Network Working Group
Request for Comments: 1201
Obsoletes: RFC 1051

D. Provan
Novell, Inc.

February 1991

Transmitting IP Traffic over ARCNET Networks

Status of this Memo

This memo defines a protocol for the transmission of IP and ARP packets over the ARCnet Local Area Network. This RFC specifies an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

1. Introduction

This memo specifies a method of encapsulating Internet Protocol (IP) [1] and Address Resolution Protocol (ARP) [2] datagrams for transmission across ARCNET [3] using the "ARCNET Packet Header Definition Standard" [4]. This memo offers a replacement for RFC 1051. RFC 1051 uses an ARCNET framing protocol which limits unfragmented IP packets to 508 octets [5].

2. ARCNET Packet Format

In 1989, Apple Computers, Novell, ACTINET Systems, Standard Microsystems, and Pure Data Research agreed to use the ARCNET datalink protocol defined in "ARCNET Packet Header Definition Standard" [4]. We'll begin with a brief description of that protocol.

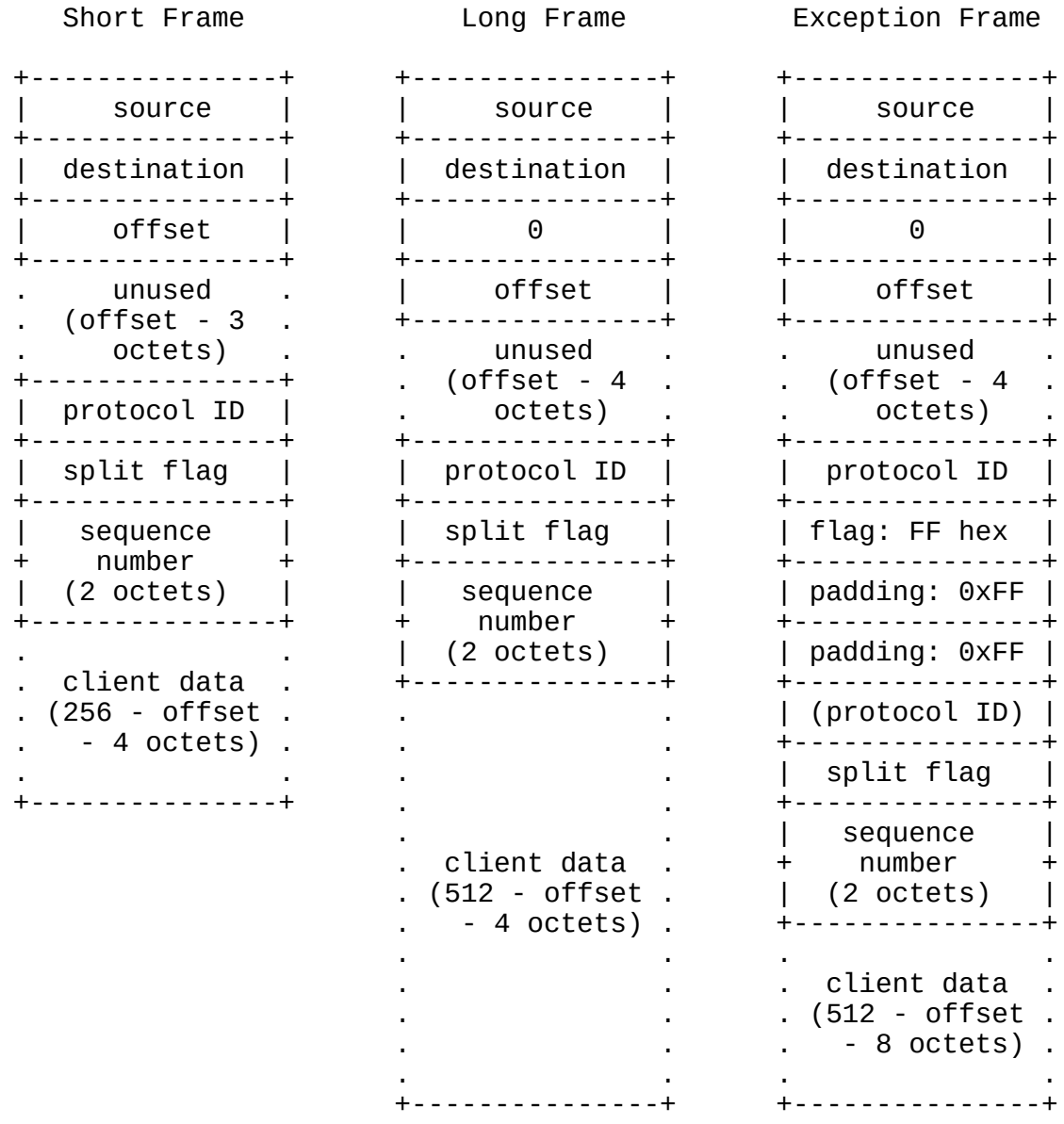
2.1. ARCNET Framing

ARCNET hardware supports two types of frames: short frames, which are always 256 octets long, and long frames, which are always 512 octets long. All frames begin with a hardware header and end with the client's data preceded by a software header. Software places padding in the middle of the packet between the hardware header and the software header to make the frame the appropriate fixed length. Unbeknown to the software, the hardware removes this padding during transmission.

Short frames can hold from 0 to 249 octets of client data. Long frames can hold from 253 to 504 octets of client data. To handle frames with 250, 251, or 252 octets of data, the datalink protocol

introduces a third frame type: the exception frame.

These three frame formats are shown here. Except as noted, each block represents one octet.



These packet formats are presented as software would see them through ARCNET hardware. [3] refers to this as the "buffer format". The actual format of packets on the wire is a little different: the destination ID is duplicated, the padding between

the offset field and the protocol ID field is not transmitted, and there's some hardware framing information. In addition, the hardware transmits special packets for buffer allocation and reception acknowledgement which are not described here [3].

2.2. Datalink Layer Fragmentation

ARCNET hardware limits individual frames to 512 octets, which allows 504 octets of client data. This ARCNET datalink protocol allows the datalink layer to break packets into as many as 120 fragments for transmission. This allows ARCNET clients to transmit up to 60,480 octets in each packet.

The "split flag" describes datalink layer packet fragments. There are three cases: an unfragmented packet, the first fragment of a fragmented packet, and any other fragment of a fragmented packet.

Unfragmented packets always have a split flag of zero.

The first fragment of a fragmented packet has a split flag equal to ((T-2)*2)+1, where T is the total number of fragments to expect for the packet.

Subsequent fragments of a fragmented packet have a split flag equal to ((N-1)*2), where N is the number of this fragment. For example, the fourth fragment of a packet will always have the split flag value of six ((4-1)*2).

The receiving station can identify the last fragment of a packet because the value of its 8-bit split flag will be one greater than the split flag of the first fragment of the packet.

A previous version of this ARCNET datalink protocol definition only allowed packets which could be contained in two fragments. In this older standard, the only legal split flags were 0, 1, and 2. Compatibility with this older standard can be maintained by configuring the maximum client data length to 1008 octets.

No more that 120 fragments are allowed. The highest legal split flag value is EE hex. (Notice that the split flag value FF hex is used to flag exception packets in what would otherwise be a long packet's split flag field.)

All fragments of a single packet carry the same sequence number.

2.3. Datalink Layer Reassembly

The previous section provides enough information to implement

datalink reassembly. To avoid buffer allocation problems during reassembly, we recommend allocating enough space for the entire reassembled packet when the first fragment arrives.

Since fragments are sent in order, the reassembly procedure can give up on a packet if it receives a fragment out of order. There is one exception, however. It is possible for successfully received fragments to be retransmitted. Reassembly software should ignore repetitious fragments without giving up on the packet.

Since fragments will be sent briskly, the reassembly procedure can give up on a partially reassembled packet if no additional fragments for it arrive within a few seconds.

2.4. Datalink Layer Retransmission

For each unicast ARCNET packet, the hardware indicates to the sender whether or not the receiver acknowledged the packet. To improve reliability, datalink implementations are encouraged to retransmit unacknowledged packets or packet fragments. Several retransmissions may be necessary. Broadcast packets, however, are never acknowledged and, therefore, they should never be retransmitted.

Packets which are successfully received may not be successfully acknowledged. Consequently, retransmission by the datalink implementation can cause duplicate packets or duplicate fragments. Duplicate packets are not a problem for IP or ARP. As mentioned in the previous section, ARCNET reassembly support should ignore any redundant fragments.

3. Transmitting IP and ARP Datagrams

IP and ARP datagrams are carried in the client data area of ARCNET packets. Datalink support places each datagram in an appropriate size ARCNET frame, fragmenting IP datagrams larger than 504 octets into multiple frames as described in the previous section.

4. IP Address Mappings

This section explains how each of the three basic 32-bit internet address types are mapped to 8-bit ARCNET addresses.

4.1. Unicast Addresses

A unicast IP address is mapped to an 8-bit ARCNET address using ARP as specified in [2]. A later section covers the specific values which should be used in ARP packets sent on ARCNET networks.

It is possible to assign IP addresses such that the last eight bits are the same as the 8-bit ARCNET address. This would allow direct mapping of IP address to ARCNET address without using a discovery protocol. Some implementations might provide this as an option, but it is not recommended practice. Although such hard-wired mapping is initially appealing, experience shows that ARP is a much more flexible and convenient approach which has a very small cost.

4.2. Broadcast Addresses

All IP broadcast addresses must be mapped to the ARCNET broadcast address of 0.

Unlike unicast packets, ARCNET does not attempt to insure delivery of broadcast packets, so they may be lost. This will not have a major impact on IP since neither IP nor ARP expect all packets to be delivered.

4.3. Multicast Addresses

Since ARCNET provides no support for multicasts, all IP multicast addresses must be mapped to the ARCNET broadcast address of 0.

5. ARP

The hardware address length is 1 octet for ARP packets sent over ARCNET networks. The ARP hardware type for ARCNET is 7. ARP request packets are broadcast by directing them to ARCNET broadcast address, which is 0.

6. RARP

Reverse Address Resolution Protocol [6] packets can also be transmitted over ARCNET. For the purposes of datalink transmission and reception, RARP is identical to ARP and can be handled the same way. There are a few differences to notice, however, between RARP when running over ARCNET, which has a one octet hardware address, and Ethernet, which has a six octet hardware address.

First, there are only 255 different hardware addresses for any given ARCNET while there’s an very large number of possible Ethernet addresses. Second, ARCNET hardware addresses are more likely to be duplicated on different ARCNET networks; Ethernet hardware addresses will normally be globally unique. Third, an ARCNET hardware address is not as constant as an Ethernet address: ARCNET hardware addresses are set by switches, not fixed in ROM as they are on Ethernet.

7. Maximum Transmission Unit

The maximum IP packet length possible using this encapsulation method is 60,480 octets. Since this length is impractical, all ARCNET implementations on a given ARCNET network will need to agree on a smaller value. Therefore, the maximum packet size MUST be configurable in implementations of this specification.

In any case, implementations must be able to send and receive IP datagrams up to 576 octets in length, and are strongly encouraged to handle IP datagrams up to 1500 octets in length.

Implementations may accept arriving IP datagrams which are larger than their configured maximum transmission unit. They are not required to discard such datagrams.

To minimize the amount of ARCNET fragmentation, implementations may want to aim at an optimum IP packet size of 504 bytes. This avoids the overhead of datalink fragmentation, but at the expense of increasing the number of IP packets which must be handled by each node in the path. In addition to encouraging local applications to generate smaller packets, an implementation might also use the TCP maximum segment size option to indicate a desire for 464 octet TCP segments [7], or it might announce an IP MTU of 504 octets through an MTU discovery mechanism such as [8]. These would inform non-ARCNET nodes of the smaller optimum packet size.

8. Assigned Numbers

Datapoint Corporation assigns ARCNET protocol IDs to identify different protocols running on the same ARCNET medium. For implementations of this specification, Datapoint has assigned 212 decimal to IP, 213 decimal to ARP, and 214 decimal to RARP. These are not the numbers assigned to the IP encapsulation defined by RFC 1051 [5]. Implementations of RFC 1051 can exist on the same ARCNET as implementations of this specification, although the two would not be able to communicate with each other.

The Internet Assigned Numbers Authority (IANA) assigns ARP hardware type values. It has assigned ARCNET the ARP hardware type of 7 [9].

Acknowledgements

Several people have reviewed this specification and provided useful input. I'd like to thank Wesley Hardell at Datapoint and Troy Thomas at Novell's Provo office for helping me figure out ARCNET. In addition, I particularly appreciate the effort by James VanBokkelen at FTP Software who picked on me until all the fuzzy edges were

smoothed out.

The pioneering work in transmitting IP traffic on ARCNET networks was done by Philippe Prindeville.

References

[1] Postel, J., "Internet Protocol", RFC 791, DARPA, September 1981.

[2] Plummer, D., "An Ethernet Address Resolution Protocol", RFC 826, MIT, November 1982.

[3] Datapoint, Corp., "ARCNET Designer's Handbook", Document Number 61610, 2nd Edition, Datapoint Corporation, 1988.

[4] Novell, Inc., "ARCNET Packet Header Definition Standard", Novell, Inc., November 1989.

[5] Prindeville, P., "A Standard for the Transmission of IP Datagrams and ARP Packets over ARCNET Networks", RFC 1051, McGill University, March 1988.

[6] Finlayson, R., Mann, T., Mogul, J., and M. Theimer, "A Reverse Address Resolution Protocol", RFC 903, Stanford, June 1984.

[7] Postel, J., "Transmission Control Protocol", RFC 793, DARPA, September 1981.

[8] Mogul, J., Kent, C., Partridge, C., and K. McCloghrie, "IP MTU Discovery Options", RFC 1063, DEC, BBN, TWG, July 1988.

[9] Reynolds, J., and J. Postel, "Assigned Numbers", RFC 1060, USC/Information Sciences Institute, March 1990.

Security Considerations

Security issues are not discussed in this memo.

Author's Address

Don Provan
Novell, Inc.
2180 Fortune Drive
San Jose, California, 95131

Phone: (408) 473-8440
EMail: donp@Novell.Com