Inferring and Debugging Path MTU Discovery Failures

Matthew Luckie
University of Waikato
mjl@wand.net.nz

Kenjiro Cho
Internet Initiative Japan
kjc@iijlab.net

Bill Owens
NYSERNet
owens@nysernet.org

Abstract

If a host can send packets larger than an Internet path can forward, it relies on the timely delivery of Internet Control Message Protocol (ICMP) messages advising that the packet is too big to forward. An ICMP Packet Too Big message reports the largest packet size – or Maximum Transmission Unit (MTU) – that can be forwarded to the next hop. The iterative process of determining the largest packet size supported by a path by learning the next-hop MTU of each MTU-constraining link on the path is known as Path MTU Discovery (PMTUD). It is fundamental to the optimal operation of the Internet. There is a perception that PMTUD is not working well in the modern Internet due to ICMP messages being firewalled or otherwise disabled due to security concerns. This paper provides a review of modern PMTUD failure modes. We present a tool designed to help network operators and users infer the location of a failure. The tool provides fairly detailed information about each failure, so the failure can be resolved. Finally, we provide data on the failures that occurred on a large jumbo-capable network and find that although disabling ICMP messages is a problem, many other failure modes were found.

1 Introduction

Given a volume of data to send, it is desirable to encapsulate the data in the fewest number of packets possible, as "much of the cost of packetised communication is per-packet rather than per-probe" [1]. To send the fewest number of packets possible, a host must determine the largest IP packet size – or Maximum Transmission Unit (MTU) – supported by the path. The iterative process to determine the largest possible MTU on an end-to-end path by consecutively learning the next-hop MTU of each MTU-constraining link on the path is known as Path MTU Discovery (PMTUD). PMTUD allows a host or application to determine the largest IP packet size supported by an Internet path, and thus send the fewest number of packets.

Path MTU Discovery is documented in RFC 1191 for IPv4 [2] and RFC 1981 for IPv6 [3]. An application or kernel determines the largest supported MTU on an Internet path in an iterative manner, starting with the outgoing interface's MTU. It reduces the Path MTU each time a Packet Too Big (PTB) message is received until the destination host is reached, using the next-hop MTU value included in each successive PTB message. When this approach to PMTUD works, it allows an end host to quickly determine the Path MTU. There are, however, a number of well-known limitations of this technique [4], and work is in progress in the IETF to redefine the PMTUD method. This work discusses the current approach to PMTUD.

The failure modes of PMTUD are often difficult to debug, as they are triggered by relatively large packets. For example, a TCP connection may be established through a path where a PMTUD failure exists, as the TCP three-way handshake involves small packets that are unlikely to trigger a PMTUD failure. However, a PMTUD failure is likely to occur when either end of the TCP connection attempts to send a packet that is larger than can be forwarded through the path without fragmentation. A scenario like this is likely to cause the TCP connection to stall for some period of time before either failing, sending smaller packets, or allowing retransmitted packets to be fragmented.

This work introduces a technique for inferring and debugging PMTUD failures which occur on the forward path. Our technique uses a traceroute-like method to infer the location of a failure and the maximum packet size which can be forwarded through it. The technique does not infer failures that occur on the reverse path, such as the over-zealous firewalling of all inbound ICMP packets – including PTB messages – in order to protect a machine from security concerns related to ICMP or crude Denial of Service (DoS) attacks [5]. A recent study on the TCP behaviour of web-servers [6] found that PMTUD on the reverse path failed for 17% of 81776 targets tested and 35% of 500 popular web-sites tested – presumably because of middle-boxes which blocked inbound ICMP to the web-servers.

The rest of this paper is organised as follows. We begin by reviewing some of the known PMTUD failures in Section 2. We then discuss the debugging techniques used in this work to infer the location and mode of a PMTUD failure, and discuss the implementation of these techniques in our publicly available tool, scamper, in Section 3. In Section 4, we discuss the data collection that we did in support of this work, and then present some analysis of the results obtained in Section 5. Finally, we discuss a few anecdotes of strange behaviours we observed separate to the data collection for this study, before presenting our conclusions.

2 Path MTU Discovery Failure Modes

2.1 Router Configuration Issues

The most well known PMTUD failure mode is the ICMP Black Hole discussed in RFC 2923 [4]. The ICMP Black Hole problem has two halves; routers which do not send PTB messages due to misconfiguration or implementation bugs, and hosts which do not receive PTB messages due to a middle-box or firewall filtering them. The problem of router misconfiguration was first documented in RFC 1435 [7], where it was reported that code had been added to some routers to provide the capability to disable ICMP message generation in order to protect old BSD hosts, which were faulty in their handling of some ICMP messages. The RFC recommended that router code be updated to exclude PTB messages from suppression, as that particular message type did not trigger the faulty behaviour. However, it appears that this recommendation has either not been widely implemented, or operators are not using it. In the modern Internet, a router which does not send any ICMP message is almost certainly configured that way due to security concerns.

2.2 MTU Mismatches

An MTU mismatch occurs when a router and the path to the next-hop do not have a consistent understanding of the MTU. Specifically, a router believes that the path to the next hop is capable of forwarding packets larger than it actually can. Such a mismatch causes PMTUD to fail because the MTU change occurs below the IP layer, where a PTB message is not sent. A common scenario where this occurs is connecting a jumbo-capable gigabit Ethernet interface and a non-jumbo interface, which could be gigabit or fast Ethernet, across a switch. It can also occur if two jumbo interfaces are connected to a switch that does not support jumbo packets. The jumbo-capable Ethernet interface can send packets larger than 1500 bytes to the switch. However, the switch either cannot accept these packets, or cannot forward them to the next interface, and so the packets are silently discarded.

2.3 No Suggested Next-Hop MTU

The original IPv4 ICMP protocol [8] did not define the next-hop MTU field that PMTUD relies on to determine the largest packet size supported to the next hop. The next-hop MTU field was first defined in RFC 1191 [2], and makes use of otherwise unused space in the ICMP message. Routers that do not set the next-hop MTU field in a PTB message are easily detected, as the unused space is set to zero. In the face of a PTB message without a suggested next-hop MTU, current practice in the NetBSD kernel – among others – is to determine the size of the packet that caused the PTB message by examining the length field returned with the IP header embedded in the PTB message and then select a smaller packet size from a table of known MTU values.

2.4 Private Addressing

Some operators choose to use RFC 1918 [9] private addresses when numbering router interfaces in order to avoid using public addresses. The use of RFC 1918 addresses can cause PMTUD to fail if PTB messages are sent with an RFC 1918 source address, since packets with RFC 1918 source addresses are often dropped by ingress filters at the network edge.

2.5 Unspecified Implementation Bugs

There are other possibilities of PMTUD failure modes related to implementation bugs. For example, a router may send a PTB message with a suggested next-hop MTU larger than the size of the packet which caused it to be sent. Possible causes of this failure mode include not sending the next-hop MTU field in network byte order, or a router not adjusting internal state correctly when adding or removing headers. Other possible implementation bugs include: sending a PTB message with the embedded IP packet modified in some way such that the PTB message is unable to be matched with an active connection or application; sending an ICMP error message without generating a valid ICMP checksum; and sending an ICMP error message that is not a PTB message when it should have been.

3 Debugging Techniques

We have implemented two forward path debugging techniques into scamper, our publicly available measurement tool. The initial goal of the PMTUD code in scamper was to enable the detection of IPv6-over-IPv4 tunnels when comparing IPv4 and IPv6 paths between pairs of dual-stack nodes [10]. The code has evolved beyond this requirement, in part due to experiences in inferring tunnels in uncooperative paths.

To begin with, scamper conducts a standard traceroute with small UDP probes to unused ports. The purpose of this initial phase is to infer the forward IP path topology, determine which routers will provide ICMP feedback to small TTL-limited probes, and ensure that small probes are terminated somewhere in the path by an ICMP Destination Unreachable message so that scamper can distinguish between large probes being silently discarded and all probes being silently discarded. After the traceroute completes, scamper begins a PMTUD phase, where it solicits PTB messages in response to large probes until the destination is reached. scamper infers that PMTUD has failed when it does not obtain an expected reply packet to a probe the size of the currently known Path MTU value. When a PMTUD failure is detected, it uses one of two debugging techniques to infer the location of the failure and the largest packet which can be forwarded. Before we describe the two debugging techniques in detail, we describe the process by which the next-hop MTU is inferred.

3.1 Next-hop MTU Search

The purpose of the next-hop MTU search is to infer the largest packet size which can be forwarded to the next-hop. The general strategy is to, as quickly as possible, reduce a search space bounded by the smallest packet size to obtain a valid response and the largest packet size to not obtain a valid response, to find the underlying next-hop MTU. A binary search is not well suited to this task, for two reasons. First, MTU values tend to cluster due to the fairly limited combinations of media MTU values and encapsulations commonly used. Second, each probe that is discarded without the source receiving any ICMP feedback incurs a timeout delay that is often at least an order of magnitude larger than the delay incurred when probing with a packet that does obtain ICMP feedback. By default, scamper will retry a probe that obtains no ICMP feedback once, five seconds after sending the initial probe. In this scenario, a choice of probe size that does not obtain ICMP feedback incurs a ten second penalty before a different probe size can be tried. In order to determine the actual next-hop MTU as quickly and efficiently as possible, scamper is pre-loaded with a table of known MTU values.

When scamper begins a next-hop MTU search, it defines the lower bound by selecting an MTU in the table smaller than the failed probe, depending on three criteria. First, if the failed probe is larger than 1500 bytes, then scamper tries with a 1500 byte packet, as Ethernet is ubiquitous and likely to be the cause of an MTU restriction from larger frame sizes. Second, if the failed probe is larger than 1454 bytes, then scamper tries with a 1454 byte probe because 1454 is a lower bound of a series of MTU values that indicate some tunnel or encapsulation of IP over Ethernet. Otherwise, scamper selects the largest MTU from the table that is smaller than the size of the failed probe. The search for the initial lower bound is complete when ICMP feedback is obtained; the upper bound is reduced each time a probe for the initial lower bound does not obtain feedback.

After the lower bound is set, scamper then narrows the search space until it converges on the actual next-hop MTU. The approach to choosing a suitable probe size consists of three criteria, which are checked in order until a matching condition is found. First, if the lower bound of the search space is 1500 bytes or is a known MTU value in the table, and the upper bound is smaller than the next largest known MTU, then scamper probes with a packet one byte larger than the lower bound. The rationale for this is that if the search space is narrowed to within two entries in the MTU table, then there is a fair chance that the actual next-hop MTU is the current lower bound, and we can confirm this by sending a probe one byte larger. Second, if the next largest MTU in the table is smaller than the current upper bound, then scamper chooses this MTU as its next probe size. The rationale for this decision is that scamper can quickly determine the next-hop MTU if it is one of the values in the table. Lastly, if scamper is working within two known MTU values, then it will resort to a binary search to determine the next-hop MTU.

3.2 Inferring MTU without Feedback

This technique is used to infer the next-hop MTU and location of a hop that does not send PTB messages when it should. This technique is used when scamper does not obtain ICMP feedback with large packets the size of the current working Path MTU value. The technique consists of two stages. The first stage is a next-hop MTU search to infer the largest packet that can be forwarded, as described in Section 3.1. The second stage is a Time-to-Live (TTL) or Hop-Limit (HLIM) search of the forward path to infer the hop where large packets are silently discarded by determining the largest TTL or HLIM value that can be set in the IP header which still obtains an ICMP Time Exceeded message in response. This debugging technique is illustrated in Figure 1. This technique can infer a series of failure modes which are difficult to distinguish from each other, as there are many reasons why a source host may not receive a PTB message, and we have incomplete information to definitively infer why. We can, however, use a few heuristics to narrow the failure modes down.

If the farthest hop from which we obtain an ICMP Time Exceeded message with a large TTL-limited probe is immediately before a hop from which we obtain no ICMP Time Exceeded messages, we infer that the failure is likely to occur at the next hop either because all ICMP messages are disabled, or all ICMP responses from the router are being filtered somewhere in the network, possibly due to the use of RFC 1918 addresses. If we are able to receive ICMP

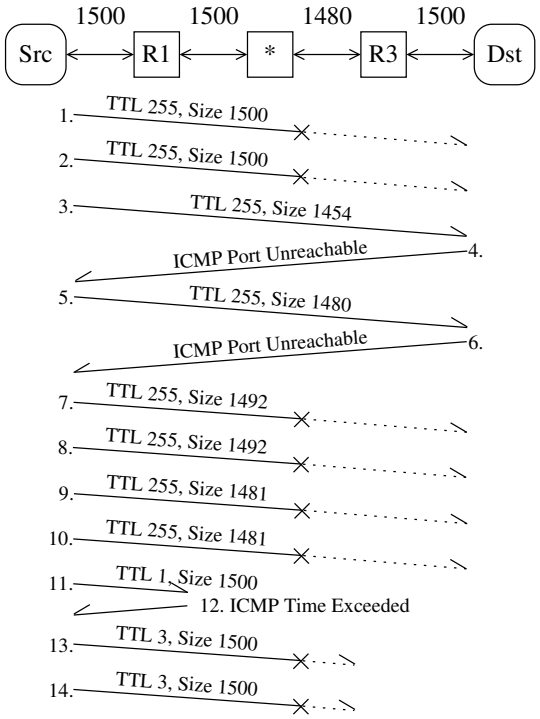

Figure 1: Inferring the MTU without feedback. An ICMP Black Hole exists between routers R1 and R3 where the MTU is restricted to 1480 bytes. A PMTUD failure is detected with probes 1 and 2, probes 3 to 10 infer that the next-hop MTU is 1480, and probes 11 to 14 infer that the large packets are probably being discarded at hop 2.

Time Exceeded messages with small TTL-limited probes from adjacent hops but we only receive Time Exceeded messages with large probes from the first hop in the path, we infer that the failure mode is likely to be either due to an interface being configured to not send any ICMP Destination Unreachable messages, or an MTU mismatch between the adjacent routers, or the PTB message originating from a different interface than the interface that sends Time Exceeded messages – with a source address that causes the PTB message to be subsequently filtered.

3.3 Inferring MTU with Invalid Feedback

This technique is used when a PTB message is received in response to a large probe, but the next-hop MTU included in the PTB message is either not set, or is larger than the probe which triggered the message. This technique uses a variation of the next-hop MTU search technique described in Section 3.1; instead of using the absence of a PTB message to reduce the upper-bound of the search space, this technique uses the faulty PTB message. This method can converge on the actual next-hop MTU fairly rapidly if ICMP feedback is received for packets smaller than the next-hop MTU past the faulty router, as the test for each probe size costs one round-trip-time (RTT). We use a slightly different technique if the path does not provide ICMP feedback after the faulty router due to another failure further in the path. When this occurs, scamper works progressively downwards through the MTU table soliciting faulty PTB messages rather than moving progressively upwards, as it would normally do. This is because scamper has to time-out on a probe which does not obtain ICMP feedback before it can send another probe, which has a much larger cost than sending packets which trigger faulty PTB messages.

3.4 Limitations

As the techniques we described rely on ICMP messages as feedback, they can be unreliable when ICMP rate-limiting is encountered. By default, scamper will send each probe twice before trying another probe type, with a five second timeout between each attempt. If two successive probes do not receive ICMP feedback due to rate-limiting, we may infer an incorrect next-hop MTU, or infer the wrong location of a failure, or infer a failure where one does not exist.

4 Methodology

We collected PMTUD failure data from two IPv4 hosts with 9000-byte MTU interfaces connected to networks that peer with Internet2, which itself is 9000-byte clean through the core, on April 28th 2005. The first location was from NYSERNet in New York, and the second was an Internet2 measurement machine in Chicago. The target list consists of 147 NLANR AMP machines, which are typically either on university campuses connected to the Internet2 network, or connected to networks that peer with Internet2. Most of the AMP machines connect to their host network with an Intel Pro100 Ethernet interface, which is capable of sending 1500 byte IP packets. Some have Gigabit Ethernet interfaces which are capable of sending IP packets larger than 1500 bytes, but are not configured to do so. The purpose of this dataset is to understand PMTUD failures on networks that can natively carry jumbo packets, and thus will require fragmentation at least at the edge of the campus network closer to each individual AMP machine.

5 Results

Of the 147 AMP machines in each dataset, we were able to complete a traceroute to at least 134 machines, or 91% of the target list. However, we inferred a PMTUD failure for 30% of the reachable machines. A summary of the failures is presented in Table 1. We categorised the failures into four groups: failure points where no ICMP messages

Dataset:	NYSERNet-east	nms1-chin	Intersection	Total
Location:	New York, NY	Chicago, IL	–	–
Hostname:	east.nysernet.org	nms1-chin.abilene.ucaid.edu	–	–
Date / Time:	Apr 28 2005, 21:50 EDT	Apr 28 2005, 20:10 CDT	–	–
Target Count:	147	147	147	–
Reachable:	136 (92.5%)	134 (91.2%)	134	–
PMTUD Failures:	41 (30.1%)	40 (29.9%)	25	–
No ICMP messages:	6 (6 unique)	5 (5 unique)	4 (4 unique)	7 unique
No PTB messages:	26 (17 unique)	27 (18 unique)	13 (13 unique)	22 unique
Incorrect PTB messages:	2 (2 unique)	2 (2 unique)	2 (2 unique)	2 unique
Target MTU Mismatch:	7 (7 unique)	6 (6 unique)	6 (6 unique)	7 unique

Table 1: Summary of the two data collections. 30% of reachable targets had a PMTUD failure.

are received (7), failure points where no PTB message is received (22), failure points where a PTB message is received with an incorrect next-hop MTU (2), and target machines which have an MTU mismatch with a router on their subnet (7). We identify a failure point by the IP addresses either side of the fault in the IP path. For example, the failure point would be identified as being between R1 and R3 in Figure 1. For each fault, we approached the technical and administrative contacts for the relevant AMP machine if the fault was determined to be local to that campus, or the operators of the relevant transit network.

We inferred seven failure points from which we did not receive any ICMP messages; of these, six were at routers where the next-hop MTU was inferred to be 1500 bytes, while the seventh had a next-hop MTU of 1536 bytes. One failure appeared to be caused by two successive routers in the path that both sent ICMP messages with a source address of 127.0.0.1, which were then discarded by a filter close to both of our measurement hosts. Similarly, another router located at the campus border used RFC 1918 addresses to number its interfaces, which also caused all ICMP messages from it to be filtered out. Another failure was caused by a BGP routing issue that, despite the fact that end-to-end connectivity was available, a significant portion of the routers on the forward path had no route back to the source host. This included one router which was therefore unable to send a PTB message to the source to signal that it was sending packets which were too big to forward. Finally, one other was due to a firewall designed to protect systems from security exploits by blocking all packets with a source or destination address matching particular addresses, including the addresses of core routers.

We found 22 hops from which we received ICMP Time Exceeded messages, but did not receive PTB messages when it was inferred that we should have. Sixteen of these hops had a next-hop MTU of 1500 bytes, accounting for just over two-thirds of the failures. Due to the method of counting hops where a failure occurs, the actual number of unique failure locations is a little less, as there is some repetition in the source address of some failure points. We determined that there were 20 failure locations. Two points were upgraded before a diagnosis could be obtained. We obtained a technical diagnosis of each fault for seven failures; three reported that they had disabled ICMP Destination Unreachable messages, while the other four were the result of an MTU mismatch or misconfiguration. For the 11 other failures for which we do not have a technical diagnosis, we probed the particular routers with a UDP probes to unused ports, in order to determine if they had disabled Destination Unreachable messages or not. Eight systems did not reply with a Destination Unreachable message.

We found two hops at one location from which we received a PTB message, but the next-hop MTU reported in the message was incorrect. The particular router would send a PTB message with a suggested next-hop MTU of 4586. It was, however, unable to forward packets larger than 4472 bytes to the next hop.

Seven targets were inferred to be on a subnet where nodes did not have a consistent agreement regarding the MTU. Two of the seven AMP targets with an MTU mismatch were able to receive IP packets larger than 1500 bytes, despite their use of 1500 byte MTU interfaces. One was able to receive packets up to 2016 bytes, while the other was able to receive packets up to 1506 bytes. We established that IP packets were arriving complete at these monitors by examining the probe packets with tcpdump.

6 Two Anecdotes

As discussed in Section 3.3, we implemented a technique to infer the correct next-hop MTU when a router sends a PTB message with an invalid next-hop MTU. The data included in this paper did not include such a failure, although we encountered one when implementing our tool. The router in question was located in New York City in the network of a large Internet Service Provider. For packet sizes between 4458 and 4470 bytes, the router would return a PTB message with an invalid next-hop MTU of 4470. Initial at-

tempts to determine the cause of what appeared to be a bug were difficult. Initially, we were told the fault was somehow related to the next-hop having an MPLS header with room for three 4-byte MPLS labels. It was also suggested that the fault could be a particular known router bug, although the bug number suggested seems unrelated. At this time we have been unable to determine the cause of the fault, and are pursuing this matter with a router vendor.

Unspecified router bugs can also prevent PMTUD from succeeding, as discussed in Section 2.5. During the course of scamper's development, we found an IPv6 router which appeared to route IPv6 packets over an IPv6-in-IPv4 tunnel with an MTU of 1480 bytes. However, for IPv6 packets larger than 1480 bytes, we did not receive any PTB messages. Rather, it sent two Destination Unreachable, No Route messages. The first message was returned with the IPv6 probe packet intact and caused scamper to cease PMTUD to the target beyond it. The second message – which we picked up by accident while monitoring all ICMPv6 packets into the machine – was unable to be matched to any probe we sent, as the encapsulated probe packet had the source and destination port fields zeroed out. We contacted the site responsible and reported the fault. To our knowledge, the fault was never identified and corrected, and went away when the particular path was replaced with a native IPv6 path.

7 Conclusion

The consensus is that Path MTU Discovery – in its current form – is unreliable due to it relying on the timely delivery of PTB messages, which are disabled or firewalled in many networks. We hypothesise that these failures go unnoticed in routine operational testing and monitoring, as they are only noticeable with larger probe packets. The default size of probe packets sent using traceroute and ping is too small to trigger PMTUD failures, and in the absence of packet loss with these basic connectivity measures, it is tempting to declare a path as fully operational.

In this paper, we presented a series of debugging techniques which infer PMTUD failures on the forward path. Using our implementation, we collected data on PMTUD failures found in jumbogram-capable networks. We found that of the reachable targets, 30% had a failure that would prevent efficient end-to-end communication from taking place. Less than half of these failures were caused by a configuration decision to disable the ICMP messages that are necessary for PMTUD to work. As the Internet MTU is raised, particularly as jumbo-capable Ethernet interfaces become more commonplace and jumbo transit services are offered, it seems likely that the classical PMTUD methods will continue to be strained. Until the new approach to PMTUD is completed and widely deployed amongst end-hosts, we believe our tool is a useful operational utility.

Acknowledgements

scamper's development was generously funded by the WIDE project in association with CAIDA from April 2004 to March 2005. The NLANR Measurement and Network Analysis Group (NLANR/MNA) is supported by the National Science Foundation (NSF) under cooperative agreement no. ANI-0129677. Matt Zekauskas (Internet2) collected the nms1-chin dataset. Maureen C. Curran and Joe Groff provided valuable editorial assistance. Matt Brown, Nevil Brownlee, Alan Holt, and Perry Lorier provided useful feedback on the paper.

References

[1] C.A. Kent and J.C. Mogul. Fragmentation considered harmful. *ACM SIGCOMM Computer Communication Review*, 17(5):390–401, 1987.

[2] J. Mogul and S. Deering. Path MTU Discovery. RFC 1191, IETF, November 1990.

[3] J. McCann, S. Deering, and J. Mogul. Path MTU Discovery for IP version 6. RFC 1981, IETF, August 1996.

[4] K. Lahey. TCP problems with Path MTU Discovery. RFC 2923, IETF, September 2000.

[5] R. van den Berg and P. Dibowitz. Over-zealous security administrators are breaking the Internet. In *Proceedings of LISA '02: Sixteenth Systems Administration Conference*, pages 213–218, Berkeley, CA, November 2002.

[6] A. Medina, M. Allman, and S. Floyd. Measuring the evolution of transport protocols in the Internet. *ACM SIGCOMM Computer Communication Review*, 35(2):37–52, April 2005.

[7] S. Knowles. IESG advice from experience with Path MTU Discovery. RFC 1435, IETF, March 1993.

[8] J. Postel. Internet Control Message Protocol. RFC 792, IETF, September 1981.

[9] Y. Rekhter, B. Moskowitz, D. Karrenberg, G.J. de Groot, and E. Lear. Address allocation for private internets. RFC 1918, IETF, February 1996.

[10] K. Cho, M. Luckie, and B. Huffaker. Identifying IPv6 network problems in the dual-stack world. In *Proceedings of the ACM SIGCOMM workshop on Network troubleshooting: research, theory and operations practice meet malfunctioning reality*, pages 283–288, Portland, OR., September 2004.