Internet Experiment Note # 20
Notebook section 2.3.3.6

A note on

Inter-Network Fragmentation and the TCP

John F. Shoch

January 1978

Xerox Palo Alto Research Center
Palo Alto, California 94305

Introduction

We continue to struggle with the problem of determining the appropriate upper limit on the size of packets in an inter-network environment. The problem is simple: some networks are prepared to generate packets which may be larger than the maximum packet size of a subsequent network.

The general approach to a solution is obvious: we must either 1) find a way to carve up large packets, or 2) legislate them out of existence; and there are strong differences of opinions on this subject:

> "A universal maximum length must be adopted if datagrams are to be used for internetwork data transmission." [Bochmann&Goyer 1977]
> "...a standard minimum maximum packet size for all nets ... appears ill-advised for several reasons...." [Sunshine 1977]

The TCP design currently attacks this problem by integrating a fragmentation scheme into the higher-level protocol; I believe this is a mistake. If we are going to support some form of internetwork fragmentation, it should be at a level "below" the TCP.

Tactics: The Fragmentation Alternatives Available to a Gateway

The critical decision about fragmenting a large packet would usually take place at an inter-network gateway: it might be necessary to forward the packet to the next gateway, through a network which could not directly handle a packet of that size. There are three possible alternatives available to the designer of a gateway process:

1. Discard the packet

The gateway could find itself completely unable to forward the packet, and merely discard it. In our model of gateway processes it is acceptable -- but certainly not desirable -- for a gateway to take such an action. One would hope that this information might propogate back through the inter-network, so that subsequent packets would be routed away from this black hole.

2. Intra-network fragmentation

The gateway can chose to do some sort of network-specific fragmentation, breaking up the oversize packet into smaller pieces, and allowing the gateway at the other end of this

network to do the reassembly. (This might be the Arpanet host-to-host protocol, or the Packet Radio Network's SPP protocol.)

Advantages:

---This form of fragmentation is esssentially performed by the network-specific driver, and is invisible to any higher level process in the gateway.
---The network can select a locally efficient protocol for transporting the fragments, and need not, for example, replicate the inter-network header on every fragment.
---If there is only one network in the path which necessitates fragmentation, all of the small pieces are brought together at the exit from that net, and the many pieces need not individually cascade through the rest of the path.

Disadvantages:

---This is, of course, more complicated than throwing the packet on the floor: there is some processing to be done, and the destination gateway must try to accumulate the fragments and perform reassembly.
---One could not take advantage of alternate gateways out of the intermediate network, since all of the fragments must reach the same exit gateway for reassembly.
---If there are many subsequent nets in the path which also require fragmentation, then there may be a fair amount of duplication as the same large packet is repeatedly fragmented and reassembled within each network.

It is important to note that this kind of fragmentation is always an acceptable decision by the gateway, since it remains transparent to other inter-network processes.

We should already be familiar with circumstances where it might take place:
---Oversize "messages" are fragmented into smaller "packets" by the Arpanet IMPs.
---In a low-bandwidth telephone link between two gateways, it may be desirable to fragment a large data packet in order to 1) minimize the probability of an error which would wipe out the whole unit, and 2) allow smaller packets (characters) to be interleaved with the fragments, and not be stuck for several seconds behind a lengthy data block

3. Inter-network fragmentation

In this case the gateway may instead divide the packet into smaller pieces which are themselves inter-network packets, or fragments. These fragments remain addressed to the ultimate destination, and must be reassembled there. The gateway uses a suitable means for identifying the related pieces: sequence numbers, hierarchical subdivision, etc.

Advantages:

---There is no necessity for the next gateway to do reassembly.
---If an entrance gateway fragments into inter-network fragments, they need not all depart the net through the same exit gateway. (Provided, of course, that the inter-network routing process in the gateway has enough information to sensibly decide how to apportion the outgoing pieces.)

Disadvantages:

---The gateway must be able to suitably divide up the original inter-network packet into pieces that the destination -- any destination -- can properly reassemble; it cannot merely use a network-specific technique, and may need to know some intimate details about the internals of the inter-network packet.
---The inter-network header must be replicated in all of the outgoing fragments.
---The destination now has to perform the reassembly task.

Strategy: The Selection of a System-wide Technique

Now, what kinds of strategies for handling over-size packets can we build with these alternatives?

1. Discard packets

The simplest strategy is to just use the simplest tactic: if one tries to route a packet out over a network for which it is too large, just drop it. Certainly not an acceptable approach.

2. Avoid small networks

If a source process can specify details of the inter-network route, and can obtain good information about the topology and size constraints in specific nets, then the source routing mechanism could merely dispatch "large" packets in such a way that they did not encounter any networks where they would not fit.

This strategy means that the gateways need only implement the simplest of the tactics: discard a packet if it wanders into the wrong gateway. But the preconditions are severe: if we are using some form of dynamic routing, we must be able to propogate the size parameter back through intervening networks to each source -- a non-trivial task. (If, on the other hand, we are limited to fixed source routing where the size limitations can also be known in advance, then this may become an attractive strategy.)

3. Use intra-network fragmentation

Here we permit gateways to use either of the first two tactics: discard a packet or fragment it in a network-specific fashion, but don't try to create inter-network fragments.

Such a strategy is usually accompanied with the specification of a maximum size for inter-network packets, implying that any network which wishes to join the inter-net must have a driver that is able (by some means) to forward a packet of this size to the next gateway.

As a corollary, a process can then be assured that any packet up to this size can be transmitted through any inter-network gateway. Furthermore, the destination process need not provide any kind of reassembly mechanism.

4. Use inter-network fragmentation

Finally, one could endorse the use of all three tactics by the gateways. Note that the gateway process would still have the choice of using some form of intra-network fragmentation, if there were some information available which made that the preferred choice.

Now, however, the destination process would be required to perform reassembly.

It is worth noting that this approach still specifies a packet size which all networks must be able to handle: an inter-network header plus (presumably) one byte of data.

Choosing a Strategy

We have very little emperical evidence upon which to base the choice of any particular strategy -- I know of very little analytical work describing the error probabilities or efficiency associated with fragmenting a packet into ten small ones which cascade through subsequent nets.

My current attitude is based on instinct, some experience, and a desire to minimize the

"complexity coefficient": endorsing intra-network fragmentation seems perfectly adequate, and eliminates the need for simple destination processes to perform reassembly.

This latter point is important: we should be able to layer our protocols in such a way that a destination process is able to specifiy if it is willing to do reassembly of inter-network packets, in return for possible benefits.

In any case, however, the selection of a fragmentation strategy should be an inter-net issue, and not part of a higher level end-to-end protocol.

A Proposal

In one does still want to provide inter-network fragmentation to support the TCP, here is one strategy which may prove worthwhile:

1. We include in the inter-network header a bit which indicates if the destination process is willing to reassemble inter-network fragments.

2. When a gateway receives a packet which must then be fragmented, this bit is checked. If it is off, then the gateway is not permitted to generate inter-net fragments, and must either use a network-specific method to reach the next gateway, or discard the packet. If the bit is on, the gateway has the option of doing inter-network fragmentation.

Thus, a minimal gateway need not even implement any inter-network fragmentation scheme. Alternatively, if a gateway has no intra-net method available, but the destination refuses to reassemble inter-net fragments, then the packet can only be discarded (although it would be nice if some sort of error message were returned to the source). (If this turned out to be a common situation -- which I doubt -- it might be a good arguement for adaptive source routing.)

3. This strategy would even support the coresident use of different inter-network fragmentation systems: an optional field in the internet header could indicate the type of fragmentation to be done on the way to this destination (based on byte sequence numbers, larger blocks, hierarchical, etc.).

4. In the process of establishing a connection, a destination could indicate to the source of packets what kind of reassembly it could perform, thus providing the information needed to appropriately mark packets which will then flow to that destination. The packets exchanged while setting up the connection would have to indicate that they themselves could not be fragmented.

By splitting out the fragmentation strategy, as outlined here, it would now be possible for other higher-level protocols to share this mechanism with the TCP, or decline to use it if it were deemed unnecessary.

Acknowledgements

The material in this note has evolved from many conversations with David Boggs, Ed Taft, Bob Metcalfe, and Yogen Dalal; a discussion with Vint Cerf several months ago led us to the notion of specifying the type of fragmentation as an option within TCP.

Bibliography

[Bochmann&Goyer 1977]
Gregor V. Bochmann and Pierre Goyer, *Datagrams as a public packet-switched data transmission service*, IEN #17, March 1977.

[Cerf&Kahn 1974]
Vinton G. Cerf and Robert E. Kahn, *A protocol for packet network intercommunication*, IEEE Trans. Communications, COM-22, 1974.

[Cerf&Postel 1978]
Vinton G. Cerf and Jonathan B. Postel, *Specification of Internetwork Transmission Control Program: TCP Version 3*, January 1978, ISI.

[LeMoli 1975]
G. LeMoli, *A Proposal for Fragmenting Packets in Internetworking*, INWG Protocol Note #21, April 1975.

[McKenzie 1974]
A. M. McKenzie, *Internetwork host-to-host protocol*, INWG Note #74, December 1974.

[Shoch 1978]
John F. Shoch, *Inter-network Naming, Addressing, and Routing*, IEN #19, January 1978.

[Sunshine 1977]
Carl A. Sunshine, *Interconnection of Computer Networks*, Computer Networks, 1977.

file: frag.b
January 29, 1978 8:07 PM