Network Working Group
Request for Comments: 1018

A. McKenzie
BBN Labs

August 1987

Some Comments on SQuID

Status of this Memo

This memo is a discussion of some of the ideas expressed in RFC-1016 on Source Quench. This memo introduces the distinction of the cause of congestion in a gateway between the effects of "Funneling" and "Mismatch". It is offered in the same spirit as RFC-1016; to stimulate discussion. The opinions offered are personal, not corporate, opinions. Distribution of this memo is unlimited.

Discussion

It appears to me that there are at least two qualitatively different types of congestion which may occur at Internet gateways. One form of congestion is the result of the merger of several independent data streams from diverse sources at a common point in their communication path. I'll refer to this as "Funneling". The architecture of the Internet (apparently) assumes that traffic flows are bursty and asynchronous; therefore congestion which occurs at the result of Funneling will typically be the result of "bad luck" as several independent bursts happen to arrive at a common point simultaneously. It is expected that Funneling congestion will be short-lived, just as individual bursts are. I don't claim that any such assumptions are documented or formalized; nevertheless I got a clear sense of this class of assumptions both from reading the protocol documentation and from personal recollections of long-past design meetings.

A second form of Internet congestion may arise during a prolonged (non-bursty) data transfer between hosts when the resulting traffic must pass through a node connecting two communications subsystems with significantly different throughput rates. I'll refer to this as "Mismatching". By contrast with Funneling, Mismatching can be caused by the innocent action of a single host, is highly repeatable (definitely not just "bad luck"), and will be long-lived.

RFC- 1016 discusses two interrelated strategies; one for when to send a SQ, and a second for what to do when an SQ is received. There is also a discussion of some experiments, which deal almost exclusively with Mismatching congestion. (I understand that the simulation can generate multiple flows, but these simply further increase the degree of Mismatch; the flow under study is long-lived by design.) It seems to me that the strategy RFC- 1016 proposes for sending SQ's, based on queue length, may be appropriate for Funneling Congestion, but inappropriate for Mismatch congestion, as discussed below. The host

behavior proposed in RFC- 1016 may be appropriate for both cases.

Since we assume that Funneling congestion is the result of short-lived phenomena, it is appropriate for gateways which are the sites of this congestion to attempt to smooth it without excessive control actions. This is the basis for the "hint" in the ICMP specification that maybe an SQ should be sent only when a datagram is dropped. It is the basis for the idea in RFC- 1016 that a gateway should be slow to cry "congestion" (SQK = 70% of queue space filed), even if persistent in attempting to eliminate it (SQLW = 50% of queue space filled). Since Funneling congestion is the result of the actions of multiple senders, the growth of internal queues is the only reasonable place to watch for its existence or measure its effects.

Mismatch congestion, on the other hand, is the result of incorrect or inadequate information about available transmission bandwidth on the part of a single sender. The sending host has available to it information about destination host capacity (TCP window size and ACK rate) and about local link capacity (from the hardware/software interface to the directly-connected network), but no real information about the capacity of the Internet path. As noted in RFC-1016, hosts can obtain the best throughput if their datagrams are never dropped, and the probability of dropped datagrams is minimized when hosts send at the appropriate steady-state rate (no "bunching"). Therefore, it is a disservice to a host which is the source of Mismatch congestion to wait a "long" time before taking control action. It would be preferable to provide immediate feedback, via SQ's, to the host as soon as datagrams with too short an inter-arrival time begin to arrive. The sending host could then immediately (begin to) adjust its behavior for the indicated destination.

There are, of course, many implementation issues which would need to be addressed in order to implement the type of SQ-sending behavior suggested here. Perhaps, though, they are not as severe as they might appear. Two specific issues and possible solutions, are:

1. How should a gateway differentiate between Funneling and mismatch congestion? Perhaps whenever there are more than q" items on an output queue to a slower subnet which have been received from a faster subnet, then look to see if any h" of them have the same source. It so assume Mismatch and send an SQ to that source. The "q" test might be implemented by a small set of counters which are incremented when a packet is placed on an output queue and decremented when a packet is sent. The search for a common source might require more cycles but be performed less often. The value of "q" would have to be small enough to give an early warning, but bigger than a small multiple of "h". The value of "h" would have to be big enough to avoid triggering

on common cases of source datagram fragmentation by an intermediate gateway.

2. How can a gateway determine which subnets are "slower" and faster", as well as appropriate inter-arrival times? There may be lots of clever ways for a gateway to measure the dynamic bandwidth of its directly-connected subnets. However, I'm more in favor of starting with configuration parameters related to the known (at installation time) general characteristics of subnet types (e.g. Ethernet is 10Mbps, ARPANET is 50 Kbps, SATNET is 100 Kbps, etc). This sort of approximation is quite adequate for determining which subnet is faster, or what inter-arrival time is appropriate for packets being routed to a slower subnet.

Summary

Funneling congestion and Mismatch congestion are qualitatively different, and it would not be surprising if different SQ-sending strategies were best for dealing with them. RFC- 1016 suggests a specific SQ-sending strategy which may be inappropriate for dealing with Mismatch congestion. This RFC suggests guidelines for an additional SQ-sending strategy for dealing with Mismatch. Hosts implementing the SQuID algorithm of RFC-1016 should be expected to achieve better performance if they received SQ's sent according to either or both of these strategies. However, all these ideas are still only in half-baked form; real engineering is clearly needed.