         SOFTWARE CHECKSUMMING IN THE IMP AND NETWORK RELIABILITY

   As the ARPA Network has developed over the last few years, and our
   experience with operating the IMP subnetwork has grown, the issue of
   reliability has assumed greater importance and greater complexity.
   This note describes some modifications that have recently been made
   to the IMP and TIP programs in this regard. These changes are
   mechanically minor and do not affect Host operation at all, but they
   are logically noteworthy, and for this reason we have explained the
   workings of the new IMP and TIP programs in some detail. Host
   personnel are advised to note particularly the modifications
   described in sections 4 and 5, as they may wish to change their own
   programs or operating procedures.

1. A Changing View of Network Reliability

   Our idea of the Network has evolved as the Network itself has grown.
   Initially, it was thought that the only components in the network
   design that were prone to errors were the communications circuits,
   and the modem interfaces in the IMPs are equipped with a CRC checksum
   to detect "almost all" such errors. The rest of the system,
   including Host interfaces, IMP processors, memories, and interfaces,
   were all considered to be error-free. We have had to re-evaluate
   this position in the light of our experience. In operating the
   network we are faced with the problem of having to perform remote
   diagnosis on failures which cannot easily be classified or
   understood. Some examples of such problems include reports from Host
   personnel of lost RFNMs and lost Host-Host protocol allocate
   messages, inexplicable behavior in the IMP of a transient nature,
   and, finally, the problem of crashes -- the total failure of an IMP,
   perhaps affecting adjacent IMPs. These circumstances are infrequent
   and are therefore difficult to correlate with other failures or with
   particular attempted remedies. Indeed, it is often impossible to
   distinguish a software failure from a hardware failure.

   In attempting to post-mortem crashes, we have sometimes found the IMP
   program has had instructions incorrect--sometimes just one or two
   bits picked or dropped. Clearly, memory errors can account for
   almost any failure, not only program crashes but also data errors
   which can lead to many other syndromes. For instance, if the address
   of a message is changed in transit, then one Host thinks the message
   was lost, and another Host may receive an extra message. Errors of
   this kind fall into two general classes: errors in Host messages,

McQuillan   [Page 1]

whether in the control information or the data, and errors in inter-
IMP messages, primarily routing update messages. In the course of
the last few years, it has become increasingly clear that such errors
were occurring, though it was difficult to speculate as to where,
why, and how often.

One of the earliest problems of this kind was discovered in 1971.
The Harvard IMP was sometimes crashing in an unknown manner so that
all the other IMPs were affected. It was finally determined that its
memory was faulty and sometimes the routing messages read out from
memory by the modem output interfaces were all zeroes. The adjacent
IMPs interpreted such an erroneous message as stating that the
Harvard IMP had zero delay to all destinations -- that it was the
best route to everywhere! Once this information propagated to the
other IMPs, the whole network was in a shambles. The solution to
this problem was to generate a software checksum for each routing
message before it was sent from one IMP, and to check it after it was
received at the other IMP. This software checksum, in addition to
the hardware checksum of the circuit, checks the modem interfaces and
memories at each IMP, and protects the IMPs from erroneous routing
information. The overhead in computing these checksums is not great
since the messages are only exchanged every 2/3 of a second.

In the first few months of 1973, we began to have a great deal of
trouble with the reliability of some IMPs, especially these in the
Washington area. The normal procedures of calling in and working
with Honeywell field engineers had not cleared up several of these
persistent failures, and it was felt that an escalation of BBN
involvement was needed to identify the exact causes of the problems.
Therefore, during much of February and March there were one or more
members of the staff at various sites in the network where hardware
problems were suspected. The first thing we found out was that the
operational IMP program did not give enough diagnostic information
about failures when they occurred, and that the available test
programs did not detect errors frequently enough to justify their
use. That is, the errors were appearing at rather low frequency,
from once every few hours to once every few days, compared to message
rates of once a second or faster. Therefore, we decided to try to
make the operational IMP program run when it could, and report more
information about detected hardware errors, rather than keep the
failing IMPs off the network for days at a time.

Modifications to the IMP program had two independent goals: we wanted
to make the software less vulnerable to hardware failures, and we
wanted the software to isolate the failures and report them to the
NCC. The technique we chose to use was generating a software
checksum on all packets as they are sent out over a line. We
suspected that the hardware failures in the Washington area were

happening between IMPs, that is, the packets were correct before they
were sent. Thus, a memory-to-memory software checksum, similar to
the technique installed two years before for routing messages only,
should be able to detect these errors. On March 13, a new version of
the IMP program was released with software checksum code. In this
program, when a packet is found to have an incorrect checksum it is
discarded, and a copy of the data is sent to the NCC. The previous
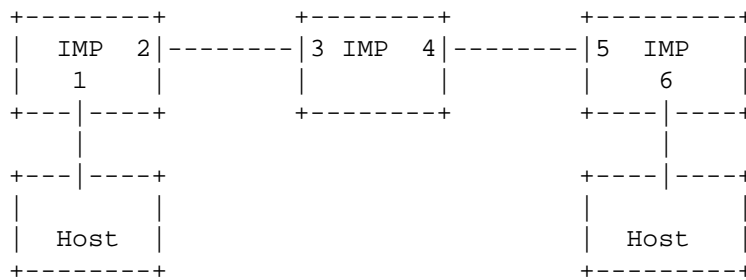IMP retransmits the packet, since an acknowledgment is not returned.

A partial list of the hardware problems that were uncovered by
software checksums, and subsequently fixed, includes:

   * One modem interface at the Aberdeen IMP dropped several bits
      from several successive words in transferring data into memory.

   * One modem interface at the Belvoir IMP picked one or two bits
      in a single word in transferring data into memory.

   * One modem interface at the ETAC TIP dropped the first word in
      transferring data out of memory.

   * A region in memory at the Utah IMP changed the low order two
      bits in some words on an irregular basis.

Each of these problems resulted in two or three detected errors per
day. There were other problems that were not detected by the
software checksum, such as dropped interrupts. This set of problems
may be explained by the electronics of the high-speed DMC on 316
IMPs. The first three machines cited above are 316 IMPs with 3 modem
interfaces, and they are the only such machines in the network. The
third interface is in a separate drawer and the total bus length
seems to be too long for the driving electronics in the original
design. We are presently investigating various ways to fix these
problems, and have had some success already.

2. An End-to-End Software Checksum on Packets

This last experience, and the earlier checksum on routing messages,
proved the value of a software checksum on all inter-IMP
transmissions. We have decided to extend the checksum to detect
intra-IMP failures as well, and make software checksums on all
network transmissions a permanent feature of the IMP system. We can
obtain an end-to-end software checksum on packets, without any time
gaps, as follows:

```
    +--------+         +--------+         +---------+
    |  IMP  2|--------|3 IMP  4|--------|5  IMP   |
    |   1    |         |        |         |    6    |
    +---|----+         +--------+         +----|----+
        |                                      |
    +---|----+                            +----|----+
    |       |                             |         |
    | Host  |                             |  Host   |
    +-------+                             +---------+
```

   * A checksum is computed at the source IMP for each packet as it
      is received from the source Host. (interface 1)

   * The checksum is verified at each intermediate IMP as it is
      received over the circuit from the previous IMP. (interfaces 3
      and 5)

   * If the checksum is in error, the packet is discarded, and the
      previous IMP retransmits the packet when it does not receive an
      acknowledgment. (interface 2 and 4)

   * The previous IMP does not verify the checksum before the
      original transmission, to cut the number of checks in half.
      But when it must retransmit a packet it does verify the
      checksum. If it finds an error, it has detected an intra-IMP
      failure, and the packet is lost. If not, then the first
      transmission was lost due to an inter-IMP failure, a circuit
      error, or was simply refused by the adjacent IMP. The previous
      IMP holds a good copy of the packet, which it then retransmits.
      (interface 2 and 4)

   * After the packet has successfully traversed several
      intermediate IMPs, it arrives at the destination IMP. The
      checksum is verified just before the packet is sent to the
      Host. (interface 6)

  This technique provides a checksum from the source IMP to the
  destination IMP on each packet, with no gaps in time when the packet
  is unchecked. Any errors are reported to the NCC in full, with a
  copy of the packet in question. This method answers both
  requirements stated above: it makes the IMPs more reliable and
  fault-tolerant, and it provides a maximum of diagnostic information
  for use in fault isolation. This expanded checksum logic was
  installed in the network on June 19.

  On of the major questions about such approaches is their efficiency.
  We have been able to include the software checksum on all packets
  without greatly increasing the processing overhead in the IMP. The

method described above involves one checksum calculation at each IMP
through which a packet travels. We developed a very fast checksum
technique, which takes only 2 msec per word. The program computes
the number of words in a packet and then jumps to the appropriate
entry in a chain of add instructions. This produces a simple sum of
the words in the packet, to which the number of words in the packet
is added to detect missing or extra words of zero. With the
inclusion of this code, the effective processor bandwidth of a 516
IMP is reduced by one-eighth for full-length store-and-forward
packets, from a megabit per second to 875 kilobits per second. That
is, the IMP now has the processing capability to connect to 17 full
duplex 50 kilobit per second lines, as compared to 20 such lines
without the checksum program. We are aware that this add checksum is
not a very good one in terms of its error-detecting capabilities, but
it is as much as the IMP can afford to do in software. Furthermore,
we emphasize that the primary goal of this modification is to assist
in the remote diagnosis of intermittent hardware failures.

3. Checksumming to Improve the Reliability of Routing

We mentioned earlier the catastrophic effects that follow for the
Network as a whole when a single IMP begins to propagate incorrect
routing information. The experience described above involved a
specific memory failure which has not recurred in the last two years,
but the problem is easily understood to be of a general nature. In
fact, we recently had another network-wide failure that was traced to
a hardware error that resulted in erroneous routing messages, after
we had installed a software checksum on all inter-IMP transmissions.
The problem we had were due to a single broken instruction in the
part of the IMP program that builds the routing message. As a
result, the routing messages from that IMP were random data, and the
neighboring IMPs interpreted these messages as routing update
information. When this happened, traffic flow through the Network
was completely disrupted and no useful work could be done until the
failed IMP was halted.

This kind of problem, the introduction of incorrect routing
information into the Network, can happen in three ways:

    * The routing message is changed in transmission. The inter-IMP
       checksum should catch this. The bad routing messages we saw in
       the Network had good checksums.

    * The routing message is changed as it is constructed, say by a
       memory or processor failure, or before it is transmitted. This
       is what we termed above an intra-IMP failure.

    * The routing program is incorrect for hardware or software
       reasons.

We have attempted to solve the last two kinds of problems by
extending the concept of software checksums. The routing program has
been modified to build a software checksum for the routing message as
it builds the message, just as if it came from a Host. It is
important that this checksum refer to the intended contents of the
routing message, not the actual contents. That is, the program which
generates the routing message builds its own software checksum as it
proceeds, not by reading what has been stored in the routing message
area, but by adding up the intended contents for each entry as it
computes them. The process which sends out routing messages then
always verifies the checksum before transmitting them. This scheme
should detect all intra-IMP failures.

Finally, the routing program itself can be checksummed to detect any
changes in the code. The programs which copy in received routing
messages, compute new routing tables, and send out routing messages
each calculate the checksum of the code before executing it. If the
program finds a discrepancy in the checksum of the program it is
about to run, it immediately requests a program reload from an
adjacent IMP. These checksums include the checksum computation
itself, the routing program and any constants referenced. This
modification should prevent a hardware failure at one IMP from
affecting the Network at large by stopping the IMP before it does any
damage in terms of spreading bad routing. A version of the IMP
program with this added protection for routing was released on May
22.

In the first few months of 1973, there have been several other
efforts aimed at improving the reliability of the Network, in
addition to software checksumming in the IMPs. At the same time that
we were discovering inter-IMP failures with the software checksum
packets, we began to notice a different kind of problem with intra-
IMP failures. In these cases we were primarily faced with memory
problems, and they often affected the IMP program itself, rather than
the packets flowing through the IMP. Our first attack on this
problem was to build a PDP-1 program to verify the running IMP and
TIP programs at a site against the correct core images held at the
PDP-1. The program interrogates the IMP with DDT messages, and
prints out a list of discrepancies. Using this program, we have
already found memory failures at one site.

4. TIP Modifications

   The hardware difficulties which we began to experience during the
   first few months of 1973 had two effects on Host-to-Host
   communication. First, the intermittent modem interface failures, of
   the type seen at Belvoir, Aberdeen, and ETAC, meant that messages
   were occasionally lost by the network. This loss is reported to the
   transmitting Host by the "Incomplete Transmission" message generated
   by the source IMP; the Host must then decide whether to retransmit or
   to take some other action. Second, the higher than normal incidence
   of machine failures meant that the network sometimes "partitioned" so
   that there was no path between the two communicating Hosts. (It
   should be noted that, contrary to the original design, two sites are
   currently connected to the network by only a single path; other
   similar connections are planned. For any such sites, any failure
   along the single path will be seen as a partition.) Since a TIP acts
   as a Host for its users, its resilience when these types of failures
   occur has a major effect on user satisfaction.

   Prior to this time the TIP program "aborted" the user's connection if
   it received an Incomplete Transmission indication from the IMP
   program. In March the TIP program (and the programs of several other
   Hosts) was changed to retransmit messages for which the Incomplete
   Transmission indication was returned; some Hosts (e.g. MULTICs) have
   done this from the start. This modification has turned out to be
   relatively simple, and we urge other Hosts to consider implementing
   some sort of error recovery software. On the other hand, it has not
   seemed reasonable to continue attempting to transmit when the program
   receives a "Destination Unreachable" indication, since this could
   arise either from a network partition or from a failure at the
   destination site. The interactive user is, of course, free to try
   again manually.

   A different situation pertains to tape transfers involving TIPs with
   the magnetic tape option. In these cases, the user would like to
   start the process and then ignore it until the transfer is finished.
   Network partitions, even if infrequent, may occur when tape transfers
   many hours in length are in progress. Therefore, we made a
   significant modification to the TIP magnetic tape option to include a
   sequencing mechanism in the tape transfer protocol which permits
   automatic recovery and transmission continuation after most kinds of
   network transients. With this mechanism in effect, and assuming a
   tape is mounted at the "other end", the complete transfer of a tape
   is possible with a single command given at either end. If the
   connection goes dead in mid-transfer, the TIP magnetic tape software
   will attempt to reopen the connection until successful and then
   continue the transfer from where it was left off. In addition to
   modifying the TIP magnetic tape option as specified above, we also

modified the TENEX program which is able to communicate with the TIP
magnetic tape option so that it remained compatible. These changes
were installed in April.

5. Future Plans

We have been considering some of the issues of network reliability
discussed above in connection with the development of the new High
Speed Modular IMP. This design effort and the experiences with the
current IMP system are, of course, linked together, and we have
already decided on several approaches to be taken in the new line of
IMPs:

   * The IMP will have a hardware CRC checksum generator which
      returns the checksum on a specified range of memory.

   * The IMP will use this facility to generate and check an end-
      to-end checksum on messages. This checksum will therefore be
      more comprehensive and better for error detection than the
      current software checksum. It will insure a high degree of
      reliability for Host transmissions.

   * In addition, the IMP will perform a verification of a packet
      checksum at each hop to provide diagnostic information. This
      check will be on an optional basis, whenever the system has
      available resources for the check.

   * The code for the new IMP system will be read-only (this is
      impractical for the present 516 and 316 IMPs), and the program
      will periodically checksum itself using the hardware CRC
      generator. We hope to design the program so that it can be
      reloaded in segments in the event of a detected error in the
      code, with no service interruption.

   * Finally, we are looking into the structure of an optional IMP-
      Host/Host-IMP checksum to complete Host/Host end-to-end
      checksum. Under such an arrangement, the IMP and Host could
      agree to verify the checksums on the messages transferred over
      the interface between them, and the appropriate signalling
      mechanisms would be provided to handled errors. With this
      technique in effect, two Hosts could be certain that their
      messages were delivered error-free or else they would be
      notified of an error, and could then retransmit their message
      if desired.

More details on any such modifications to the IMP and to the
IMP-Host interface will be published when appropriate.

[This RFC was put into machine readable form for entry]
[into the online RFC archives by Via Genie 12/1999]