Accepted for publication in *Electronics Letters*

Fast IP Packet Delineator

Abstract: A fast-acting synchronisation mechanism for Internet Protocol (IPv4) packet delineation from an ingress bitstream is presented. It builds on delineation and scrambling mechanisms developed for Asynchronous Transfer Mode (ATM) technology, but provides enhancements that cater for variable-sized packets. Implications of the scheme for IP version 6 are considered.

Introduction: Typically, IP data is transported over datalink layer technologies such as ATM or Point-to-Point Protocol (PPP) that, in turn, are carried by physical and optical layer protocols. Each of these protocols provides a number of services to the layer above. Frequently, this can lead to a duplication of functionality. Protocol de-layering is a process where several layers of a protocol stack are implemented as a single layer or eliminated, avoiding this duplication. De-layering a protocol stack can also enable various alarm timers to be set more stringently.

For example, to avoid the inappropriate triggering of network layer restoration activity during the period when datalink or physical layer functions may be resolving the error condition, it is normal practice to set alarm timers higher up the stack to larger values. Removing lower layer continuity functionality allows for fast acting IP layer protection to be implemented, possibly providing a more cost-effective solution.

This letter describes a novel and efficient IP packet delineation mechanism that provides a vital step towards protocol de-layering by removing the need for datalink layer framing. It can be readily implemented in hardware, providing wire-speed delineation within Local and Metropolitan Area Networks.

Operation with IPv4: A detailed definition of IP is provided in [1] and the IPv4 packet format is shown in Figure 1. Traditionally, delineation of IP packets, that is the identification of the packet boundaries within a data stream, is provided by the underlying datalink layer technology that encapsulates them. This is because IP is asynchronous, uses variable length packets and variable length headers, provides no byte alignment and has no explicit frame alignment sequence.

However, IP version 4 (IPv4) provides a number of reference points within the header that are either invariant, or vary in an easily verifiable manner. For example, the first *nibble* in every IPv4 packet contains the binary version field identifier “0100”. Next, the header checksum field is always present at a fixed offset to this version field. This operates on the entire IP packet header including options fields, should they exist. The checksum algorithm is the 16 bit one’s complement of the one’s complement sum of all 16-bit words in the header. For the purposes of computing the checksum, the value of the checksum field is zero. This is simple to compute and can be readily achieved using combinational logic.

The packet delineation functionality is illustrated in Figure 2. To obtain and maintain synchronisation the design assumes that the ingress stream is an abutted series of IPv4 packets. When user packets are unavailable for transmission, empty packets are inserted into the flow in a similar manner to ATM's idle cells (refer to Section 4.4 of ITU I.432 [2]). This has minimal detrimental effects on the efficiency of the scheme as idle packets are only generated in the absence of all other forms of IP data. The only penalty comes in the form of transmission latency. Once an idle packet commences transmission, it cannot be pre-empted. However, idle packets need only be 20 bytes in length, the minimum legal header size, for the delineation mechanism to maintain synchronisation.

Functionally, the action of the packet delineator may be summarised as follows: Firstly, clock recovery is performed on the incoming bitstream. The resultant clock pulses drive a free-running counter. Each bit of the bitstream is fed through a 480-bit Shift Register (SR) at each clock edge and is available via a transparent latch. 480 bits equates to the maximum packet header size. A free running Check Module (CM), containing combinational logic, rapidly performs the header error check provided the leading four bits in the SR contain binary "0100", indicative of IP version 4. The next four bits give the Header Length, expressed as multiples of 32 bits. This header length is then used to determine the range of bits over which the calculation should be made. The minimum legal value of 5 is used as a further assessment criterion. If a successful match is found, the CM sends a "Test_OK" signal to the Processing Engine (PE).

Upon receipt of a "Test_OK" signal from the CM, the Processing Engine initiates the delineation search by issuing a "lock" pulse. This causes the contents of the transparent latch to be frozen and the bit locations corresponding to the Packet Length field to be extracted. The lock pulse also causes a second transparent latch to retain the current Counter value.

The packet length field is modulo added to the latched counter value to make a prediction regarding the count expected for the next delineation point. This value is stored in count-order within the Prediction List (PL). If the current leading entry in the PL happens to match the latched Counter value, a Comparator yields a positive "Match" signal, indicating that a previous prediction has been successful. Under these circumstances, the *score* field associated with this previous prediction is incremented and placed in the *score* field alongside this new *expected count* entry. If no Match signal is provided by the Comparator, the *score* is initialised to zero.

If the previous score value exceeds a user-selectable threshold, then a confirmed delineation is deemed to have taken place. Thereafter, whilst predicted counts and actual header tests continue to be located as expected, a delineation synchronisation pulse is issued when the first bit of each packet leaves the SR. Under all circumstances, when the predicted count at the top of the prediction list becomes older than the counter, it is simply removed.

The robustness of the delineation can be controlled by selecting the number of sequential positive matches that must take place before delineation is announced (as typified by the score field). Given the size of the header check field in combination with the IP version signature field, it is anticipated that only a couple of sequential matches will be necessary to locate the true delineation point, provided payload scrambling is employed as in ATM [3]. This limits false positives arising from deliberately inserted header anomalies within the payloads, improving the robustness and reliability of the scheme.

The specificity of the header delineation testing process can be further improved by taking into account heuristic information. For example, bit 0 of the Flags field is reserved and must always be zero according to RFC791.

Performance Assessment: The probability of the "Test_OK" succeeding on a random bitstream in the SR is approximately 6.56 x 10^{-7}. To see this, consider that the Internet checksum field is 16-bits long, so the probability of the checksum succeeding on a random selection of bits is

$1/2^{16}$ or 1.53 x 10^{-5}. The 4-bit Version field has a 1/16 chance of being "0100", and the 4-bit Header Length field has an 11/16 chance of being greater than four. The firing of each "Test_OK" signal triggers a new checkpoint to be entered into the Prediction List. The probability of a predicted checkpoint also testing positive (a "double match"), and a delineation point being incorrectly signalled, is therefore 4.30 x 10^{-13}.

Table 1 shows the results of computer simulations performed on both random bitstreams and randomly generated IP packets. There is one feature of the Internet Checksum that warrants particular mention. Once a "Test_OK" signal has occurred, there is a small (approximately 1/1024 or 9.77 x 10^{-4}) chance that a correlated "Test_OK" will occur sometime immediately after it. This explains the small increase in average frequency of "Test_OK" signals in the trials using simulated IP packets, and also the increased size of the Prediction List.

The benefits of fast and simple delineation of IP packets at the IP layer appear to outweigh the potential disadvantages of false positive delineation points being identified. Because of the extremely low probability of finding a double match, and hence a bad packet, it is felt that the simple double match will be sufficient to guarantee delineation.

If each delineation signal is used only to determine the end of one packet and the start of another, with no check that the Total Length field is correct, then the effect of a false delineation signal is not particularly detrimental to the overall performance of the scheme. When a false delineation signal occurs, although the packet it interrupts will be cut short and an extra packet created, the subsequent packet in the stream will still be delineated correctly.

Adapting the Scheme for IPv6: IP version 6 (IPv6) promises to be the next widely deployed network layer protocol [4], likely to replace IPv4. It contains many amendments to the IPv4 packet structure, in particular, there is no checksum field in the IPv6 fixed header. IPv6 assumes that datalink/physical layer mechanisms will be in place to provide error checking and/or correction mechanism to ensure data transmit reliability. Although this is the antithesis of the de-stratification approach, it is possible to adapt the scheme to IPv6 whilst remaining within the existing "standards". For example, IPv6 supports the concept of Extension Headers. In particular, the Hop-by-Hop Options Header, allows operators to insert type-length-value (TLV) encoded options that are examined by every IPv6 router along a packet's path. By including an error check or delineation flag within a TLV value field the fast IP delineator is able to function satisfactorily. Rather than a simple delineation flag, using an error check provides the additional benefit of providing a checksum over the IPv6 header that may be absent if there is only a "leaner" or null datalink layer.

References

[1] University of Southern California, “Internet Protocol”, DARPA, RFC791, September 1981

[2] “B-ISDN User-Network Interface Physical Layer Specification”, ITU Recommendation I.432, March 1993.

[3] “B-ISDN Operation and Maintenance Principles and Functions”, ITU-T Recommendation I.610, November 1995.

[4] S.Deering, S.R.Hinden, “Internet Protocol, Version 6 (IPv6) Specification”, IETF RFC2460 Standards Track, December 1998.

Authors’ affiliations:

R. A. Bourne and C. I. Phillips (Department of Electronic Engineering, Queen Mary, University of London, Mile End Road, London E1 4NS)

Email address of corresponding author: r.a.bourne@elec.qmul.ac.uk

Figure Captions

Fig. 1 IPv4 Packet Format

Fig. 2 IPv4 Packet Delineator

Tables Captions

Table 1 Results of Computer Simulations

Figure 1

<table>
<tr><td>Version</td><td>IHL</td><td colspan="2">Type of Service</td><td colspan="4">Total Length</td></tr>
<tr><td colspan="4">Identification</td><td>Flags</td><td colspan="3">Fragment Offset</td></tr>
<tr><td colspan="2">Time to Live</td><td colspan="2">Protocol</td><td colspan="4">Header Checksum</td></tr>
<tr><td colspan="8">Source Address</td></tr>
<tr><td colspan="8">Destination Address</td></tr>
<tr><td colspan="8">Options (+ Padding)</td></tr>
<tr><td colspan="8">Data (Variable)</td></tr>
</table>

Figure 2

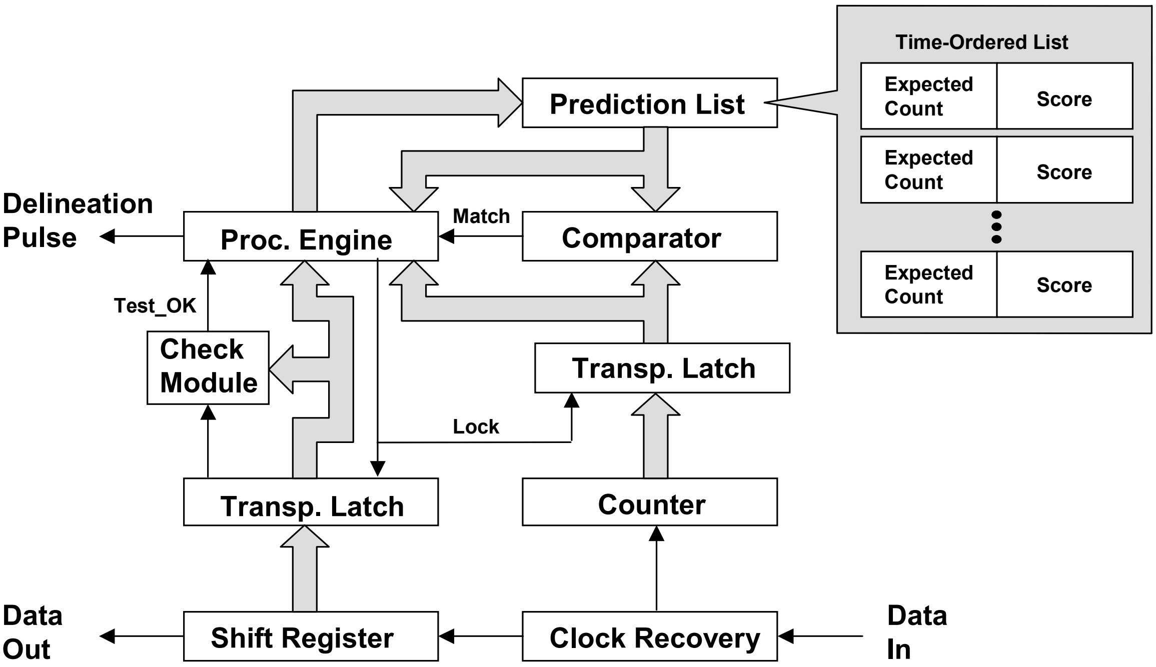
Time-Ordered List
Expected Count
Score
Expected Count
Score
Expected Count
Score
Prediction List
Delineation Pulse
Proc. Engine
Match
Comparator
Test_OK
Check Module
Transp. Latch
Lock
Transp. Latch
Counter
Data Out
Shift Register
Clock Recovery
Data In

Table 1

	Sample size	*Probability/Frequency*		*Standard deviation*	*Max size of PL*
		Test_OK	Double match		
Theory		6.56E-07	4.30E-13		
Random bits	1.93E+09	6.47E-07	0	4.84E-08	3
Random IP packets	1.82E+09	7.14E-07	0	4.65E-08	4