Network Working Group
Request for Comments: 4963
Category: Informational

J. Heffner
M. Mathis
B. Chandler
PSC

July 2007

IPv4 Reassembly Errors at High Data Rates

Status of This Memo

Copyright Notice

Abstract

IPv4 fragmentation is not sufficiently robust for use under some conditions in today's Internet. At high data rates, the 16-bit IP identification field is not large enough to prevent frequent incorrectly assembled IP fragments, and the TCP and UDP checksums are insufficient to prevent the resulting corrupted datagrams from being delivered to higher protocol layers. This note describes some easily reproduced experiments demonstrating the problem, and discusses some of the operational implications of these observations.

1. Introduction

The IPv4 header was designed at a time when data rates were several orders of magnitude lower than those achievable today. This document describes a consequent scale-related failure in the IP identification (ID) field, where fragments may be incorrectly assembled at a rate high enough that it is likely to invalidate assumptions about data integrity failure rates.

That IP fragmentation results in inefficient use of the network has been well documented [Kent87]. This note presents a different kind of problem, which can result not only in significant performance degradation, but also frequent data corruption. This is especially pertinent due to the recent proliferation of UDP bulk transport tools that sometimes fragment every datagram.

Additionally, there is some network equipment that ignores the Don't Fragment (DF) bit in the IP header to work around MTU discovery problems [RFC2923]. This equipment indirectly exposes properly implemented protocols and applications to corrupt data.

2. Wrapping the IP ID Field

The Internet Protocol standard [RFC0791] specifies:

> "The choice of the Identifier for a datagram is based on the need to provide a way to uniquely identify the fragments of a particular datagram. The protocol module assembling fragments judges fragments to belong to the same datagram if they have the same source, destination, protocol, and Identifier. Thus, the sender must choose the Identifier to be unique for this source, destination pair and protocol for the time the datagram (or any fragment of it) could be alive in the Internet."

Strict conformance to this standard limits transmissions in one direction between any address pair to no more than 65536 packets per protocol (e.g., TCP, UDP, or ICMP) per maximum packet lifetime.

Clearly, not all hosts follow this standard because it implies an unreasonably low maximum data rate. For example, a host sending 1500-byte packets with a 30-second maximum packet lifetime could send at only about 26 Mbps before exceeding 65535 packets per packet lifetime. Or, filling a 1 Gbps interface with 1500-byte packets requires sending 65536 packets in less than 1 second, an unreasonably short maximum packet lifetime, being less than the round-trip time on some paths. This requirement is widely ignored.

Additionally, it is worth noting that reusing values in the IP ID field once per 65536 datagrams is the best case. Some implementations randomize the IP ID to prevent leaking information out of the kernel [Bellovin02], which causes reuse of the IP ID field to occur probabilistically at all sending rates.

IP receivers store fragments in a reassembly buffer until all fragments in a datagram arrive, or until the reassembly timeout expires (15 seconds is suggested in [RFC0791]). Fragments in a datagram are associated with each other by their protocol number, the value in their ID field, and by the source/destination address pair. If a sender wraps the ID field in less than the reassembly timeout, it becomes possible for fragments from different datagrams to be incorrectly spliced together ("mis-associated"), and delivered to the upper layer protocol.

A case of particular concern is when mis-association is self-propagating. This occurs, for example, when there is reliable ordering of packets and the first fragment of a datagram is lost in the network. The rest of the fragments are stored in the fragment reassembly buffer, and when the sender wraps the ID field, the first fragment of the new datagram will be mis-associated with the rest of the old datagram. The new datagram will be now be incomplete (since it is missing its first fragment), so the rest of it will be saved in the fragment reassembly buffer, forming a cycle that repeats every 65536 datagrams. It is possible to have a number of simultaneous cycles, bounded by the size of the fragment reassembly buffer.

IPv6 is considerably less vulnerable to this type of problem, since its fragment header contains a 32-bit identification field [RFC2460]. Mis-association will only be a problem at packet rates 65536 times higher than for IPv4.

3. Effects of Mis-Associated Fragments

When the mis-associated fragments are delivered, transport-layer checksumming should detect these datagrams as incorrect and discard them. When the datagrams are discarded, it could create a performance problem for loss-feedback congestion control algorithms, particularly when a large congestion window is required, since it will introduce a certain amount of non-congestive loss.

Transport checksums, however, may not be designed to handle such high error rates. The TCP/UDP checksum is only 16 bits in length. If these checksums follow a uniform random distribution, we expect mis-associated datagrams to be accepted by the checksum at a rate of one per 65536. With only one mis-association cycle, we expect corrupt data delivered to the application layer once per 2^32 datagrams.

This number can be significantly higher with multiple concurrent cycles.

With non-random data, the TCP/UDP checksum may be even weaker still. It is possible to construct datasets where mis-associated fragments will always have the same checksum. Such a case may be considered unlikely, but is worth considering. "Real" data may be more likely than random data to cause checksum hot spots and increase the probability of false checksum match [Stone98]. Also, some applications or higher-level protocols may turn off checksumming to increase speed, though this practice has been found to be dangerous for other reasons when data reliability is important [Stone00].

4. Experimental Observations

To test the practical impact of fragmentation on UDP, we ran a series of experiments using a UDP bulk data transport protocol that was designed to be used as an alternative to TCP for transporting large data sets over specialized networks. The tool, Reliable Blast UDP (RBUDP), part of the QUANTA networking toolkit [QUANTA], was selected because it has a clean interface which facilitated automated experiments. The decision to use RBUDP had little to do with the details of the transport protocol itself. Any UDP transport protocol that does not have additional means to detect corruption, and that could be configured to use IP fragmentation, would have the same results.

In order to diagnose corruption on files transferred with the UDP bulk transfer tool, we used a file format that included embedded sequence numbers and MD5 checksums in each fragment of each datagram. Thus, it was possible to distinguish random corruption from that caused by mis-associated fragments. We used two different types of files. One was constructed so that all the UDP checksums were constant -- we will call this the "constant" dataset. The other was constructed so that UDP checksums were uniformly random -- the "random" dataset. All tests were done using 400 MB files, sent in 1524-byte datagrams so that they were fragmented on standard Fast Ethernet with a 1500-byte MTU.

The UDP bulk file transport tool was used to send the datasets between a pair of hosts at slightly less than the available data rate (100 Mbps). Near the beginning of each flow, a brief secondary flow was started to induce packet loss in the primary flow. Throughout the life of the primary flow, we typically observed mis-association rates on the order of a few hundredths of a percent.

Tests run with the "constant" dataset resulted in corruption on all mis-associated fragments, that is, corruption on the order of a few hundredths of a percent. In sending approximately 10 TB of "random" datasets, we observed 8847668 UDP checksum errors and 121 corruptions of the data due to mis-associated fragments.

5. Preventing Mis-Association

The most straightforward way to avoid mis-association is to avoid fragmentation altogether by implementing Path MTU Discovery [RFC1191] [RFC4821]. However, this is not always feasible for all applications. Further, as a work-around for MTU discovery problems [RFC2923], some TCP implementations and communications gear provide mechanisms to disable path MTU discovery by clearing or ignoring the DF bit. Doing so will expose all protocols using IPv4, even those that participate in MTU discovery, to mis-association errors.

If IP fragmentation is in use, it may be possible to reduce the timeout sufficiently so that mis-association will not occur. However, there are a number of difficulties with such an approach. Since the sender controls the rate of packets sent and the selection of IP ID, while the receiver controls the reassembly timeout, there would need to be some mutual assurance between each party as to participation in the scheme. Further, it is not generally possible to set the timeout low enough so that a fast sender’s fragments will not be mis-associated, yet high enough so that a slow sender’s fragments will not be unconditionally discarded before it is possible to reassemble them. Therefore, the timeout and IP ID selection would need to be done on a per-peer basis. Also, it is likely NAT will break any per-peer tables keyed by IP address. It is not within the scope of this document to recommend solutions to these problems, though we believe a per-peer adaptive timeout is likely to prevent mis-association under circumstances where it would most commonly occur.

A case particularly worth noting is that of tunnels encapsulating payload in IPv4. To deal with difficulties in MTU Discovery [RFC4459], tunnels may rely on fragmentation between the two endpoints, even if the payload is marked with a DF bit [RFC4301]. In such a mode, the two tunnel endpoints behave as IP end hosts, with all tunneled traffic having the same protocol type. Thus, the aggregate rate of tunneled packets may not exceed 65536 per maximum packet lifetime, or tunneled data becomes exposed to possible mis-association. Even protocols doing MTU discovery such as TCP will be affected. Operators of tunnels should ensure that the receiving end’s reassembly timeout is short enough that mis-association cannot occur given the tunnel’s maximum rate.

6. Mitigating Mis-Association

It is difficult to concisely describe all possible situations under which fragments might be mis-associated. Even if an end host carefully follows the specification, ensuring unique IP IDs, the presence of NATs or tunnels may expose applications to IP ID space conflicts. Further, devices in the network that the end hosts cannot see or control, such as tunnels, may cause mis-association. Even a fragmenting application that sends at a low rate might possibly be exposed when running simultaneously with a non-fragmenting application that sends at a high rate. As described above, the receiver might implement to reduce or eliminate the possibility of conflict, but there is no mechanism in place for a sender to know what the receiver is doing in this respect. As a consequence, there is no general mechanism for an application that is using IPv4 fragmentation to know if it is deterministically or statistically protected from mis-associated fragments.

Under circumstances when it is impossible or impractical to prevent mis-association, its effects may be mitigated by use of stronger integrity checking at any layer above IP. This is a natural side effect of using cryptographic authentication. For example, IPsec AH [RFC4302] will discard any corrupted datagrams, preventing their deliver to upper layers. A stronger transport layer checksum such as SCTP's, which is 32 bits in length [RFC2960], may help significantly. At the application layer, SSH message authentication codes [RFC4251] will prevent delivery of corrupted data, though since the TCP connection underneath is not protected, it is considered invalid and the session is immediately terminated. While stronger integrity checking may prevent data corruption, it will not prevent the potential performance impact described above of non-congestive loss on congestion control at high congestion windows.

It should also be noted that mis-association is not the only possible source of data corruption above the network layer [Stone00]. Most applications for which data integrity is critically important should implement strong integrity checking regardless of exposure to mis-association.

In general, applications that rely on IPv4 fragmentation should be written with these issues in mind, as well as those issues documented in [Kent87]. Applications that rely on IPv4 fragmentation while sending at high speeds (the order of 100 Mbps or higher) and devices that deliberately introduce fragmentation to otherwise unfragmented traffic (e.g., tunnels) should be particularly cautious, and introduce strong mechanisms to ensure data integrity.

7. Security Considerations

If a malicious entity knows that a pair of hosts are communicating using a fragmented stream, it may be presented with an opportunity to corrupt the flow. By sending "high" fragments (those with offset greater than zero) with a forged source address, the attacker can deliberately cause corruption as described above. Exploiting this vulnerability requires only knowledge of the source and destination addresses of the flow, its protocol number, and fragment boundaries. It does not require knowledge of port or sequence numbers.

If the attacker has visibility of packets on the path, the attack profile is similar to injecting full segments. Using this attack makes blind disruptions easier and might possibly be used to cause degradation of service. We believe only streams using IPv4 fragmentation are likely vulnerable. Because of the nature of the problems outlined in this document, the use of IPv4 fragmentation for critical applications may not be advisable, regardless of security concerns.

8. Informative References

[Kent87] Kent, C. and J. Mogul, "Fragmentation considered harmful", Proc. SIGCOMM ’87 vol. 17, No. 5, October 1987.

[RFC2923] Lahey, K., "TCP Problems with Path MTU Discovery", RFC 2923, September 2000.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

[Stone98] Stone, J., Greenwald, M., Partridge, C., and J. Hughes, "Performance of Checksums and CRC’s over Real Data", IEEE/ ACM Transactions on Networking vol. 6, No. 5, October 1998.

[Stone00] Stone, J. and C. Partridge, "When The CRC and TCP Checksum Disagree", Proc. SIGCOMM 2000 vol. 30, No. 4, October 2000.

[QUANTA] He, E., Alimohideen, J., Eliason, J., Krishnaprasad, N., Leigh, J., Yu, O., and T. DeFanti, "Quanta: a toolkit for high performance data delivery over photonic networks", Future Generation Computer Systems Vol. 19, No. 6, August 2003.

[Bellovin02] Bellovin, S., "A Technique for Counting NATted Hosts", Internet Measurement Conference, Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement, November 2002.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC2960] Stewart, R., Xie, Q., Morneault, K., Sharp, C., Schwarzbauer, H., Taylor, T., Rytina, I., Kalla, M., Zhang, L., and V. Paxson, "Stream Control Transmission Protocol", RFC 2960, October 2000.

[RFC4251] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH) Protocol Architecture", RFC 4251, January 2006.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

[RFC4459] Savola, P., "MTU and Fragmentation Issues with In-the-Network Tunneling", RFC 4459, April 2006.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

Appendix A. Acknowledgements

This work was supported by the National Science Foundation under Grant No. 0083285.

Authors' Addresses

John W. Heffner
Pittsburgh Supercomputing Center
4400 Fifth Avenue
Pittsburgh, PA 15213
US

Phone: 412-268-2329
EMail: jheffner@psc.edu

Matt Mathis
Pittsburgh Supercomputing Center
4400 Fifth Avenue
Pittsburgh, PA 15213
US

Phone: 412-268-3319
EMail: mathis@psc.edu

Ben Chandler
Pittsburgh Supercomputing Center
4400 Fifth Avenue
Pittsburgh, PA 15213
US

Phone: 412-268-9783
EMail: bchandle@gmail.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.