurement is only performed on one router on the Internet, we believe that a larger number of problematic routers are deployed and used in practice. Note that a problematic router is harmful to all the flowing-through traffic, and TCP segments may flow through many intermediate routers to the destination, as long as one problematic router is on the path, TCP segments will be fragmented.

[5]CAIDA declined our request on newer datasets due to non-technical reasons; instead, we conduct our own measurement study on the Internet and make similar observations.

[6]Due to anonymity and ethical considerations, we do not expose the locations of our backbone router owned by a Tier-2 ISP.

TABLE II. NUMBER OF PROBLEMATIC ROUTERS IN THE TOP 20 ASES.

AS No.	Organization	Router	AS No.	Organization	Router
AS56047	China Mobile	69	AS9808	China Mobile	16
AS4134	China Telecom	47	AS38197	Sun Network(HK)	15
AS1267	WIND TRE S.P.A.	41	AS8103	Florida Services	15
AS4760	HKT Limited	23	AS4847	China Networks	14
AS4837	China Unicom	22	AS4812	China Telecom	14
AS3462	Chunghwa Telecom	21	AS37963	Hangzhou Alibaba	14
AS8452	TE-AS	20	AS48503	Mobile Telecom	13
AS17816	China Unicom	19	AS18881	TELEFÔNICA	13
AS45090	Shenzhen Tencent	17	AS4538	CERNET	7
AS17621	China Unicom	16	AS132203	Tencent Building	5

Besides, we deploy five vantage points in California, Singapore, Beijing, Hong Kong, and London to perform our active measurement study. From these vantages, we send ICMP request packets to Alexa top 1 million websites. To extend our measurement scale, we also randomly generate 12 percent of the entire IPv4 address space within 20 days as the destination of our request packets. The length of the request packets is set to 1500 octets and the `DF` flag in the packet header is set to `True`. Hence, if the next-hop MTU of an intermediate router is less than 1500 octets, an ICMP "Fragmentation Needed" message carried with the MTU value will be reflected to our vantage points. Within 20 days, we totally capture 2,721 ICMP "Fragmentation Needed" messages with the next-hop MTU value less than 596 octets. Among the 2,721 ICMP "Fragmentation Needed" messages, 167 are reflected during probing Alexa top 1 million, and 2,554 are captured during probing 12% of the Internet. Our active measurement study demonstrates the feasibility of identifying problematic routers from end hosts. Once a problematic router is identified, attackers can launch our attacks to poison the TCP traffic traversing the router.

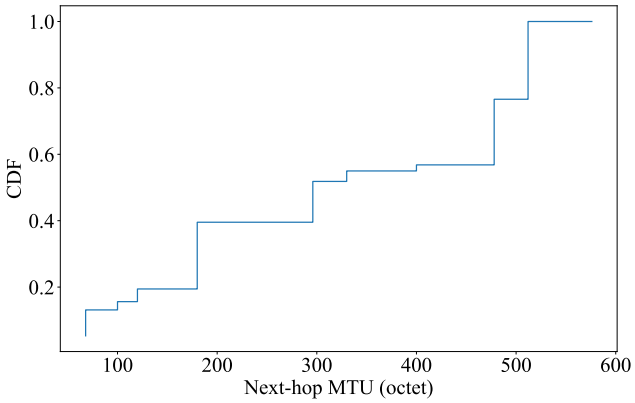


Fig. 3. CDF of the problematic routers' next-hop MTU.

Mode 3: IP Fragmentation at Originators. As shown in Figure 1(c), when an ICMP "Fragmentation Needed" message is not triggered by TCP connections (instead, by UDP or ICMP echo request/reply), even if the originator receives the ICMP error message that carries a next-hop MTU value L_{mtu} less than the originator's min_pmtu, the originator will not notify its TCP layer to resize the MSS of subsequent segments promptly due to the lack of TCP related information such as source port, destination port, or sequence number. The IP layer of the originator defers the feedback of the updated PMTU until it receives segments from TCP and perceives the presence of the TCP connection passively. The deferred response results

TABLE III. PROBLEMATIC ROUTERS ON THE INTERNET PRONE TO FRAGMENT TCP SEGMENTS.

No.	Problematic router	Next-hop MTU (Octet)	AS No.	Organization	Location	Victim originator
1	58.213.x.89	100	AS4143	China Telecom	CN	①, ②, ③, ④ [1]
2	111.49.x.171	120	AS9808	China Mobile	CN	①, ②, ③, ④
3	113.81.x.103	512	AS4134	China Telecom	CN	③, ④
4	171.11.x.83	576	AS4134	China Telecom	CN	④
5	45.113.x.146	576	AS137697	China Telecom	CN	④
6	120.32.x.231	576	AS4134	China Telecom	CN	④
7	61.190.x.218	576	AS4134	China Telecom	CN	④
8	101.4.x.2	576	AS4538	CERNET	CN	④
9	123.204.x.83	68	AS4780	Digital United Inc.	TW	①, ②, ③, ④
10	36.229.x.31	68	AS3462	Chunghwa Telecom Co., Ltd.	TW	①, ②, ③, ④
11	210.71.x.67	296	AS3462	Chunghwa Telecom Co., Ltd.	TW	③, ④
12	168.95.x.73	330	AS3462	Chunghwa Telecom Co., Ltd.	TW	③, ④
13	177.132.x.159	576	AS18881	TELEFÔNICA BRASIL S.A	BR	④
14	177.133.x.153	576	AS18881	TELEFÔNICA BRASIL S.A	BR	④
15	177.11.x.83	576	AS262882	Prefeitura Municipal de Bauru	BR	④
16	69.83.x.197	296	AS6167	Cellco Partnership, Inc.	US	③, ④
17	192.151.x.166	576	AS40065	CNSERVERS LLC	US	④
18	123.108.x.95	180	AS58895	Ebone Network Limited	PK	①, ②, ③, ④
19	202.69.x.125	400	AS23750	GERRYS INFORMATION	PK	③, ④
20	121.127.x.23	576	AS38197	Sun Network (Hong Kong) Limited	HK	④
21	117.18.x.220	576	AS38197	Sun Network (Hong Kong) Limited	HK	④
22	95.174.x.72	478	AS9009	M247 Ltd	GB	③, ④
23	37.120.x.70	478	AS9009	M247 Ltd.	GB	③, ④
24	102.134.x.55	512	AS36874	Cybersmart	ZA	③, ④
25	103.146.x.150	512	AS139841	STAR COMMUNICATION	BD	③, ④
26	149.255.x.21	576	AS50710	EarthLink Ltd.	IQ	④
27	60.253.x.86	576	AS9981	Saero Network Service LTD.	KR	④
28	190.121.x.82	576	AS27717	Corporacion Digitel C.A.	VE	④
29	88.87.x.109	576	AS34606	B.B.Bell SPA	IT	④
30	45.119.x.190	576	AS131386	Long Van System Solution JSC	VN	④

[1] ① represents FreeBSD 8.2∼12.1, ② represents Mac OS 10.11∼10.15, ③ represents Linux 2.6.32 and beyond, and ④ represents Windows.

in the desynchronization on the PMTU value between the IP layer and the TCP layer for a short period of time, causing TCP to write oversized segments to IP. Thereafter, IP will fragment those oversized TCP segments using its min_pmtu, and set the `DF` flag of the fragments as `False`. This responding mode has been implemented in Linux kernel version 3.8.1 and beyond.

Since current PMTUD implementations do not validate the source and transmission path of ICMP error messages, attackers may pretend to be a router and forge such an ICMP error message to trick the originator into fragmenting TCP segments, as long as the embedded data in the forged message can evade the originator's check and the next-hop MTU specified in the message is smaller than the originator's min_pmtu. We discover that a forged ICMP "Fragmentation Needed" message embedded with an echo reply can evade the originator's check and trick the originator into fragmenting TCP segments, which will be elaborated later in Section V-C.

To evaluate the impacts of this vulnerable handling mode on the Internet, we measure and identify how many websites in the list of the Alexa top 10k websites are vulnerable and suffer from the incorrect IP fragmentation on TCP segments. We observe that 1,407 websites among the Alexa top 10k websites are vulnerable to the forged ICMP "Fragmentation Needed" messages sent from our vantage point in California, which can be tricked into fragmenting their TCP segments even if the PMTUD mechanism is enabled. These vulnerable websites are potential victims of our attacks. In addition, 661 websites in the top 10k websites list cannot be reached from our vantage point, hence the actual number of vulnerable websites may be more than 1,407. Figure 4 presents the geographical distribution of the vulnerable websites we detected[7].

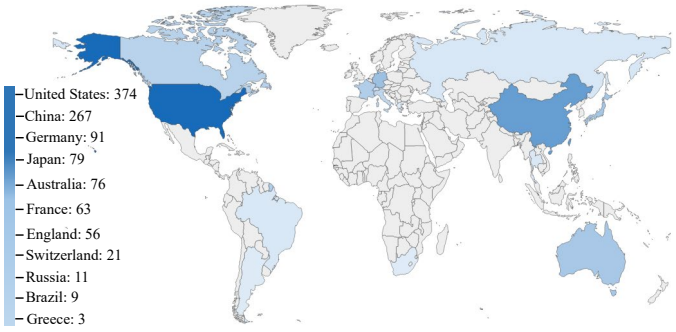


Fig. 4. Geographical distribution of vulnerable websites.

To sum up, since the received ICMP error messages may vary greatly (e.g., different MTUs and different triggers carried in the messages), originators with different OS implementations handle them in different ways for performance considerations (see Figure 1(b)) or for exceptions handling (see Figure 1(c)), rather than just decreasing the MSS of TCP segments accordingly (see Figure 1(a)). However, the two special handling modes incur undesired IP fragmentation on TCP.

[7]We deploy five vantage points in different locations around the world, each with different filtering policy enforced by the ISPs. California is the most friendly vantage, from which we can reach 9,339 websites.

V. TCP Poisoning via IP Fragmentation

After identifying the situations under which IP fragmentation on TCP segments can be triggered, we now present the attacks that exploit IP fragmentation to poison TCP segments.

A. Attack Overview

Figure 5 shows the overview of our poisoning attack against TCP via IP fragmentation. It consists of three main steps. In the first step, the off-path attacker infers the presence of a victim TCP connection between a server and a client. For different cases, it can be achieved by using two classic approaches, i.e., unprivileged applications or side channels. In the second step, the attacker triggers IP fragmentation on the server's TCP segments by exploiting the two residual IP fragmentation situations discovered in Section IV. In the third step, the attacker impersonates the victim server and forges malicious fragments to the victim client, these forged fragments will be cached in the client[8]. Finally, the victim client will mis-reassemble benign fragments (originated from the server) and the forged ones that carry correct IPID value together. Once the checksum mechanism of TCP can be evaded, the original TCP segment will be poisoned, which circumvents the guessing of sequence and acknowledgment numbers that are always carried in the first benign fragment. Next, we detail the three steps of our attack.

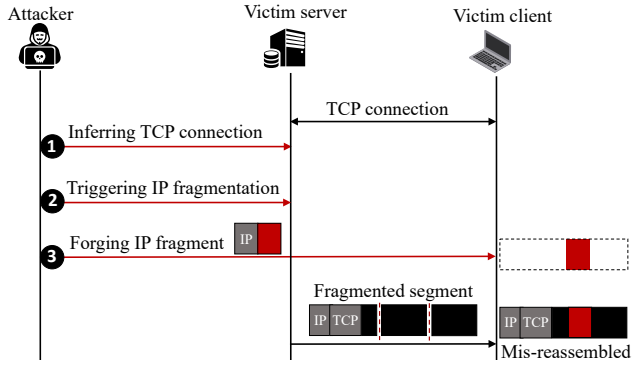


Fig. 5. Overview of TCP poisoning via IP fragmentation.

B. Inferring TCP Connections

Since the off-path attacker does not sit on the communication path between the server and the client, it cannot eavesdrop the established TCP connections. Instead, the attacker needs to infer the presence of the TCP connection between the server and the client before launching TCP poisoning attacks. A large number of previous works have studied how to infer the presence of TCP connections [30], [72], [16], [73], [15], [32], [14], [3]. For different cases, the attacker can adopt two classic approaches to infer the presence of TCP connections.

First, the attacker can induce the victim client to install an unprivileged application or visit a web page to load sandboxed scripts (i.e., puppet) [73], [32]. The puppet observes requests from the victim client to the server, and notify the attacker about the request stealthily. It is generally considered a weak puppet without capabilities to launch other malicious manipulations, and this approach has been widely used by off-path TCP exploits [30], [72], [16], [73], [32]. We use this approach to learn the presence of a victim HTTP connection (see Section VI-A).

Secondly, side channels in TCP/IP implementations (e.g., shared `SYN` backlog buffer [14], sequential assignment of source ports [30], [34] and IPID hash collisions [3], [85]) can also be used to infer the presence of a victim TCP connection. We leverage IPID hash collisions [3], [85] to detect the presence of a victim BGP connection on port 179 (see Section VI-B).

C. Triggering IP Fragmentation

According to our measurement study in Section IV, the attacker may trigger IP fragmentation on TCP segments by either detecting IP fragmentation occurring at intermediate routers or forcing IP fragmentation occurring at originators (the server in our attack) of the TCP connections.

(1) Detecting IP Fragmentation at Routers. To detect an intermediate router whose next-hop MTU value L_{mtu} is less than the server's acceptable minimum PMTU value min_pmtu, an attacker can first create a large-sized IP packet and set the `DF` flag of the packet to `True`, then send the packet to the client. If an ICMP "Fragmentation Needed" message is reflected by one intermediate router, the attacker records the next-hop MTU value carried in the message, then resizes the packet according to this value and sends a new packet to the client again. This process repeats until no ICMP "Fragmentation Needed" message is reflected.

If the ICMP error message sent from an intermediate router satisfies two conditions, i.e., (i) the next-hop MTU of the router is less than the server's system variable min_pmtu and (ii) the router is on the path from the server to the client, then TCP segments from the server to the client will be fragmented and the TCP-based applications running on the client could be poisoned. In order to check the first condition, the attacker can probe the server's OS type [62] and identify its min_pmtu in advance (since each type of OS has its own fixed min_pmtu), then compare the identified min_pmtu with the next-hop MTU value carried in the reflected ICMP error message. Secondly, by tracing the path from the network where the server is located to the network where the client is located [1], [51], the attacker can identify if the problematic router is on the traced path by locating the router's IP address that is carried in the reflected ICMP error message.

In practice, for a given TCP connection, it may not be easy for off-path attackers to detect a problematic router on the path. When this happens, the attacker needs to probe problematic routers first (e.g., gateways of subnets) and then identify potential victim TCP connections whose traffic may traverse the router.

(2) Forcing IP Fragmentation at the Server. Attackers can also force IP fragmentation occurring at vulnerable servers by actively sending forged ICMP error messages to the server. In our attack, the first step is to make the server accept the

[8]Major OSes (i.e., MacOS, Linux and Windows) adopt the fragment overlapping policy of '*first*' or its variants [5], [79], [63], [6], which means the victim client will cache and accept IP fragments that arrive first (see Appendix Section D for more details about fragment overlapping policies). Hence, attackers have adequate time to perform IP fragmentation attacks, they can inject forged fragments in advance and then wait for legitimate ones to arrive [11], [86].

forged ICMP error messages. According to RFC 792 [69], the forged ICMP error message should be embedded at least 28 octets (i.e., the IP header plus at least the first 8 octets) of a triggering packet to pass the server's check. More strict, according to the newer standard RFC 1812 [7], ICMP error messages should be embedded as much of a triggering packet as possible, usually as long as it does not exceed 576 octets. As a result, the attacker has to craft and embed feasible data into the forged ICMP error message to evade the server's check. Early Linux kernels do not apply any checks on the received ICMP error messages embedded with UDP headers, so it is possible to forge an ICMP error message embedded with a UDP data to evade the server's check and trick it into fragmenting TCP segments. Fortunately, these vulnerabilities have been fixed since Linux kernel 4.18 by checking if the server has a UDP session with the destination.

We discover that when an attacker sends a forged ICMP "Fragmentation Needed" message embedded with an ICMP echo reply data to the server equipped with Linux kernel version 3.8.1 and beyond, the server will be tricked into accepting the forged ICMP error message and then fragmenting TCP segments incorrectly due to the triggered desynchronization on PMTU value between the IP layer and the TCP layer. When a Linux-based server receives an ICMP "Fragmentation Needed" message embedded with an echo reply data, if the server already has a TCP connection to the address (the victim client) specified in the embedded echo reply data, the server's IP layer does not actively notify its TCP layer about the updated PMTU value (usually reduced by the ICMP message), thus causing desynchronization on the PMTU value between the two layers. The IP layer defers the feedback of the updated PMTU value to the TCP layer until it receives TCP segments.

After receiving the oversized TCP segments, the IP layer of the server fragments these oversized segments using its min_pmtu, sets the `DF` flag to `False`, and then sends them out (see Figure 1(c)). The number of fragments depends on how many TCP segments are written into the IP layer by the previous write action of the TCP socket, usually related to the size of the socket's congestion window $SND.CWND$. After IP is informed of the presence of the TCP connection, it then notifies the TCP layer about the updated PMTU value. TCP will reduce the MSS of subsequent TCP segments to min_pmtu - 40, and IP sets the `DF` flag of the resized packets to `False` before sending them out. This vulnerability impacts most Linux systems (i.e., kernel version 3.8.1 and beyond).

Attackers can forge an ICMP "Fragmentation Needed" message as shown in Figure 6 and send it to the server, thus forcing the server to fragment TCP segments. By simply padding 'X' to meet the length requirement, the attacker can evade the checks from both RFC 792 and RFC 1812. A next-hop MTU value L_{mtu} less than the server's acceptable minimum PMTU value min_pmtu is specified in the ICMP header. Since we observe that if L_{mtu} is equal to or greater than min_pmtu, the IP layer at the server will actively identify the corresponding sockets at the TCP layer to force the adjustment of MSS immediately (see Figure 1(a)) to avoid IP fragmentation. In practice, L_{mtu} can be set to 68 octets, which is the minimum of PMTU values on the Internet. The forged ICMP message is embedded with an echo reply data, which can be accepted by the server and cause the desynchronization on PMTU value. The message will reduce the PMTU value of the server's IP protocol but defer the feedback of the reduced PMTU to TCP, thus causing TCP to write oversized segments to IP and incurring IP fragmentation on these TCP segments. We see that it is easy for an attacker to forge such an ICMP "Fragmentation Needed" message.

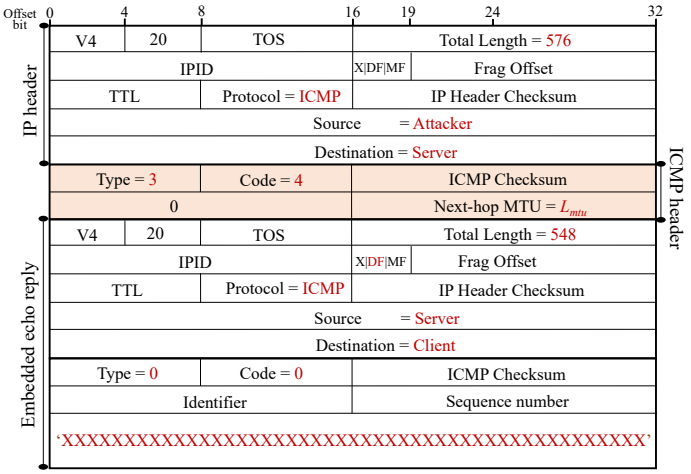


Fig. 6. The forged ICMP error message.

D. Forging IP Fragment

Once IP fragmentation could be falsely triggered on TCP segments, it is easy for off-path attackers to overcome the remaining practical issues (i.e., predicting IPID and deceiving TCP checksum) and thus successfully perform poisoning attacks against TCP.

(1) Predicting IPID. IPID indicates the uniqueness of a packet during fragmentation and reassembly [70], [81], so attackers need to predict IPID of the benign fragments and assign the value to the forged ones. The success rate of predicting IPID depends on the IPID generation algorithms, which can be classified into five categories including *global counter-based IPID assignment*, *per-destination IPID assignment*, *per-connection IPID assignment*, *random IPID assignment*, and *hash-based IPID assignment* [31], [3], [43], [50], [75].

Previous studies show that IPID generated by the global counter-based IPID assignment algorithm and the per-destination IPID assignment algorithm is predictable [31], [43], [48], IPID noises (the increments during the short period of attack window) will not significantly interfere with the prediction [65], [66]. Though the hash-based IPID assignment algorithm and the random IPID assignment algorithm are more sophisticated and secure, their generated IPID may still be predictable due to various implementation flaws [3], [85], [4].

New Linux releases (i.e., Linux kernel version 3.16 and beyond) assign IPID to TCP packets using the per-connection IPID assignment algorithm, which is considered to be more secure. However, this algorithm uses the packet's `DF` flag to check if the packet contains a TCP segment and then enables the per-connection IPID assignment algorithm [3], [24]. By triggering the fragmentation on TCP segments, attackers can clean the `DF` bit in the packet header and force a fallback to use the 2048 global shared hash-based IPID counters [24], [10], [86]. Each of the hash-based IPID counter is shared by an array of destination IPv4 and IPv6 addresses. As a result,

it is easy for attackers to predict the IPID by leveraging hash collisions [3], [85], [24][9]. On average, we can use adequate IPv6 addresses to successfully identify a hash collision and predict the IPID in 22 seconds, as shown in Figure 7. Note it is not necessary to use IPv6 addresses for IPID predictions. Since the 2048 global hash-based IPID counters are shared between IPv4 and IPv6 addresses, if attackers own IPv6 addresses, it may be easier (due to the large IPv6 address space) to construct hash collisions and predict IPIDs.

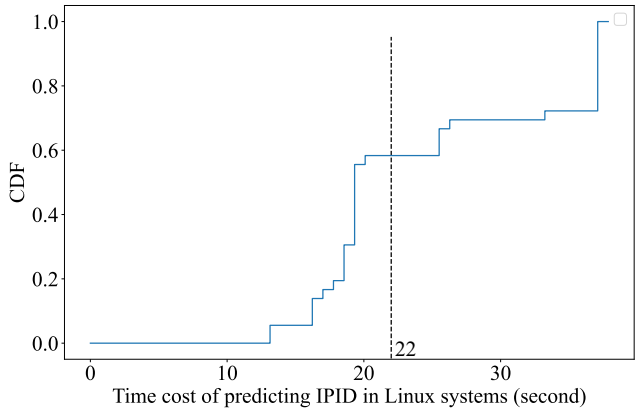


Fig. 7. CDF of time cost when predicting IPID in modern Linux systems.

In practice, we do not need to accurately predict IPID when performing our attacks. Instead, we can send as many forged fragments (with different IPID) as the defragmentation cache size (64 in Linux and 100 in Windows) in parallel, and the attack will succeed once one fragment hits. To evaluate the feasibility of predicting IPID, we survey the IPID generation algorithms adopted in various TCP/IP stack implementations. Table IV shows our survey results in different stack implementations. When analyzing the predictability, we do not consider software vulnerabilities. From Table IV, it can be convinced that the IPID assigned by many prevailing OSes still can be predicted. We also conduct a measurement study to identify the distributions of IPID assignment on the Internet, the measurement results show that a large number of hosts on the Internet still adopt vulnerable IPID assignment (see Appendix Section B for more details).

(2) Deceiving TCP Checksum. When all IP fragments arrive at the destination and are reassembled into one TCP segment, the TCP checksum field will be checked to detect transmission errors. The checksum field is a 16-bit word, deriving from the one's complement sum of all 16-bit words in the header and payload, where a 12-octet TCP pseudo header is also covered [71]. To deceive the TCP checksum and overwrite the original data areas in the segment via IP fragments, the attacker needs to know the contents of the packet that it expects to replace with the forged fragment.

We discover that by simply adjusting two octets, the TCP checksum mechanism can be easily circumvented, similar to deceiving the UDP checksum [11], [86]. Supposing the attacker aims to replace the original k 16-bit words in the TCP segment, it can forge arbitrary $k-1$ 16-bit words and use the remaining one 16-bit word (i.e., two octets) to align the original checksum, thus crafting an equivalent. Appendix Section C presents more details and examples about how to evade TCP checksum.

[9]Our measurements show that IPID assignment of modern Android systems is also vulnerable due to the adoption of Linux kernels.

VI. Case Study

In this section, we perform two case studies that use IP fragmentation to manipulate HTTP traffic and BGP routing tables. We can pick either fragmentation at intermediate routers or at vulnerable hosts, and we have succeeded either way in the two case studies.

Ethical considerations. Our experiments avoid causing real damages on the Internet. When performing IP fragmentation attacks against public websites, we exploit the vulnerability of IP fragmentation occurring at intermediate routers instead of hosts, i.e., we do not expose and exploit vulnerabilities of the target websites. When performing IP fragmentation attacks against BGP, we exploit the vulnerability of IP fragmentation occurring at vulnerable hosts, where all the hosts involved in this experiment are test-bed machines in our lab environment. The exploits are not exclusive, e.g., fragmentation occurring at vulnerable hosts can also be exploited to poison HTTP and vice versa.

A. Manipulating HTTP Traffic

Attack Setup. This attack involves four hosts, including 1) a web server on the Internet that provides online services, e.g., advertising, or Internet Banking, 2) an attack machine equipped with Ubuntu 14.10 performing the off-path poisoning attack, 3) a victim client that accesses the web server based on HTTP, and 4) a phishing website controlled by the attacker. The phishing website tricks the client into downloading a puppet, i.e., malicious JavaScript codes [26], [83] to help infer the TCP connections to the server, which has been widely used by previous off-path TCP exploits [30], [72], [16], [73], [32]. Due to the security policy of SOP [77], the puppet cannot tamper with the web data on the client. Both the attack machine and the victim client are located in our lab, and the off-path attack machine aims to manipulate the HTTP traffic sent from the server to the client. To measure the attack success rate on different client OSes, we install Windows 10 Pro, Mac OS 10.14 and Ubuntu 20.10 on the client separately (Section VII will evaluate the attack success rate).

Attack Procedure. As shown in Figure 8, ❶a the off-path attacker first conducts an early probing on the victim server's public web data, IPID assignment and the global system variable of min_pmtu. ❶b Meanwhile, due to a spam [44], the client downloads and installs a puppet. ❷ In the victim client, the puppet observes HTTP requests to the victim server and notifies the attacker stealthily, hence the attacker can learn an HTTP session existing between the server and the client. ❸ The attacker triggers IP fragmentation on the server's TCP segments. In this case study, we pick the situation of IP fragmentation occurring at routers. The attacker generates packets (with the `DF` flag equal to `True`) according to the MTU of typical network interfaces (e.g., 1500 octets of Ethernet) and then sends the packets to the victim client. When an ICMP "Fragmentation Needed" message is reflected, if the next-hop MTU value carried in the message is less than the previously probed min_pmtu and the problematic router

TABLE IV. IPID ASSIGNMENT IN DIFFERENT OS IMPLEMENTATIONS.

Operating system	Version	IPID assignment	Initial value	Increment	Predictable
Linux	Kernel 3.0.10	per-destination	0	1	✓
	Kernel 4.4.20	per-connection	Hash value	Uniform distribution	✓
	Kernel 4.10.6	per-connection	Hash value	Uniform distribution	✓
	Kernel 4.16.10	per-connection	Hash value	Uniform distribution	✓
	Kernel 4.20.16	per-connection	Hash value	Uniform distribution	✓
	Kernel 5.0.10	per-connection	Hash value	Uniform distribution	✓
	Kernel 5.5.6	per-connection	Hash value	Uniform distribution	✓
	Kernel 5.9.2	per-connection	Hash value	Uniform distribution	✓
	Kernel 5.10.36	per-connection	Hash value	Uniform distribution	✓
Windows	XP Service Pack 3	global counter-based	0	1	✓
	7 Pro	global counter-based	0	1	✓
	7 Service Pack 1	global counter-based	0	1	✓
	8.1	hash-based	Hash value	Hash value	
	10 Pro	hash-based	Hash value	1	
	Server 2008 Standard	global counter-based	0	1	✓
	Server 2012 Standard	hash-based	Hash value	Hash value	
IOS & IoT devices	Android 4.1	per-destination	0	1	✓
	Android 7.0	per-connection	Hash value	Uniform distribution	✓
	Android 9	per-connection	Hash value	Uniform distribution	✓
	Android 10	per-connection	Hash value	Uniform distribution	✓
	Android 11	per-connection	Hash value	Uniform distribution	✓
	Cisco IOS 15.3	global counter-based	0	1	✓
	VirtualBox NAT	global counter-based	0	1	✓
	lwIP	global counter-based	0	1	✓
	uIP	global counter-based	0	1	✓
Mac & BSD	Mac OS	random	–	–	
	FreeBSD	random	–	–	
	OpenBSD	random	–	–	

issuing the message is located on the traced path between the server and the client, the attacker detects IP fragmentation on the server's TCP segments[10]. ❹ The attacker pretends to be the server and forges malicious IP fragments to be cached at the client. ❺ Benign fragments from the server and the forged ones with the same IPID are mis-reassembled at the client and pushed to TCP. Once the reassembled segment evades TCP checksum, the attack succeeds in poisoning the HTTP traffic received by the client.

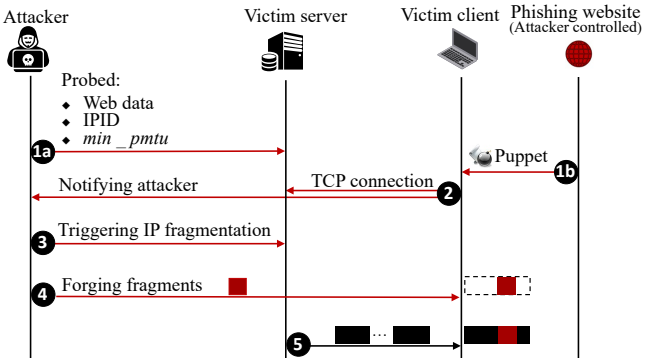


Fig. 8. Manipulating HTTP traffic via IP fragmentation.

In the following, we present two attack scenarios that can successfully poison the web cache and intercept the HTTP redirection by manipulating the HTTP traffic.

Attack Scenario 1: Web Cache Poisoning. Web caching is a popular technique to reduce the bandwidth and the pressure of back-end servers [8]. By temporarily storing network resources requested by clients (e.g., HTML pages and images), subsequent client requests may be satisfied from the local cache. By manipulating the HTTP traffic, attackers may launch web cache poisoning attacks against the integrity of a web cache repository, in which genuine content cached for an arbitrary URL is replaced with spoofed content. Stealthy web cache poisoning is a serious threat in the real world [30], [32]. A problematic router that is the last hop on the path to the client (i.e., the gateway of the client) is identified, the router will fragment TCP segments originated from websites whose min_pmtu is greater than 296 octets.

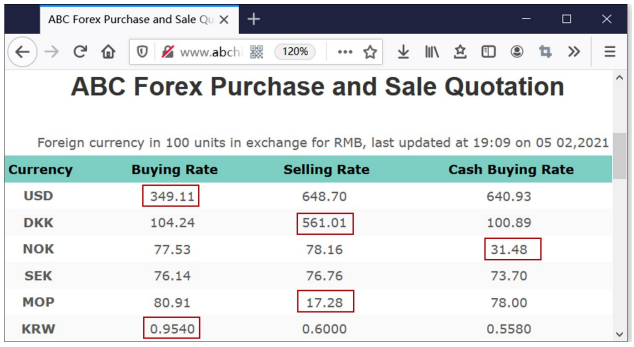


Fig. 9. Snapshots of web cache poisoning.

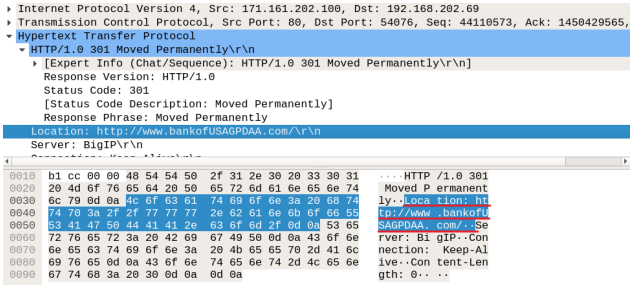
After the victim client receives all fragments, a mis-reassembled TCP segment is reconstructed and delivered to the HTTP application. Figure 9 shows the snapshot of our web cache poisoning attack against Agricultural Bank of China (ABC). The original exchange rate of 646.11, 105.07, 75.04, 80.91 and 0.5580 are stealthily replaced by 349.11, 561.01, 31.48, 17.28 and 0.9540, respectively. These fake data will be cached in the victim's browser, when users access the same

[10]In practice, the detection of problematic routers on the path to victim clients can be performed before the establishment of target TCP connections, i.e., once the puppet is installed, the attacker can be notified to probe problematic routers in advance.

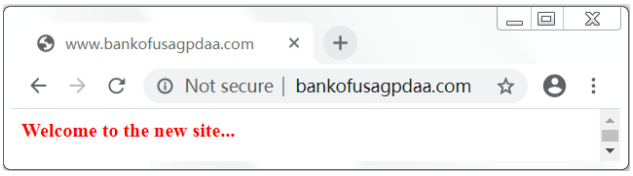
target website later, they will see the cached fake data, and not aware of being attacked. In this attack, the security mechanism of SOP [77] is circumvented. Appendix Section A presents more experimental results about web cache poisoning, and we evaluate the attack success rate in Section VII.

Attack Scenario 2: HTTP Redirection Intercepting. HTTP redirection is a popular technique to direct a request to a new domain or URL in various scenarios, e.g., switching a request of HTTP to HTTPS, forwarding a request of an abandoned domain name to the new one. In this attack scenario, we implement and evaluate the attack that intercept HTTP redirection via IP fragmentation. In our experiment, the web server enables the HTTP redirection mechanism. If the client requests resources hosting on the web server through an HTTP request, the server sends the client an HTTP redirection message with HTTP status code 301 and automatically redirects the request to a more secure one, i.e., a new website secured by TLS. In practice, the redirection depends on the new URL carried in the redirection message from the server. After receiving the redirection message, the client automatically requests this new URL and then finish the redirection. We show that the URL can be replaced by attackers via malicious IP fragments, so that the client's requests are hijacked to a fake website controlled by the attacker.

We use the site of Bank of America as the target web server. Figure 10 shows that when the client accesses the server with the URL of "http://www.bankofamerica.com/", the new advertised URL to the client is replaced by the attacker via IP fragmentation, and the redirection is hijacked to the attacker's fake website.



(a) The legal URL of "https://www.bankofamerica.com" carried in the HTTP redirection message is replaced by a malicious one.



(b) The HTTP request from the victim client is hijacked to a malicious site.

Fig. 10. Snapshots of HTTP redirection intercepting.

B. Manipulating BGP Routing Table

This case study shows that IP fragmentation on TCP can be exploited to manipulate BGP routing tables. Though the TCP MD5 option is available in some commercial BGP routers [40], BGP session protection with MD5 is not enabled by default [19]. We communicate with network operators of several big ISPs and confirm that the security mechanisms, e.g., BGP MD5 and GTSM [35], are not normally enabled in practice due to the issues of management complexity and performance loss[11]. Moreover, since the MD5 option is implemented at the TCP layer, OSes releases may not support it by default. For example, Linux systems do not support MD5 option by default and do not allow manually turning on this option by configuration. Instead, users have to reconfigure and recompile the kernel to enable MD5. Hence, BGP routing systems built on those OSes, e.g., the widely deployed open-source Quagga routing suite [45], will be vulnerable to our attacks.

Attack Setup. Three hosts are involved in this attack, an attack machine equipped with Ubuntu 14.10, and two BGP routers termed as $Router_i$ and $Router_j$. We use two software BGP routers that are equipped with Linux kernel version 4.16.10 and the BGP suite of Quagga with version 1.2.2. $Router_i$ receives BGP routing information generated from other ASes and then advertises the routing information to its peer $Router_j$, which is connected by a persistent TCP connection. The attacker aims to overwrite the advertisements announced from $Router_i$ to $Router_j$ by leveraging IP fragmentation and thus inject fake BGP routing information to manipulate the BGP routing table of $Router_j$.

Attack Procedure. Figure 11 presents the steps to manipulate the BGP routing table of $Router_j$. ❶ The attacker first detects a TCP connection between $Router_i$ and $Router_j$ on port 179 via side channels identified in Linux implementations [3], [85]. Thus, the attacker can infer a BGP connection between $Router_i$ and $Router_j$. ❷ The attacker detects BGP messages that $Router_i$ may send to $Router_j$ by probing periodical advertisements of BGP routers and utilizing prediction based methods [27], [23], [25]. ❸ In this case study, we pick the situation of IP fragmentation occurring at vulnerable Linux hosts, hence the attacker forges an ICMP "Fragmentation Needed" message embedded with an echo reply data to trick $Router_i$ into fragmenting its TCP segments. ❹ The attacker pretends to be $Router_i$ and sends forged fragments to $Router_j$, the forged fragments will be cached. ❺ Original TCP segments from $Router_i$ are fragmented, and the benign fragments are sent to $Router_j$. Finally, all fragments with the same IPID will be mis-reassembled, and a poisoned BGP message is pushed to the BGP process in $Router_j$.

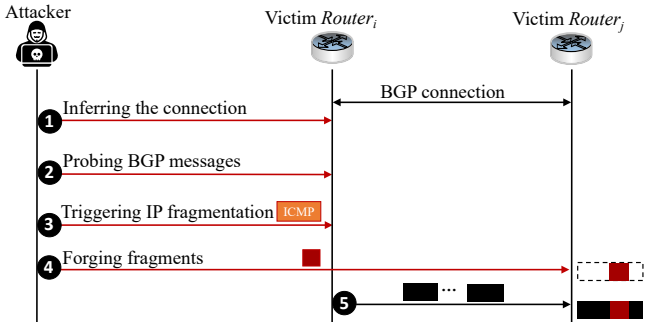


Fig. 11. Manipulating BGP routing table.

Attack Scenario 3: Poisoning BGP Updates. In our experiment, $Router_i$ (whose IP address is 10.1.0.50) advertises

[11] In our investigation, TCP MD5 between two BGP routers is enabled only when the routers connect with each other across layer-two networks.

BGP update messages embedded with NLRI (Network Layer Reachability Information) periodically to $Router_j$ (whose IP address is 10.3.0.50) and other peers. A network of 10.2.2.0/24 that can be reached via $Router_i$ is advertised in the update message. The attacker can replace 10.2.2.0/24 in the message with some other false networks via IP fragmentation, e.g., 12.2.0.0/24, thus poisoning the routing table of $Router_j$. Figure 12 illustrates the fake routing information received by $Router_j$, which is different from the original routing information received by other un-poisoned BGP peers. The value of L_{mtu} in the forged ICMP error message is set to 68.

```
BGP table version is 0, local router ID is 10.4.0.100
Status codes: s suppressed, d damped, h history, * valid, > best, =
              i internal, r RIB-failure, S Stale, R Removed
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      10.1.0.50                0             0 7675 i
*> 10.2.2.0/24      10.1.0.50                0             0 7675 i
*> 10.23.23.0/24    10.1.0.50                0             0 7675 i
*> 10.23.29.0/24    10.1.0.50                0             0 7675 i
*> 10.23.31.0/24    10.1.0.50                0             0 7675 i
```

(a) The original legal routing information

```
BGP table version is 0, local router ID is 10.3.0.50
Status codes: s suppressed, d damped, h history, * valid, > best, =
              i internal, r RIB-failure, S Stale, R Removed
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      10.1.0.50                0             0 7675 i
*> 10.23.23.0/24    10.1.0.50                0             0 7675 i
*> 10.23.29.0/24    10.1.0.50                0             0 7675 i
*> 10.23.31.0/24    10.1.0.50                0             0 7675 i
*> 12.2.0.0/24      10.1.0.50                0             0 7675 i
```

(b) The fake routing information received by $Router_j$

Fig. 12. Manipulating results on BGP routing table.

To accelerate the prediction of IPID generated by $Router_i$, we send 64 fragments to $Router_j$ all at once[12], each with a different IPID value. If one of the malicious fragments matches the original IPID generated from $Router_i$, the fragment will be mis-reassembled with the benign ones and the original BGP message will be poisoned. In this attack, most of the delay is incurred due to the detection of the BGP connection (55.4 seconds on average). Fortunately, BGP always runs over long TCP connections, so an attacker has enough time to analyze the connection. Section VII presents the evaluation results about the attack success rate. Compared with previous BGP hijacking attacks [9], [47], our attack is more stealthy, since the attacker does not need to initiate suspicious false BGP announcements.

VII. Attack Evaluation and Discussion

A. Attack Effectiveness

Table V presents the evaluation results of our attacks on the Internet. First, we evaluate the attacks against 14 public websites, where IP fragmentation occurs at intermediate routers on the path from the victim server to the victim client. In order to evaluate the impact of different path MTUs on the attack success rate, we deploy three types of problematic routers with three next-hop MTU values, i.e., 68 octets, 296 octets and 512 octets, as the gateway of the victim clients in our testbed[13]. These routers will fragment TCP segments from the public servers whose acceptable minimum path MTU is greater than the router's next-hop MTU. We also deploy three types of victim clients with Ubuntu 20.10, Windows 10 Pro, and Mac OS 10.14, respectively, to evaluate the impacts of client OSes.

We show that if the target websites only enable HTTP, the attack achieves high success rates to poison the client's web cache by leveraging IP fragmentation. For example, in the first row of Table V, when the next-hop MTU of the router is 68 octets, 296 octets, and 512 octets, the success rates are $\frac{42}{50}$, $\frac{40}{50}$, and $\frac{41}{50}$, respectively, which are mainly affected by IPID noises at the server sides [65], [24].

Particularly, if the websites are enabled with redirection from HTTP to HTTPS, the attacker can intercept the redirection by leveraging IP fragmentation. Besides being affected by IPID noises at the server side, the success rates of these attacks are affected by the difference between the size of the server's redirection packets and the router's next-hop MTU value. If the size of the redirection packet is less than the router's next-hop MTU value, the packet will not be fragmented, and thus the attacker will not be able to intercept the redirection by leveraging IP fragmentation.

Second, we evaluate the effectiveness of the attacks against vulnerable servers where IP fragmentation occurs. As shown in Table V, we deploy 8 vulnerable Linux servers, i.e., virtual machines (VM), with different kernel versions in clouds of Tencent, Alibaba, and ClientVPS. When TCP segments generated by a Linux server are fragmented to 552 octets due to a forged ICMP "Fragmentation Needed" message, IP fragmentation can be exploited to poison the client's web cache, intercept the redirection from HTTP to HTTPS, or poison BGP messages that the victim client received. The average success rate of the attacks is higher than 78%. Note that since the mis-resembling of the benign fragments with the forged ones can be finished instantly in the kernel of the victim client, it is difficult for users of poisoned clients to perceive our attacks.

B. Discussion

(1) The Impact of Encryption. It is difficult for attackers to craft equivalents with the same checksum to deceive TCP when the payload of the TCP segments is secured by encryption mechanisms (e.g., TLS [22]). However, a considerable portion of traffic on the Internet is not protected by these encryption mechanisms (e.g., 30% of page loads by Firefox do not use TLS [78]), and our evaluation in table V shows that even some financial websites are still not encrypted. Besides, even if encryption mechanisms are enabled to protect the traffic, attackers can still inject malicious fragments into the target traffic, which forces a mis-reassembling between the benign fragments and the malicious ones. Although the malformed segment cannot pass the checksum of TCP, they will interfere with the reception of benign segments, which may lead to DoS attacks. Similarly, it is not easy to poison the emerging

[12]The size of defragmentation cache in recent Linux versions is 64. However, older versions allow thousands of fragments to be cached, so attackers can send more fragments to improve the success rate of guessing IPID.

[13]To avoid ethical issues, we do not attack users attached to the identified problematic routers (e.g., routers with the next-hop MTU of 68 octets in AS4780 and AS3462). Instead, we validate the attacks based on problematic routers in our testbed.

TABLE V. EVALUATION OF IP FRAGMENTATION ATTACKS AGAINST TCP.

Victim server	Organization	Fragmentation MTUs	OS version of victim client	Attack result	Success rate with different MTU
112.65.*.149 (HTTP only)	Agricultural Bank of China	Router 68, 296, 512	Windows 10 Pro	web cache poisoning	$\frac{42}{50}$, $\frac{40}{50}$, $\frac{41}{50}$
122.119.*.139 (HTTP only)	ceair.com	Router 68, 296, 512	Windows 10 Pro	web cache poisoning	$\frac{41}{50}$, $\frac{41}{50}$, $\frac{39}{50}$
118.228.x.78 (HTTP only)	www.ccb.com	Router 68, 296, 512	Mac OS 10.14	web cache poisoning	$\frac{43}{50}$, $\frac{44}{50}$, $\frac{42}{50}$
61.159.x.11 (HTTP only)	www.gov.cn	Router 68, 296, 512	Mac OS 10.14	web cache poisoning	$\frac{44}{50}$, $\frac{39}{50}$, $\frac{41}{50}$
14.139.x.44 (HTTP only)	ignou.ac.in	Router 68, 296, 512	Mac OS 10.14	web cache poisoning	$\frac{42}{50}$, $\frac{40}{50}$, $\frac{39}{50}$
104.18.x.88 (HTTP only)	www.fao.org	Router 68, 296, 512	Ubuntu 20.10	web cache poisoning	$\frac{43}{50}$, $\frac{41}{50}$, $\frac{41}{50}$
195.7.x.12 (HTTP only)	www.finfacts.ie	Router 68, 296, 512	Ubuntu 20.10	web cache poisoning	$\frac{42}{50}$, $\frac{39}{50}$, $\frac{39}{50}$
42.81.*.70 (HTTP only)	china.com.cn	Router 68, 296, 512	Ubuntu 20.10	web cache poisoning	$\frac{44}{50}$, $\frac{43}{50}$, $\frac{40}{50}$
151.101.x.67 (HTTP redirection enabled)	CNN	Router 68, 296, 512	Windows 10 Pro	redirection intercepting	$\frac{44}{50}$, $\frac{41}{50}$, $\frac{42}{50}$
202.165.*.50 (HTTP redirection enabled)	Yahoo	Router 68, 296, 512	Windows 10 Pro	redirection intercepting	$\frac{43}{50}$, $\frac{39}{50}$, $\frac{37}{50}$
162.14.x.217 (HTTP redirection enabled)	SOHU	Router 68, 296, 512	Mac OS 10.14	redirection intercepting	$\frac{42}{50}$, $\frac{40}{50}$, ×
13.224.x.113 (HTTP redirection enabled)	ESPN	Router 68, 296, 512	Mac OS 10.14	redirection intercepting	$\frac{39}{50}$, $\frac{42}{50}$, $\frac{38}{50}$
54.192.x.199 (HTTP redirection enabled)	IMDB	Router 68, 296, 512	Ubuntu 20.10	redirection intercepting	$\frac{42}{50}$, $\frac{37}{50}$, $\frac{37}{50}$
171.161.*.100 (HTTP redirection enabled)	Bank of America	Router 68, 296, 512	Ubuntu 20.10	redirection intercepting	$\frac{44}{50}$, ×, ×
170.106.*.100 (Linux 3.9.10, HTTP only)	VM in Tencent cloud	Server 552	Windows 10 Pro	web cache poisoning	$\frac{41}{50}$
162.62.*.44 (Linux 4.16.10, HTTP only)	VM in Tencent cloud	Server 552	Ubuntu 20.10	web cache poisoning	$\frac{39}{50}$
147.139.*.126 (Linux 4.20.6, HTTP only)	VM in Alibaba cloud	Server 552	Ubuntu 20.10	web cache poisoning	$\frac{42}{50}$
45.147.x.234 (Linux 5.0, HTTP only)	VM in ClientVPS	Server 552	Mac OS 10.14	web cache poisoning	$\frac{42}{50}$
43.129.*.233 (Linux 4.20.15, HTTP redirection enabled)	VM in Tencent cloud	Server 552	Windows 10 Pro	redirection intercepting	$\frac{43}{50}$
8.208.*.114 (Linux 5.1.20, HTTP redirection enabled)	VM in Alibaba cloud	Server 552	Ubuntu 20.10	redirection intercepting	$\frac{39}{50}$
183.173.*.12 (Linux 4.16.10, Quagga 1.2.2)	VM in Alibaba cloud	Server 552	Ubuntu 20.10	poisoning BGP updates	$\frac{41}{50}$
124.156.*.135 (Linux 4.15.10, Quagga 1.2.2)	VM in Tencent cloud	Server 552	Ubuntu 20.10	poisoning BGP updates	$\frac{44}{50}$

Internet transport protocol, i.e., QUIC [39], via IP fragmentation because the QUIC communications are always secured. Nevertheless, it may still suffer from the IP fragmentation based DoS attack once the underlying UDP datagrams are fragmented. Thus, our attack can still interfere with secure connections though the attacker cannot poison the connections.

(2) Comparisons of Two IP Fragmentation Situations.

In this paper, we identify two situations under which IP fragmentation on TCP can be triggered, i.e., IP fragmentation happening at problematic routers or happening at vulnerable Linux hosts. We now compare the two situations with respect to the placement of the vantage points and their impacts. First, the placement of the vantage points used for sending forged ICMP "Fragmentation Needed" messages (that force IP fragmentation at vulnerable hosts) is not limited by locations, since the originator and path of ICMP error messages cannot be verified currently on the Internet [24]. In contrast, exploiting IP fragmentation at problematic routers may be constrained by the locations of the vantage points, since off-path attackers may not be able to identify a problematic router on the path for a given TCP connection. In practice, off-path attackers can probe problematic routers first (e.g., gateways of subnets) and then identify and poison potential victim TCP connections whose traffic may traverse the router. These two situations are not exclusive, and attackers can exploit either of them to attack TCP. Second, when considering the impacts of the two situations, a problematic router may be harmful to all the TCP traffic that traverses the router. No matter where the TCP segments are originated, as long as the originator's acceptable minimum path MTU is greater than the router's next-hop MTU, IP fragmentation will happen at the router. In contrast, the impact of the undesired IP fragmentation at vulnerable Linux hosts is limited to the established TCP connections at the host.

VIII. COUNTERMEASURES

Responsible Disclosure. We report the vulnerability of IP fragmentation in Linux to the Linux community and Bugzilla and submit our PoC to the Linux community. With respect to IP fragmentation occurring at problematic routers, we contact the affected ISPs to disclose the vulnerability. Besides, we report the vulnerability to Microsoft, since the acceptable minimum PMTU of Windows systems (i.e., 596 octets) is greater than the recommendation in RFC 1191 (i.e., 576 octets). We also propose to enhance the PMTUD mechanism to throttle the

identified attacks via addressing the root causes, i.e., avoiding IP fragmentation over TCP happening at intermediate routers or at hosts.

A. Avoiding IP Fragmentation at Routers

The root cause of IP fragmentation at intermediate routers is that the originator's acceptable minimum path MTU (i.e., the system variable of min_pmtu) is greater than the next-hop MTU of the routers. It is difficult to remove or fix all those problematic routers from the Internet in the near future, hence one alternative solution is to change the originator's acceptable minimum PMTU value (i.e., min_pmtu) being less than the intermediate router's next-hop MTU value. As a result, the originator will handle ICMP "Fragmentation Needed" messages as shown in Figure 1(a) to avoid IP fragmentation at intermediate routers.

In Linux, FreeBSD, and Mac systems, users can reset the host's acceptable minimum path MTU value (i.e., min_pmtu) to the minimum of 68 octets using the $sysctl$ command, e.g., resetting the value in Linux systems via executing the command of "*sysctl net.ipv4.min_pmtu = 68*". Since each router on the Internet can forward a packet of 68 octets without IP fragmentation [70], IP fragmentation will be avoided at intermediate routers once the originator's min_pmtu is set to 68 octets. Note it is difficult for users to reset the value of min_pmtu in Windows systems manually, which may require the help of the vendor.

B. Avoiding IP Fragmentation at Hosts

In order to trick vulnerable servers equipped with Linux kernel version 3.8.1 and beyond into fragmenting TCP segments, an attacker impersonates a router and forges an ICMP "Fragmentation Needed" message embedded with an ICMP echo reply to the server. The forged ICMP error message will result in the desynchronization on the path MTU value between the server's TCP layer and the IP layer, thus causing IP to fragment oversized TCP segments. Existing specifications such as RFC 5927 [38] only consider direct ICMP attacks against TCP, which requires that the forged ICMP error message (embedded with a TCP packet) should carry the correct random sequence number. However, the indirect attack cases where non-TCP packets are embedded into ICMP error messages to affect TCP are not considered in any specifications. We can fix the identified vulnerability of the incorrect IP fragmentation on TCP segments (due to a forged ICMP "Fragmentation Needed" message embedded with a non-TCP packet of ICMP echo reply) by using two solutions.

First, when an ICMP "Fragmentation Needed" message embedded with a packet that does not perform PMTUD (e.g., an ICMP echo reply packet) arrives, one straightforward solution for the server is to discard the received ICMP error message directly, since correct packets originated from protocols that do not perform PMTUD (e.g, ICMP echo reply) will not trigger ICMP "Fragmentation Needed" messages at all. Once the size of these packets exceeds the next-hop MTU of intermediate routers, they will be fragmented by the router, instead of reflecting an ICMP "Fragmentation Needed" message. As a result, if such an ICMP error message is detected on the Internet, it must be forged by attackers and should be discarded by the server to prevent IP fragmentation.

Second, it is worth noting that if attackers forge ICMP "Fragmentation Needed" messages embedded with protocol data that performs PMTUD, it is still challenging for the server to judge the legitimacy of the message unless the server records the sent data and then check the embedded data, which is impractical for connectionless protocols. As a result, in order to avoid erroneous discarding of ICMP "Fragmentation Needed" messages and the potential performance issues, as well as avoiding IP fragmentation at vulnerable Linux servers, the fundamental solution is to fix the desynchronization on the path MTU value caused by the ICMP error messages. Once such a message is received and its legitimacy cannot be judged accurately, the IP layer should actively update the MTU information carried in the message to the TCP layer, i.e., actively identifying related sockets at the TCP layer to notify TCP to adjust the MSS immediately. Thus, the TCP layer can achieve a strict synchronization on the path MTU value with IP to avoid writing oversized segments to IP and prevent IP fragmentation.

IX. Related Work

IP Fragmentation. Since the IDS devices and the OSes may reassemble the fragmented IP packets in different ways, it leaves spaces for attackers to evade the security validation by carefully crafting malicious fragments [67], [5]. IP fragmentation has been widely abused to conduct denial of service attacks [54], [61], DNS cache poisoning attacks [41], [11], [42], [86], and traffic interception attacks [29], [31]. Moreover, the complex and inefficient process of IP fragmentation reassembly may have negative impacts on network performance, sometimes even communication failure [49], [37], [60].

Considering that the IP fragmentation technique has been widely exploited, it is a common sense for the security community to regard the IP fragmentation as a vulnerable process. For example, RFC 1858 [87] and RFC 3128 [58] comprehensively presented the threat models inspired by IP fragmentation, e.g., bypassing the packet filter rules of security products. Therefore, it is one best practice to discover PMTU to avoid the IP fragmentation whenever possible [59], [57]. However, in this paper, we demonstrate that the reality is not always as expected, IP fragmentation can still be abused to attack TCP, which is contrary to the common belief [41], [31], [64], [82], [46], [17], [30].

TCP Injection. TCP always randomize the 32-bit sequence and acknowledgment numbers [71], [80], [74], and most implementations also randomize the 16-bit client port number [52]. Therefore, how to infer these randomness and then inject an acceptable TCP segment is the main challenge for previous off-path TCP attacks. By exploiting the global IPID counter previously adopted by Linux and Windows systems, Gilad *et al.* concluded that an off-path attacker can infer if two end hosts have established a TCP connection by a specific four-tuple and then perform an off-path TCP injection attacks [30], [34], [33]. Qian *et al.* discovered that the middlebox of firewalls can be abused to infer the sequence numbers of TCP connections [72] and proposed to conduct a collaborative TCP sequence number inference attack by leveraging unprivileged applications [73]. By exploiting the side channel vulnerability of global rate limit presenting in the challenge `ACK` mechanism, an off-path attacker can infer the TCP sequence and acknowledge

numbers of an established TCP connection, then hijack the connection [12], [13]. TCP timing side channel in the half-duplex IEEE 802.11 networks can also be exploited to inject malicious segments into a TCP connection [16]. Feng *et al.* proposed to build a side channel in the mixed IPID assignment of modern Linux systems to infer the randomness and hijack a victim TCP connection [24]. In contrast, we propose to evade PMTUD and then poison TCP via IP fragmentation, i.e., circumventing the guessing of these randomness. Most of the previous vulnerabilities have been fixed by eliminating the observability or introducing randomness to the shared resources [72], [12], [13], [73], [33], [21], [20], [2]. We report new vulnerabilities arising during the interactions between IP, ICMP and TCP, which can be exploited to poison TCP.

X. Conclusion

In this paper, we reveal that the widely deployed mechanism of PMTUD that protects TCP from IP fragmentation can be evaded, thus enabling fragmentation-based poisoning attacks against TCP. By exploiting IP fragmentation, we show that an off-path attacker can poison TCP traffic without guessing the random sequence numbers and acknowledgment numbers of a victim TCP connection. We detail the two situations of IP fragmentation involved and implement TCP poisoning attacks in different scenarios. Through comprehensive evaluation on the Internet, we demonstrate that our attacks can be performed to cause serious damages in the real world. To mitigate this threat, we propose countermeasures that enhance PMTUD to eliminate the vulnerability of triggering IP fragmentation on TCP segments.

Acknowledgment

We thank the anonymous reviewers for their insightful comments. In particular, we are grateful to our shepherd Syed Rafiul Hussain for his guidance on improving our work. This work was in part supported by China National Funds for Distinguished Young Scientists with No.61825204, National Natural Science Foundation of China with No.61932016 and No.62132011, Beijing Outstanding Young Scientist Program with No.BJJWZYJH012019100030011. Kun Sun's work was in part supported by U.S. ONR grants N00014-16-1-3214 and N00014-18-2893, U.S. ARO grant W911NF-17-1-0447. Ke Xu is the corresponding author.

References

[1] H. B. Acharya and M. G. Gouda, "A theory of network tracing," in *Symposium on Self-Stabilizing Systems*. Springer, 2009, pp. 62–74.

[2] G. Alexander and J. R. Crandall, "Off-path round trip time measurement via tcp/ip side channels," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 1589–1597.

[3] G. Alexander, A. M. Espinoza, and J. R. Crandall, "Detecting tcp/ip connections via ipid hash collisions," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 4, pp. 311–328, 2019.

[4] K. Amit, "Openbsd dns cache poisoning and multiple o/s predictable ip id vulnerability," https://seclists.org/bugtraq/2008/Feb/49, Accessed April 2021.

[5] A. Atlasis, "Attacking ipv6 implementation using fragmentation," *Black-hat europe*, pp. 14–16, 2012.

[6] M. Baggett, "Ip fragment reassembly with scapy," *SANS Institute InfoSec Reading Room*, 2012.

[7] F. Baker, "Requirements for ip version 4 routers," Internet Requests for Comments, Internet Engineering Task Force, RFC 1812, June 1995. [Online]. Available: http://www.rfc-editor.org/rfc/rfc1812.txt

[8] G. Barish and K. Obraczke, "World wide web caching: Trends and techniques," *IEEE Communications magazine*, vol. 38, no. 5, pp. 178–184, 2000.

[9] H. Birge-Lee, Y. Sun, A. Edmundson, J. Rexford, and P. Mittal, "Bamboozling certificate authorities with bgp," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 833–849.

[10] Bootlin, "Ipid assignment in linux kernel," https://elixir.bootlin.com/linux/v4.18/source/include/net/ip.h, Accessed April 2021.

[11] M. Brandt, T. Dai, A. Klein, H. Shulman, and M. Waidner, "Domain validation++ for mitm-resilient pki," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2018, pp. 2060–2076.

[12] Y. Cao, Z. Qian, Z. Wang, T. Dao, S. V. Krishnamurthy, and L. M. Marvel, "Off-path tcp exploits: Global rate limit considered dangerous," in *25th USENIX Security Symposium (USENIX Security 16)*, 2016, pp. 209–225.

[13] ——, "Off-path tcp exploits of the challenge ack global rate limit," *IEEE/ACM Transactions on Networking*, vol. 26, no. 2, pp. 765–778, 2018.

[14] Y. Cao, Z. Wang, Z. Qian, C. Song, S. V. Krishnamurthy, and P. Yu, "Principled unearthing of tcp side channel vulnerabilities," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 211–224.

[15] S. Chen, R. Wang, X. Wang, and K. Zhang, "Side-channel leaks in web applications: A reality today, a challenge tomorrow," in *2010 IEEE Symposium on Security and Privacy*. IEEE, 2010, pp. 191–206.

[16] W. Chen and Z. Qian, "Off-path tcp exploit: How wireless routers can jeopardize your secrets," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 1581–1598.

[17] CISCO, "Resolve ipv4 fragmentation, mtu, mss, and pmtud issues with gre and ipsec," https://www.cisco.com/c/en/us/support/docs/ip/generic-routing-encapsulation-gre/25885-pmtud-ipfrag.html, Accessed April 2021.

[18] D. Clark, "Ip datagram reassembly algorithms," Internet Requests for Comments, Internet Engineering Task Force, RFC 815, July 1982. [Online]. Available: http://www.rfc-editor.org/rfc/rfc815.txt

[19] J. Durand, I. Pepelnjak, and G. Doering, "Bgp operations and security," Internet Requests for Comments, Internet Engineering Task Force, RFC 7454, February 2015. [Online]. Available: http://www.rfc-editor.org/rfc/rfc7454.txt

[20] R. Ensafi, J. Knockel, G. Alexander, and J. R. Crandall, "Detecting intentional packet drops on the internet via tcp/ip side channels," in *International Conference on Passive and Active Network Measurement*. Springer, 2014, pp. 109–118.

[21] R. Ensafi, J. C. Park, D. Kapur, and J. R. Crandall, "Idle port scanning and non-interference analysis of network protocol stacks using model checking," in *USENIX Security Symposium (USENIX Security 10)*, 2010, pp. 257–272.

[22] R. Eric, "The transport layer security (tls) protocol version 1.3," Internet Requests for Comments, Internet Engineering Task Force, RFC 8446, August 2018. [Online]. Available: http://www.rfc-editor.org/rfc/rfc8446.txt

[23] N. Feamster and J. Rexford, "Network-wide prediction of bgp routes," *IEEE/ACM Transactions on Networking (TON)*, vol. 15, no. 2, pp. 253–266, 2007.

[24] X. Feng, C. Fu, Q. Li, K. Sun, and K. Xu, "Off-path tcp exploits of the mixed ipid assignment," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, p. 1323–1335.

[25] A. Flavel, J. McMahon, A. Shaikh, M. Roughan, and N. Bean, "Bgp route prediction within isps," *Computer Communications*, vol. 33, no. 10, pp. 1180–1190, 2010.

[26] M. Fraiwan, R. Al-Salman, N. Khasawneh, and S. Conrad, "Analysis and identification of malicious javascript code," *Information Security Journal: A Global Perspective*, vol. 21, no. 1, pp. 1–11, 2012.

[27] H. Geoff, R. Mattia, and A. Grenville, "Securing bgp-a literature

survey," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 2, pp. 199–222, 2011.

[28] Y. Gilad, A. Cohen, A. Herzberg, M. Schapira, and H. Shulman, "Are we there yet? on rpki's deployment and security." in *NDSS*, 2017.

[29] Y. Gilad and A. Herzberg, "Fragmentation considered vulnerable: Blindly intercepting and discarding fragments," in *Proceedings of the 5th USENIX conference on Offensive technologies*. USENIX Association, 2011, pp. 2–2.

[30] ——, "Off-path attacking the web," in *Proceedings of the 6th USENIX Workshop on Offensive Technologies*, 2012, pp. 41–52.

[31] ——, "Fragmentation considered vulnerable," *ACM Transactions on Information and System Security (TISSEC)*, vol. 15, no. 4, p. 16, 2013.

[32] ——, "When tolerance causes weakness: The case of injection-friendly browsers," in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 435–446.

[33] ——, "Off-path tcp injection attacks," *ACM Transactions on Information and System Security (TISSEC)*, vol. 16, no. 4, p. 13, 2014.

[34] Y. Gilad, A. Herzberg, and H. Shulman, "Off-path hacking: The illusion of challenge-response authentication," *IEEE Security & Privacy*, vol. 12, no. 5, pp. 68–77, 2013.

[35] V. Gill, J. Heasley, D. Meyer, P. Savola, and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)," Internet Requests for Comments, Internet Engineering Task Force, RFC 5082, October 2007. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5082.txt

[36] E. Gillet, "The deployment of the dnssec protocol," https://www.lexology.com/library/detail.aspx?g=65d6625e-464a-4409-aa7e-a2682ddc7308, 2020.

[37] M. Göhring, H. Shulman, and M. Waidner, "Path mtu discovery considered harmful," in *2018 IEEE 38th International Conference on Distributed Computing Systems*. IEEE, 2018, pp. 866–874.

[38] F. Gont, "ICMP Attacks against TCP," Internet Requests for Comments, Internet Engineering Task Force, RFC 5927, July 2010. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5927.txt

[39] I. Q. W. Group, "Quick udp internet connections," https://github.com/quicwg, Accessed April 2021.

[40] A. Heffernan, "Protection of bgp sessions via the tcp md5 signature option," Internet Requests for Comments, Internet Engineering Task Force, RFC 2385, August 1998. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2385.txt

[41] A. Herzberg and H. Shulman, "Fragmentation considered poisonous, or: One-domain-to-rule-them-all. org," in *2013 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2013, pp. 224–232.

[42] ——, "Towards adoption of dnssec: Availability and security challenges," *IACR Cryptology ePrint Archive*, vol. 2013, p. 254, 2013.

[43] ——, "Vulnerable delegation of dns resolution," in *European Symposium on Research in Computer Security*. Springer, 2013, pp. 219–236.

[44] H. Hu and G. Wang, "End-to-end measurements of email spoofing attacks," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 1095–1112.

[45] K. Ishiguro, "Quagga routing suite," http://quagga.sourceforge.net/commercial.php/, Accessed April 2021.

[46] S. Iveson, "Ip fragmentation in detail," https://packetpushers.net/ip-fragmentation-in-detail/, Accessed April 2021.

[47] Q. Jacquemart, "Towards uncovering bgp hijacking attacks," Ph.D. dissertation, Télécom ParisTech, 2015.

[48] K. Jeffrey and C. Jedidiah, R, "Counting packets sent between arbitrary internet hosts," in *Proceedings of the 4th USENIX Workshop on Free and Open Communications on the Internet (FOCI'14)*. USENIX Association, 2014.

[49] C. A. Kent and J. C. Mogul, "Fragmentation considered harmful," in *Proceedings of ACM SIGCOMM*. ACM, 1987, pp. 390–401.

[50] A. Klein and B. Pinkas, "From ip id to device id and kaslr bypass," in *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 1063–1080.

[51] M. Kühlewind, M. Walter, I. R. Learmonth, and B. Trammell, "Tracing internet path transparency," in *2018 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 2018, pp. 1–7.

[52] M. Larsen and F. Gont, "Recommendations for transport-protocol port randomization," Internet Requests for Comments, Internet Engineering Task Force, RFC 6056, January 2011. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6056.txt

[53] M. Luckie, R. Beverly, R. Koga, K. Keys, J. A. Kroll, and k. claffy, "Network hygiene, incentives, and regulation: Deployment of source address validation in the internet," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 465–480.

[54] K. Malachi, "Ping of death," https://insecure.org/sploits/ping-o-death.html, Accessed April 2021.

[55] K. Man, Z. Qian, Z. Wang, X. Zheng, Y. Huang, and H. Duan, "Dns cache poisoning attack reloaded: Revolutions with side channels," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 1337–1350.

[56] L. Mastilak, M. Galinski, P. Helebrandt, I. Kotuliak, and M. Ries, "Enhancing border gateway protocol security using public blockchain," *Sensors*, vol. 20, no. 16, p. 4482, 2020.

[57] J. McCann, S. Deering, and J. Mogul, "Path mtu discovery for ip version 6," Internet Requests for Comments, Internet Engineering Task Force, RFC 1981, August 1996. [Online]. Available: http://www.rfc-editor.org/rfc/rfc1981.txt

[58] I. Miller, "Protection against a variant of the tiny fragment attack," Internet Requests for Comments, Internet Engineering Task Force, RFC 3128, June 2001. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3128.txt

[59] J. Mogul and S. Deering, "Path mtu discovery," Internet Requests for Comments, Internet Engineering Task Force, RFC 1191, November 1990. [Online]. Available: http://www.rfc-editor.org/rfc/rfc1191.txt

[60] D. Murray, T. Koziniec, K. Lee, and M. Dixon, "Large mtus and internet performance," in *2012 IEEE 13th International Conference on High Performance Switching and Routing*. IEEE, 2012, pp. 82–87.

[61] NIST, "Cve-2018-5391," https://nvd.nist.gov/vuln/detail/CVE-2018-5391, Accessed April 2021.

[62] Nmap, "The network mapper," https://nmap.org/, Accessed April 2021.

[63] J. Novak, "Target-based fragmentation reassembly," *Sourcefire, Columbia, MD*, 2005.

[64] L. Parziale, W. Liu, C. Matthews, N. Rosselot, C. Davis, J. Forrester, D. T. Britt *et al.*, *TCP/IP tutorial and technical overview*. IBM Redbooks, 2006.

[65] P. Pearce, R. Ensafi, F. Li, N. Feamster, and V. Paxson, "Augur: Internet-wide detection of connectivity disruptions," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 427–443.

[66] ——, "Toward continual measurement of global network-level censorship," *IEEE Security & Privacy*, vol. 16, no. 1, pp. 24–33, 2018.

[67] G. Pierce, "Intrusion detection evasion techniques and case studies," *SANS Institute InfoSec Reading Room*, 2017.

[68] J. Postel, "User datagram protocol," Internet Requests for Comments, Internet Engineering Task Force, RFC 768, August 1980. [Online]. Available: http://www.rfc-editor.org/rfc/rfc768.txt

[69] ——, "Internet control message protocol," Internet Requests for Comments, Internet Engineering Task Force, RFC 792, September 1981. [Online]. Available: http://www.rfc-editor.org/rfc/rfc792.txt

[70] ——, "Internet protocol," Internet Requests for Comments, Internet Engineering Task Force, RFC 791, September 1981. [Online]. Available: http://www.rfc-editor.org/rfc/rfc791.txt

[71] ——, "Transmission control protocol," Internet Requests for Comments, Internet Engineering Task Force, RFC 793, September 1981. [Online]. Available: http://www.rfc-editor.org/rfc/rfc793.txt

[72] Z. Qian and Z. M. Mao, "Off-path tcp sequence number inference attack how firewall middleboxes reduce security," in *2012 IEEE Symposium on Security and Privacy*. IEEE, 2012, pp. 347–361.

[73] Z. Qian, Z. M. Mao, and Y. Xie, "Collaborative tcp sequence number inference attack: How to crack sequence number under a second," in *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 2012, pp. 593–604.

[74] A. Ramaiah, R. Stewart, and M. Dalal, "Improving tcp's robustness to blind in-window attacks," Internet Requests for Comments, Internet

Engineering Task Force, RFC 5961, August 2010. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5961.txt

[75] F. Salutari, D. Cicalese, and D. J. Rossi, "A closer look at ip-id behavior in the wild," in *International Conference on Passive and Active Network Measurement*. Springer, 2018, pp. 243–254.

[76] C. Schramm, "Why does linux enforce a minimum mtu of 552?" http://cschramm.blogspot.com/2012/12/why-does-linux-enforce-minimum-mtu-of.html, 2012.

[77] J. Schwenk, M. Niemietz, and C. Mainka, "Same-origin policy: Evaluation in modern browsers," in *26th USENIX Security Symposium (USENIX Security 17)*, 2017, pp. 713–727.

[78] F. Sergey and W. Eric, "The use of tls in censorship circumvention," in *Network and Distributed Systems Security (NDSS) Symposium*, 2019, pp. 1–15.

[79] U. Shankar and V. Paxson, "Active mapping: Resisting nids evasion without altering traffic," in *2003 Symposium on Security and Privacy*. IEEE, 2003, pp. 44–61.

[80] J. Touch, "Defending tcp against spoofing attacks," Internet Requests for Comments, Internet Engineering Task Force, RFC 4953, July 2007. [Online]. Available: http://www.rfc-editor.org/rfc/rfc4953.txt

[81] ——, "Updated specification of the ipv4 id field," Internet Requests for Comments, Internet Engineering Task Force, RFC 6864, February 2013. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6864.txt

[82] G. Van Den Broek, R. van Rijswijk-Deij, A. Sperotto, and A. Pras, "Dnssec meets real world: dealing with unreachability caused by fragmentation," *IEEE communications magazine*, vol. 52, no. 4, pp. 154–160, 2014.

[83] Y. Wang, W.-d. Cai, and P.-c. Wei, "A deep learning approach for detecting malicious javascript code," *security and communication networks*, vol. 9, no. 11, pp. 1520–1534, 2016.

[84] M. Zalewski, "A new tcp/ip blind data injection technique?" https://seclists.org/bugtraq/2003/Dec/161, 2003.

[85] X. Zhang, J. Knockel, and J. R. Crandall, "Onis: Inferring tcp/ip-based trust relationships completely off-path," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 2069–2077.

[86] X. Zheng, C. Lu, J. Peng, Q. Yang, D. Zhou, B. Liu, K. Man, S. Hao, H. Duan, and Z. Qian, "Poison over troubled forwarders: A cache poisoning attack targeting dns forwarding devices," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 577–593.

[87] G. Ziemba, D. Reed, and P. Traina, "Security considerations for ip fragment filtering," Internet Requests for Comments, Internet Engineering Task Force, RFC 1858, October 1995. [Online]. Available: http://www.rfc-editor.org/rfc/rfc1858.txt

[88] ZMap, "The internet scanner," https://zmap.io/, Accessed April 2021.

APPENDIX

A. Snapshots of web cache poisoning

Figure 13 shows the experimental result of web cache poisoning against MIT, where the original correct value of 1861 and 1464 are replaced by 1960 and 5064, respectively. Figure 14 shows the experimental result against Bank of China (BOC). The original exchange rate of 195.48, 454.99 and 743.96 are stealthily replaced by 991.08, 167.98 and 228.98, respectively (after our disclosure, the website of BOC is now protected by TLS).

B. Measurement of IPID

To ascertain the distributions of IPID assignment on the Internet, we conduct a measurement study based on the network scanner ZMap [88]. As shown in Figure 15, we detect more than 270 million hosts in 15 days and observe that about 68% hosts linearly increase their IPID, i.e., selecting the algorithms of global counter-based IPID assignment or per-destination IPID assignment, whose IPID is predictable. About 17% hosts

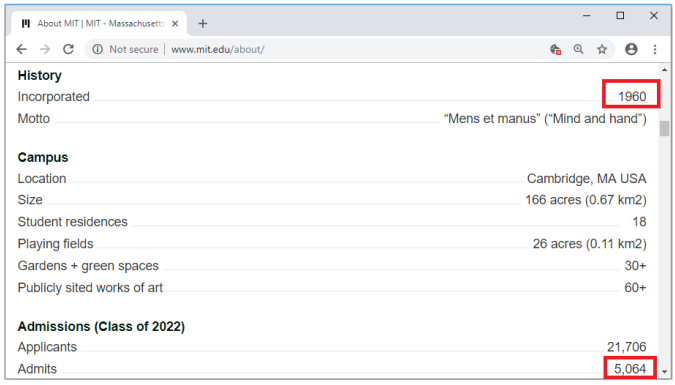


Fig. 13. Fake college news cached in the victim client.

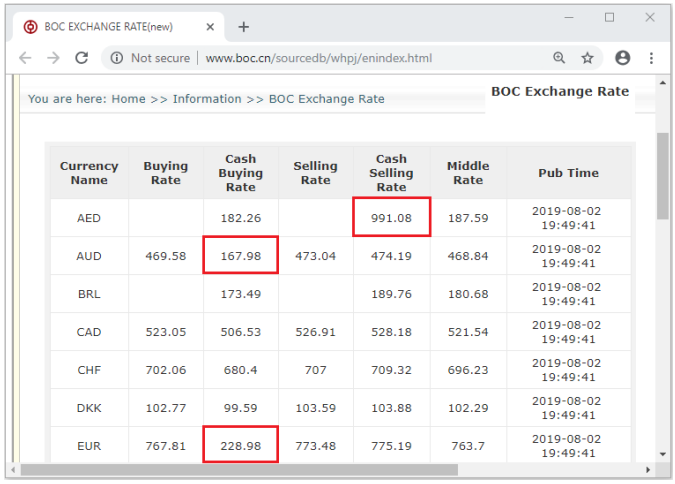


Fig. 14. Fake financial news cached in the victim client.

select a constant as the packets ID because the IPID algorithm in these hosts enables RFC 6864 [81]. In this case, when no IP fragmentation occurs, IPID is purposeless and set to a constant (e.g., 0). Instead, if IP fragmentation occurs, the IPID is usually assigned randomly. About 9% hosts in our measurement assign IPID using the random IPID assignment or hash-based IPID assignment algorithms. We cannot determine the algorithms of the rest 6% hosts due to the error responses to our requests. Based on our measurement results, we conclude that IPID is still predictable on a large number of hosts on the Internet.

C. Deceiving TCP Checksum

The checksum mechanism of TCP works based on the one's complement sum of all 16-bit words in the header and payload, where a 12-octet TCP pseudo header is also covered. By simply adjusting two octets, attackers can easily craft equivalents that have the same complement sum with the original segment, thus circumventing the TCP checksum mechanism.

We assume that the string of "Bob loves HanMeiMei for 5555 years" is the original payload in a TCP segment, the one's complement sum of the string is 38135. Table VI illustrates 10 alternative strings that have the same checksum as the original string. Attackers can construct a dictionary in advance

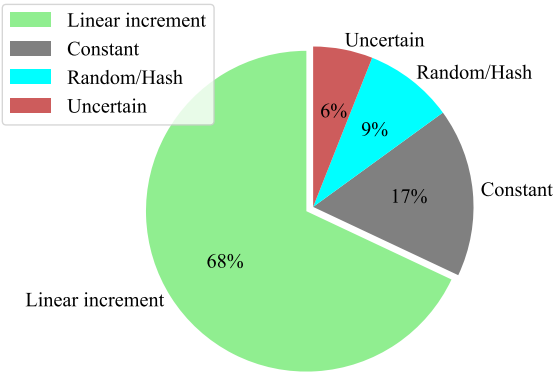


Fig. 15. Distribution of IPID on the Internet.

to preserve the alternative strings of sensitive objects and then query the dictionary to avoid expensive real-time computing.

TABLE VI. EQUIVALENT STRINGS WITH THE SAME CHECKSUM.

No.	Equivalent string	Checksum
	Bob loves HanMeiMei for 5555 years	38135
1	Bob loves HanMeiMei for 1595 years	38135
2	Bob loves HanMeiMei for 2585 years	
3	Bob loves HanMeiMei for 5258 years	
4	Bob loves HanMeiMei for 5357 years	
5	Bob hates HenrraFRi for 3674 years	
6	Bob hates HenrraFRi for 2585 years	
7	Malee misses BeGyAi for 4268 years	
8	Malee misses BeGyAi for 6149 years	
9	H-r-y lives 1#1-xGi for 7238 years	
10	H-r-y lives 1#1-xGi for 8327 years	

D. Fragment Overlapping Policies

Existing reassembly algorithms do not discuss how to deal with fragments overlapping and the retransmission of fragments [70], [18]. Particularly, it is ambiguous on whether the client should favor overlapping fragments with the lowest offset or the highest offset. Moreover, it is not clearly defined on if the first "retransmitted" fragment, the second "retransmitted" fragment, or the last one should be favored by the destination. As a result, various fragment reassembly implementations handle the problem of overlapping fragments differently. In total, there are seven fragment overlapping policies used in different OS releases, i.e., *First* and its four variants including *Linux*, *Windows*, *Solaris* and *BSD*, *Last* and its variant *BSD-right* [5], [79], [63], [6].

***First* and the Four Variants**. *First* favors the original fragment with a given offset. HP-UX and Mac OS adopt this policy. *Linux* favors an original fragment with an offset that is less than a subsequent fragment. Besides Linux, OpenBSD also adopts this policy. *Windows* favors the original fragment except if the offset of a subsequent fragment begins before the original fragment and ends after the original fragment. In this case, it favors the subsequent fragment. *Solaris* favors the original fragment except if the offset of a subsequent fragment begins before the original fragment and ends at an offset equal to or greater than the original fragment. In this case, it favors the subsequent fragment. *BSD* favors an original fragment with an offset that is less than or equal to a subsequent fragment. AIX and FreeBSD adopt this policy.

***Last* and the Variant**. *Last* favors the subsequent fragment with a given offset. Cisco adopts this policy. *BSD-right* favors a subsequent fragment when the original fragment has an offset that is less than or equal to the subsequent one except when the original fragment ends at the same or greater offset than the subsequent fragment. In this case, *BSD-right* favors the original fragment. HP JetDirect printers adopt this policy.