60th IETF, PMTUD WG:

Path MTU Discovery Using Options
draft-welzl-pmtud-options-01.txt

Michael Welzl
http://www.welzl.at , michael.welzl@uibk.ac.at

Distributed and Parallel Systems Group
Institute of Computer Science
University of Innsbruck, Austria

Motivation

- In the end, (any kind of) PMTUD always loses a packet
  - it would be nice to avoid this
- Also, PMTUD should converge fast
- I am in favor of performance related signaling like ECN and XCP...
  - no "ECN flag" for PMTUD up to now (to avoid loss)
  - no "XCP" for PMTUD up to now (to converge faster)
- Proposal: add such signaling

How it works

- Before doing (no matter which) PMTUD, include "Probe MTU" IP option
  - Initialized with MTU of outgoing link
  - Updated by routers if MTU of incoming or outgoing link is smaller
  - "TTL-Check" field decremented by each "Probe MTU" capable router: used to determine if all routers were involved
- Receiver feeds back result to source
  - either at IP layer (not recommended) or at packetization layer (specified for TCP, SCTP and DCCP, with IPv4 and IPv6)
- Sender reacts to feedback
  - Information complete (from TTL-Check): immediately use new MTU value
  - Information incomplete: use as upper limit (i.e. starting point for RFC1191 PMTUD or to terminate PLPMTUD)

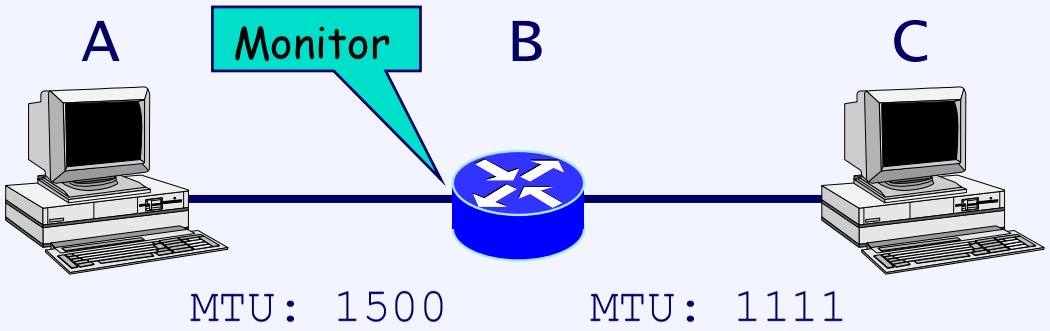
Potential benefits

- No loss, faster convergence
  - if lucky (result = PMTU)
  - really fast convergence if really lucky (all routers support the option)
- Less ICMP packets: less traffic, no risk of lost ICMP packet, reduced processing overhead for routers with small MTU
  - if lucky (result = PMTU)

Most beneficial for such routers, which are most beneficial for end points!

- May circumvent Black Hole Detection of RFC 1191 PMTUD in some cases
  - if received upper limit from PMTU-Options < value that would cause troubles due to routers that don't send "Fragmentation needed"
- Works across tunnels with small effort for endpoints (simply copy the option)
  - if supported by routers within a tunnel

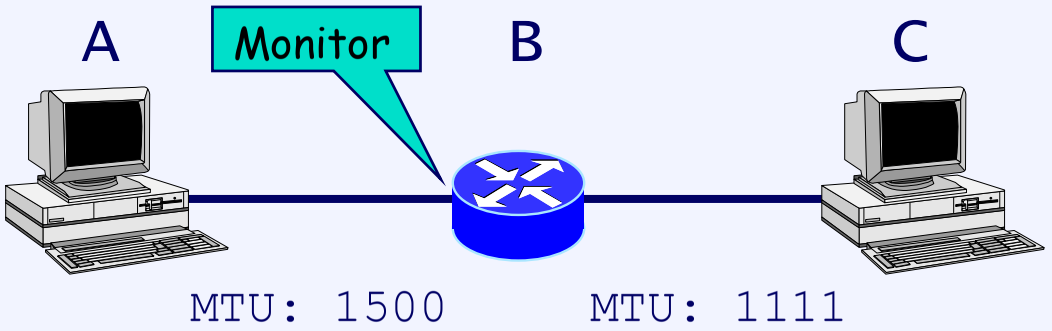
Example trace without PMTU-Options



3 normal PCs
100 Mbit/s Ethernet
Linux Kernel version 2.4.26
RedHat 9.0 standard installation
TCP file transfer A→B with netcat

IP Size	Sender	Receiver	Packet information
. . .			
1500	A	C	**lost**
576	B	A	ICMP Dest. unreachable
1500	A	C	**lost**
576	B	A	ICMP Dest. unreachable
1111	A	C	

Example trace with PMTU-Options



3 normal PCs
100 Mbit/s Ethernet
Linux Kernel version 2.4.26
RedHat 9.0 standard installation
TCP file transfer A→B with netcat
kernel patch installed!

IP Size	Sender	Receiver	Packet information
68	A	C	**pmtu-ask 1500**
60	C	A	**pmtu-reply 1111**
. . .			
1111	A	C	
1111	A	C	

Problems with IP Options

- Slow Path processing
- Some routers drop these packets

- Series of measurement studies carried out with NOP IP Option...
  data from 2004 (100 pings of each type per host (alternating), 1 ping / second):
  - 12889 different hosts addresses, 14508 different router addresses
  - path lengths ranging from approx. 5 to 35 (majority around 15-25)
  - 29.48% of hosts did not respond when there was an IP option
  - average additional delay of 26.5% of a RTT (options used in forward direction only)

- Unknown problems
  - processing effort for routers
  - delay / drop results when a long series of packets carry options
  - Does Slow Path processing lead to reordering?

Deployment considerations

- Clearly not recommendable for each and every e2e TCP connection
- Also, security issues
  - lie about number of routers: prevented by random initial TTL
  - send a MTU value that is too large:
    MAY be prevented by Nonce; not much harm otherwise
  - send a MTU value that is too small: cannot be prevented :-(
    note: IPSec authentication still feasible
- Recommended mainly for "special" scenarios
  - detecting increased PMTU, RTT-robust transport protocols (e.g. UDP)
- Experimental status envisioned

Patch, measurement results, future updates available from
http://www.welzl.at/research/projects/ip-options/