So You Think You Understand IP Fragmentation

Introducing fragquiz

Outline

People think they understand IP fragmentation... but they don't

How to teach people something they think they already know?

Introducing **fragquiz**: a game to test your IP fragmentation knowledge

Implementation challenges and solutions

Let's play **fragquiz** together!

Bonus: a PLPMTUD implementation with the lowest latency Path MTU search algorithm possible

val@valerieaurora.org

A little about me: Valerie Aurora

Freelance software consultant

Specializing in operating systems, networking, and file systems

- many network drivers
- TCP/IP
- ZFS
- ext4
- union mounts
- VPN

Valerie Aurora

val@valerieaurora.org

So I was implementing PLPMTUD the other day...

Packetization Layer Path MTU Discovery: find the largest packet that can be sent to a host without IP fragmentation, at a packetization layer like IP or Wireguard

The other programmers confidently told me that the application would never send fragmented packets

They were surprised by my packet traces

I was often surprised too!

People on Stack Overflow, etc. also disagreed with my packet traces

val@valerieaurora.org

How to change minds of people sure they are right?

Write answers on Stack Overflow, etc?

➔ No, people will just downvote them

Write a blog post?

➔ No, people will just tell me I am wrong in the comments

Write a program to demonstrate the behavior?

➔ No, people will just claim they already knew the correct answer and that this is a boring silly unnecessary program

val@valerieaurora.org

Flashback: the TCP/IP Drinking Game

The TCP/IP Drinking Game was a session at early DEFCON conferences

I couldn't go to the game in person and it was 2000 so no videos

So I made up a game that was literally questions about TCP/IP

Played the first game with Alan Cox (of Linux networking fame)

Everyone thought it was fun!

And we learned things about networking

(The real TCP/IP Drinking Game was about using the most acronyms)

Solution: fragquiz

Write a program that:

- Makes people guess what will happen when a particular kind of packet is sent
- Records that guess
- Send an actual packet live and compare it to your guess
- At the end, it tells you your score
- Encourages you to send it to someone else

val@valerieaurora.org

Implementation goals

Minimal setup

Don't require superuser privileges

Don't write any packet tracing code

Don't be fooled by segmentation offload to the NIC

Don't use loopback or virtualization or other distorting workarounds

Don't require a separate server program

Be able to test IPv6 with only link local addresses

val@valerieaurora.org

Solution: Use TTL/hop limit and unprivileged ICMP listener

Open an unprivileged ICMP listener

Set the don't fragment socket option (OS-dependent)

Connect to any open TCP port at least one hop away (use same for UDP)

Set the TTL/hop limit to a low value

Send the packet

Listen for TTL/hop limit exceeded ICMP message from a router

Read IP header of original packet from ICMP message

Still some problems

Unprivileged ICMP listeners are available on Linux and MacOS

- But Linux only allows ECHO REQUEST/REPLY so must run as root 😡
- Linux needs a kernel patch to catch up to MacOS 😈

Many networks do not generate TTL/hop limit exceeded correctly

- Fortunately the RIPE meeting legacy network does for both IPv4 and IPv6!

Be able to test IPv6 with only link local addresses

Lots of people don't have IPv6 connections yet

Wrote a baby port scanner to find an open TCP port on the local link

But not working yet - need to force a hop through a router?

For now, you need a real IPv6 connection to test IPv6

Fortunately we are at a RIPE meeting!

Use **ripemtg-legacy-87** network - **ripemtg** doesn't generate IPv4 TTL correctly

val@valerieaurora.org

A few implementation details

If a router reassembles a packet before sending an ICMP TTL/hop limit exceeded, this won't work

But surely no router would bother doing that!

... Actually a lot of routers reassemble packets first 😭

Solution: automatically probe with increasing TTL/hop limit until we hit a router that DOESN'T reassemble packets before sending ICMP TTL/hop limit exceeded

OS-level path MTU caching changes behavior—TODO: clear MTU cache

Ready to play?

I will run the program live

The entire audience will vote for each answer

You can also keep track of your own score

val@valerieaurora.org

Bonus: PLPMTUD path MTU search algorithm

I co-implemented PLPMTUD on a packetization layer similar to Wireguard with Salman Aljammaz and James Tucker

RFC 8899 says: "Implementations could optimize the search procedure by selecting step sizes from a table of common PMTU sizes"

Just send packets of all the common PMTU sizes at the same time

Every ten minutes, send PMTU+1 size packet to see if MTU increased

~1 RTT latency, extremely simple, no timers, state, etc.

Who else has implemented this? Send it to me: val@valerieaurora.org

Run fragquiz at home

https://valerieaurora.org/fragquiz

It kind of ended up being a command line packet generator?

Patches welcome!

Read my cool debugging stories https://bit.ly/bestbugs

Let me know if you have hard systems problem that no one else can solve

Or if you need something like **fragquiz**

val@valerieaurora.org