

PMTU Path MTU Discovery

Some Servers are Unusable for Many Internet Users

Summary

Internet computers, mostly servers, sometimes send packets too large for part of a given path. Not handling this correctly can make the server unusable for some people.

Correct handling of oversize packets is by one of two means, as chosen by the sending computer (the server):

- Permitting Packet fragmentation - used mostly by older systems
- Path MTU discovery - asking for ICMP notification when fragmentation would be needed

By default, modern servers disable fragmentation and try to use path MTU discovery, but sometimes the system administrators block the ICMP notifications. Overall, these servers ask for ICMP notification of packets that are too large and then refuse to accept the notifications they ask for.

Unable to learn that they are sending packets are too big for some paths, the servers are unable to send data to some users. To overcome this they need to accept appropriate ICMP types or allow fragmentation.

Path MTU Discovery

Every network link has a maximum packet size called the link's MTU (Maximum Transmission Unit). The full path from one computer to another may travel across many links with different MTUs. The smallest MTU for all the links in a path is the path MTU.

If a packet starts out on a network segment with a large MTU, it may arrive at a link with a smaller MTU and be too big to fit. Most servers are on segments with large MTUs, but it is increasingly common for internet users to be connected via links with reduced MTUs, so it is becoming common for some packets to be too big.

How the problem of oversize packets has been handled has evolved considerably over time. The original approach was to send only small packets corresponding to the TCP/IP default MTU (576 bytes). (To this day, a sending system needs permission from the receiving system to send larger packets, but that permission is given as a matter of routine.)

For some packets, especially those sent by older equipment, an oversize packet can be sent by breaking it into fragments and sending the fragments as smaller packets. The fragments can be reassembled downstream to reconstruct the original large packet, but this packet fragmentation has several problems involving both efficiency and security.

Newer servers try to optimize their transmissions by discovering the path MTU and sending packets of the maximum size when there's enough data to fill them. The procedure for doing this was standardized and published as RFC 1191 in 1990, but it did not become widely deployed until years later. By mid 2002, 80% to 90% of computers on the internet used path MTU discovery.

PMTU Path MTU Discovery

Some Servers are Unusable for Many Internet Users

The basic procedure is simple - send the largest packet you can, and if it won't fit through some link get back a notification saying what size will fit. The notifications arrive as ICMP (Internet Control Message Protocol) packets known as "fragmentation needed" ICMPs (ICMP type 3, subtype 4). The notifications are requested by setting the "do not fragment" (DF) bit in packets that are sent out.

Some network and system administrators view all ICMPs as risky and block them all, disabling path MTU discovery, usually without even realizing it. Of the several dozen ICMP types and subtypes, some do pose some risk, but the risk is mostly mild and is of the "denial of service" nature. That is, an attacker can use them to interfere with service on and from the network.

By blocking all ICMPs the administrator himself interferes with service on and from his own network. Unless he also turns off path MTU discovery on his network's servers, he makes his servers unusable by users with reduced-MTU links in their paths. Because service is affected only in relatively unusual cases, it can be difficult to convince the administrator that a problem exists. The prevalence of such "unusual" cases is growing rapidly though.

Administrators who want to block all ICMPs should disable path MTU discovery on their computers, especially on their servers. It makes no sense to ask for ICMP notifications and then refuse to accept them. In addition, doing so opens the server to a special type of distributed denial of service attack based on resource exhaustion from a large number of fully-open connections.

Cisco's website has summary instructions for disabling path MTU discovery for these systems:

- Windows 95/98/ME
- Windows NT 3.1/3.51 and 4.0
- Windows 2000/XP
- Solaris
- HP-UX

Elsewhere on Cisco's site they recommend using the Dr. TCP utility for Windows.

Disabling path MTU on Windows systems requires modifying the Windows Registry. Microsoft's support website has information and cautions about the Registry.

Path MTU discovery can be turned off:

- Under Linux, with the command: `echo 1 >/proc/sys/net/ipv4/ip_no_pmtu_disc`
- Under FreeBSD, by using sysctl(8) to zero: `tcp.path_mtu_discovery`
- On Cisco routers, with the configuration command: `no ip tcp mtu-path discovery`

Example Path MTU Discovery Failure Scenario

Joe User has a DSL line. Like many DSL lines, Joe's line actually comes from a large DSL wholesaler and uses PPPoE (PPP over Ethernet) to create a logical channel between Joe and his

PMTU Path MTU Discovery

Some Servers are Unusable for Many Internet Users

favorite ISP. The DSL line physically connects to a small PPPoE router with a built-in hub, which Joe uses as the basis for his home network.

The MTU on Joe's network is Ethernet's usual 1500 bytes. The MTU on his DSL line is also 1500 bytes, but PPPoE uses 8 of these bytes for an 8-byte PPPoE header, so the MTU of his PPPoE channel to his ISP is only 1492 bytes.

A computer on Joe's Ethernet doesn't know about the PPPoE channel. The largest packet it can receive is a 1500-byte Ethernet packet, so when Joe uses his computer to connect to a remote server, his computer tells the server it's ok to send packets up to 1500 bytes.[*]

Establishing connections uses only very small packets, so Joe can connect to the server without any MTU issues, but as soon as Joe asks for enough data to fill a 1500-byte packet, the server sends a 1500-byte packet. When that packet gets to Joe's ISP, it doesn't fit down the PPPoE channel. The packet has the DF bit set, telling Joe's ISP to drop the packet and send the server an ICMP saying what packet size will fit. The ISP sends an ICMP saying the largest size is 1492 bytes.

If the server gets that ICMP, it re-sends the first 1492 bytes as a new packet. This new packets fits down the PPPoE channel, so Joe's computer gets it and sends back an acknowledgement. The server continues sending 1492-byte packets until Joe has what he asked for. In most cases the server will remember the reduced path MTU, typically for ten minutes, and use it for future connections from Joe's computer.

However, if the server does not get that ICMP, things go downhill fast. The server is expecting an acknowledgement from Joe's computer, but Joe's computer didn't get the packet, so the acknowledgment never comes. After awhile, the server gives up waiting and sends the same 1500-byte packet again. Joe's ISP sends back another ICMP. The server doesn't get it again and tries sending the 1500-byte packet a third time, then a fourth, a fifth, and so on.

Meanwhile Joe's computer can't tell this is happening and is waiting for a response from the server. Eventually, it gives up and sends the server a connection reset. It reports a network failure to Joe, who is left wondering what happened. He may soon discover he can access nearly all other sites, just not the one he wanted.

If the server he's trying to access is a web server, he may get part of the website, such as getting the ads but not the main page content. This happens because the pages normally displayed for many websites are assembled from data obtained from multiple servers on a variety of networks. The networks dispensing the ads all support path MTU discovery.

If Joe complains to the server operator, they may tell him the problem has to be his ISP because other users can view their server. However, Joe's ISP is not doing anything wrong, but the server operator is. PPPoE and other connections with reduced MTUs are perfectly legitimate and increasingly common. Having a server ask for ICMP notification that it refuses to accept is what's

PMTU Path MTU Discovery

Some Servers are Unusable for Many Internet Users

broken behavior. Some later day, when the server site has gotten enough complaints from people like Joe to believe something is wrong, it may either stop asking for the ICMPs or start accepting the useful ones. Then Joe will be able to access it.

[*] What Joe's computer actually tells the remote server is that the largest payload it can send in any packet is 1460 bytes, which is the 1500-byte maximum Ethernet packet size minus 40 bytes of overhead. This maximum payload is called the MSS (Maximum Segment Size).

Workarounds

There is something Joe can do in the meantime. He can reduce the MTU setting on his computer's Ethernet interface to the PPPoE MTU, 1492 bytes. Then when his computer first connects to the server, the maximum allowed packet size it gives the server will be 1492 bytes, and the server's packets will fit down the PPPoE channel.

In other words, if he is knowledgeable enough, Joe can overcome the server network misconfiguration by doing manually what is supposed to happen automatically.

If Joe's computer is a Windows system, changing his MTU requires changing a Windows Registry setting. This is normally something that only experienced Windows-savvy users should attempt. However, misconfigured servers causing path MTU discovery problems for DSL users has become common enough that there are Windows "tuning" programs available that will do this.

If Joe runs Linux or another UNIX variant, setting the MTU is much easier, using the "ifconfig" command. However, Linux (and perhaps other UNIX variants) has something even better. He can reduce the MTU for only his default route out the PPPoE/DSL channel while continuing to use the full 1500-byte MTU for traffic on his own network.

Tunnels and Path MTU Discovery

Here at NetHeaven, we do not sell PPPoE DSL connections, but we do sell network tunnels that use VPN technologies. Some of these tunnels can have reduced MTUs, and our tunnel users can run into this problem.

However, many of our tunnel users use their tunnel to connect only a single host. Then the host running the tunnel sees the tunnel device as a network device with reduced MTU. Being aware of the lower MTU, it takes that into account when giving a maximum packet size to servers upon making connections, avoiding any path MTU discovery problems.

Overall, whether our tunnel users need worry about the path MTU discovery problem depends both on whether they use the tunnel to connect a network or a single host and on what tunnel technology they use.

Among the tunnel types we support, PPTP/MPPE tunnels are usually tunnels to a single client computer, usually a Windows or Mac system. These need not worry about path MTU discovery.

PMTU Path MTU Discovery

Some Servers are Unusable for Many Internet Users

Our Linux users running SSL-wrapped PPP tunnels also need not worry about path MTU discovery, because these tunnels have their MTU set to 1500.[**]

Users of CIPE and OpenVPN tunnels are likely to encounter occasional path MTU discovery problems if they use the computer running the tunnel as a router to connect a LAN. Fortunately it's these users who are most likely to be capable of adjusting the MTU of their LAN systems.

In addition, our tunnel users who use their tunnels to run servers should be aware of the path MTU discovery issue so their own server doesn't wind up being one that can't be used by some other users.

[**] A tunnel having an MTU of 1500 when it's carried by packets on a link with reduced MTU is possible because PPP is level 2 even though it is encapsulated in level 3 TCP. PPP fragmenting packets is analogous to ATM shredding of packets and does not violate IP semantics. PPPoE could do this too, but apparently it does not. (If you didn't understand this paragraph, don't worry about it.)