INTERNET-DRAFT
Expires May 27, 1996

Carl Marcinik
FORE Systems,Inc.
Maryann Perez Maher
ISI
November 22, 1995

Distributed ATMARP Service in Classical IP over ATM Networks

<draft-marcinik-ipatm-dist-atmarp-00.txt>

Status of this Memo

Abstract

One of the basic limitations of the ATMARP service model specified in RFC 1577 [LAUB94] is the requirement that only one ATMARP server be utilized to provide the address resolution service for a given LIS. Besides introducing a single-point-of-failure into a LIS, this model also presents obvious scaling issues. A proposal was put forth in "Classical IP and ARP over ATM Update" [LAUB95] to resolve these shortcomings through the introduction of a model supporting multiple ATMARP servers that maintain fully-replicated, synchronized address resolution databases. This model necessitates a certain amount of complexity introduced by the addition of a server database synchronization protocol. This synchronization protocol also requires additional packet types and formats as well as associated semantics. It is felt, however, that a fully-replicated, synchronized database scheme is not required to provide a reasonably robust ATMARP service that addresses the limitations of the basic model.

This memo describes an alternative scheme that provides a distributed ATMARP service for a Classical IP LIS by introducing simple,

straight-forward extensions to the basic model described in RFC 1577 [LAUB94]. An arbitrary number of ATMARP servers cooperate (when required) to resolve client address resolution requests in a manner that both distributes the client load among the cooperating servers and supports reasonable scaling properties. Rather than a synchronization protocol, this scheme proposes a few simple mechanisms to help maintain cache consistency between distributed ATMARP servers. Moreover, these require no additional (In)ATMARP packet types and only very minor changes to the existing packet formats and semantics (the reserved bit in the type and length fields is utilized, otherwise the formats remain unchanged). Finally, this scheme supports both the ATMARP client autoconfiguration mechanism proposed in [MAR95] as well as non-autoconfiguring (i.e., UNI 3.0/3.1) clients. Backward compatibility is also provided for existing (i.e., unmodified) RFC1577 clients. Aside from the difference in approach outlined here for supporting multiple ARP servers, this memo (for the most part) assumes the procedures contained in [LAUB95]. Other differences, where they might exist, will be explicitly stated.

1. Terminology

This memo introduces a few terms that have not been used in the context of previous discussions concerning ATMARP. These terms are briefly defined here for the sake of clarity.

First, this memo makes a distinction between "connected" and "unconnected" ATMARP clients. In this context, a "connected" client is a client that (persistently) maintains a connection to its ATMARP server. An "unconnected" client only uses a connection to the ATMARP server to register its address resolution information and then releases it.

The term "directly-registered" is used to refer to a client's registration status relative to a given server. A client is only permitted to register its address resolution information with one server in the LIS. This server is referred to as the client's "designated" server. The client determines its "designated" server either through manual configuration (and associated procedures) or through autoconfiguration [MAR95]. When a client registers its address information with its "designated" server, the client is said to be "directly-registered" with that server regardless of whether or not the client maintains a connection to its ATMARP server (i.e., is "connected" or "unconnected"). Clients not "directly-registered" with a particular server are termed "nondirectly-registered" clients relative to that server.

Finally, cache information (pertaining to a client's IP/ATM address mapping), maintained by either a client or server, is categorized as being "authoritative" or "non-authoritative." From a server perspective, "authoritative" cache information refers to cache

entries maintained by the server for "directly-registered" clients. A server's cache entries for "nondirectly-registered" clients are considered "non-authoritative." From a client perspective, cache entries are considered "authoritative" only when explicitly indicated as such by the server from which they are obtained. Otherwise they are considered "non-authoritative."

2. Introduction

The model proposed in this memo consists of an arbitrary number distributed ATMARP servers interconnected by means of a set of point-to-multipoint (PMP) connections. Servers accept point-to-point

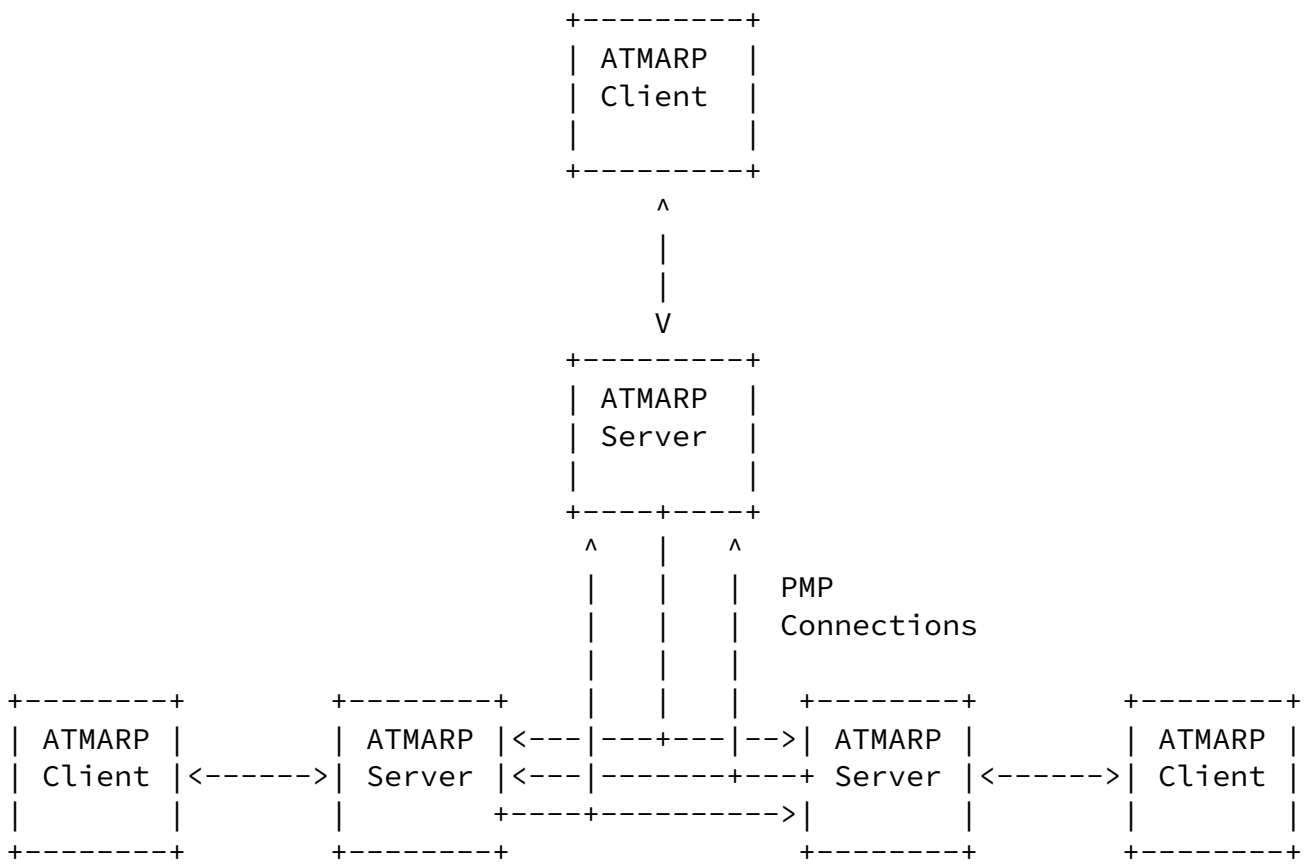

Figure 1 Distributed ATMARP Model

connections from both autconfiguring [MAR95] and non-autoconfiguring ATMARP clients. The clients utilize their point-to-point connections to register their address resolution (i.e., IP address/ATM address mapping) information with their designated server. Subsequently, clients perform address resolution operations to obtain mapping information (which they cache locally) for destinations throughout the LIS. Servers use their PMP connections to distribute client mapping information among all the ATMARP servers serving the LIS. Each server caches this mapping information but may also query all servers simultaneously to obtain mapping information needed to resolve a client's request when the information is not available in

its local cache. A representative example of this model is shown in Figure 1.

While the server connection topology employed here is perhaps more constrained than that proposed in [LAUB95], it is simpler to maintain and may be more amenible to possible server autoconfiguration mechanisms (see Section X). Although [LAUB95] permits an arbitrary server connection topology, creating such a topology is accomplished entirely through manual configuration. It is felt that supporting an arbitrary server topology might provide some benefit, however, to be most useful such a scheme would require additional protocol mechanisms. These mechanisms would be needed to maintain the server topology in the face of dynamic topology changes. Such mechanisms would result in considerable cost, in terms of protocol complexity (e.g., loop detection, partition repair, etc.), and are felt to be too expensive to warrant this approach. However, even if it were deemed necessary or desirable to provide such support, the basic model described here could be employed as long as server packets could be forwarded to, and received from, each of the other cooperating servers in the LIS.

3. Overview

A client must register its address resolution information upon its initialization using the applicable registration procedures in RFC 1577 [LAUB94] or in [LAUB95]. Included in the client registration procedure is need for the client to locate its designated server. This memo supports the ATMARP client autconfiguration proposal described in [MAR95] which essentially allows a client to locate and connect to its designated ATMARP server by utilizing a LIS-specific anycast address (determined a priori). While this mode of client operation is preferred, this scheme also supports non-autoconfiguring clients as well (e.g., a client supporting only UNI 3.0/UNI 3.1 signalling capability). A non-autoconfiguring client must be administratively configured with the ATM address of each ATMARP server in the LIS. The client randomly selects its designated server and then establishes a connection to it. Utilizing either of these procedures, clients are distributed to cooperating servers throughout the LIS. The client registers its address resolution information with its server using the previously established connection. The server, in turn, reliably distributes the client's mapping information to all cooperating servers in the LIS using its (outgoing) point-to-multipoint connection (replys are returned on the incoming VCs associated with the other servers' leaf nodes) Each server caches the client information which is updated from time to time when the client re-registers as part of its refresh operations.

Of particular interest is the means by which address resolution operations are carried out by the scheme proposed in this memo. Servers attempt to resolve client address resolution requests from their local caches, if possible. If a server cannot resolve a request

with mapping information contained in its own cache, it forwards the request to all other servers and awaits a response. A server may elect to solicit a reply for either authoritative or non-authoritative cache information. In the authoritative case, only the server with a directly-registered client matching the target IP address in the request may reply. In the non-authoritative case, any (and every) server that contains the sought after cache information may reply. The approach utilizing an authoritative reply requires the least amount of overhead and minimizes the propagation of stale cache information to other servers and clients in the LIS.

The proposal put forth in [LAUB95] requires that any server in a LIS be able to resolve an address resolution request received from any client that is a member of the same LIS. The way in which this is achieved is by ensuring that every server fully replicates all cache entries for all the clients in the LIS. The scheme proposed here aims to achieve the same goal (i.e., any server can resolve a client's request), but does so in a slightly different manner. Instead of "guaranteeing" (with a high degree of probability) that each server will have the cache entry needed to resolve a given client request, this scheme "guarantees" (with a high degree of probability) that, if a client's designated server cannot resolve the request from its own cache, it can obtain the information needed from another server. The scheme, therefore, approximates a fully-replicated database from a client perspective, but uses fewer and simpler mechanisms to do so.

The goal is to be able to satisfy any client request at any server without allowing a period of non-connectivity to exist for any client. For unconnected clients, the possibility exists that a designated server could lose its client mapping information for directly-registered clients (e.g., due to a server crash). In such a case, a designated server will not be able to resolve a request for such clients until those clients refresh their address resolution information. This scheme allows servers with non-authoritative cache information (however, only cache information previously obtained from an authoritative source) to answer a request for such information. Since client information for each client is reliably propagated to all other servers when a client registers, we can "guarantee" with a high degree of probability that another server on the LIS has the information needed. The probability increases as the number of servers in the LIS increases.

As intimated above, there are two possible schemes that may be employed for address resolution operations discussed in this memo. Each will interoperate with the other so servers are free to implement either one. If a server receives an address resolution request that it cannot resolve from its own cache, it basically has two choices: it can forward a request for either authoritative cache information or non-authoritative cache information to all the other servers in the LIS. Authoritative cache information is desirable, but the requesting server cannot be absolutely sure that it can be

obtained. It is more certain, however, that it can obtain non-authoritative information. Just to be clear: only the server with cache information for a directly-registered client may respond to a request for authoritative cache information. However, any server (including the server holding the authoritative answer) may respond to a request for non-authoritative information if it can provide the answer with cache information obtained from an authoritative source (e.g., a server forwarding address resolution information as a result of a client's registration)

These two choices lead to two strategies, the first being: Try to obtain an authoritative reply. If this is unsuccessful, try to obtain a non-authoritative reply. If this is unsuccessful, then the address resolution request fails. This strategy leads to less stale cache information in the network, but could occasionally suffer from a delayed response to the client. However, viewed from an amortized perspective, this may be the most preferred strategy.

The other strategy is obvious: Try to obtain non-authoritative information first. If the initial request fails to produce any responses (not very likely) within a short, bounded period of time, retry the request a small number of times. If this is unsuccessful, then the address resolution request fails. This strategy will produce an answer with less delay in the case where there is no authoritative cache information in the LIS (e.g., due to a server crash). However, this will also propagate more stale cache information throughout the LIS. This is somewhat offset by requiring both servers and clients to impose a more strict limit on the lifetime on cache information obtained from a non-authoritative source. Furthermore, servers are only permitted to return cache information obtained from an authoritative source in non-authoritative replys. Finally, unconnected clients are required to refresh their address resolution information more often (perhaps as much as three times the rate of a connected client, e.g., once every 5 minutes).

The remaining sections in this memo focus on specifying the details of this scheme. Section 4 presents the details of client registration and the associated server procedures. Section 5 focuses on the details of address resolution operations for both clients and servers. Section 6 discusses server failure and recovery. Section 7 presents the minor modifications required to the ATMARP packet formats to support this scheme.

4. ATMARP Registration Procedures

4.1 ATMARP Server Startup Procedures

Before any client registrations can be accepted, each server must establish an outgoing point-to-multipoint connection, with each of the cooperating ATMARP servers in the LIS as leaves, using the applicable procedures outlined in RFC 1755 [PEREZ95] or UNI 4.0

[ATMF95]. This is necessary so that all client registration information can be propagated to all other servers as clients begin to register. Currently this scheme requires that the identity of cooperating servers be determined by examining a list of individual ATM addresses administratively configured at each server. Each server must be identically configured with regard to the number and value of the ATM addresses supported with the obvious exception that the server, itself, is not included in the list. While a server autoconfiguration procedure would be desirable (and may be specified in a future version of this or another memo; see Section X), all servers must support administrative configuration of all other cooperating servers in the LIS.

A failure to successfully add a server to the existing point-to-multipoint call should be continuously retried, backing off each time until a reasonable "polling" rate is achieved. A persistent failure to add a server may indicate a configuration error and therefore should be reported through the server's management facility. During this period of time, a server must also accept incoming point-to-multipoint calls from other servers so that it may be added as a leaf on PMP calls originating from other servers. A server may reject incoming point-to-multipoint call attempts if the calling address contained in the setup message does not match any of the ATM addresses of its configured servers. After all servers have been added to the server's own outgoing point-to-multipoint call (notwithstanding signalling problems for certain parties), servers must be prepared to accept incoming calls from clients that have identified this server as being their designated server. Servers supporting the ATMARP client autoconfiguration procedure, must also perform the required server initialization outlined in [MAR95].

4.2 ATMARP Client Startup Procedures

In order to facilitate server initialization operations, ATMARP clients should delay the start of the client registration procedure by a random period of time of between TBD and TBD seconds. Note, however, that an RFC 1577 client will not necessarily delay the start of its registration procedure. Therefore, an RFC 1577 client may initially experience setup failures if the network is in a mode where all ATMARP servers (and in particular its own server) are simultaneously initializing.

Once this period of random delay has elapsed, a client must determine its designated server. For autoconfiguring clients, this procedure is outlined in [MAR95]. For non-autoconfiguring clients supporting this scheme, this is accomplished by randomly choosing an address from its list of administratively configured server ATM addresses.

Each client supporting this procedure, whether capable of autoconfiguration operation or not, must provide a means of being administratively configured with the ATM addresses of all the

cooperating servers in the LIS. No ordering or priority is necessarily to be imposed on the configured server addresses. The default behavior of a non-autoconfiguring client is to choose a server address at random as its designated server. However, all implementations must support the capability to override this default behavior so that an ordering or priority among the configured addresses can be obtained. Each client must support the capability to be administratively configured with the maximum number of servers supported for a LIS. This number is TBD. RFC 1577 clients may be configured with any of the individual ATM addresses of any server on the LIS as its atm$arp-req address.

4.3 ATMARP Client Registration Procedures

Client registration procedures may be initiated as part of a client's initial startup sequence or due to the necessity of the client to refresh its address resolution information (see section 4.5). Once a client determines its designated server, it must establish a connection to that server and register its address resolution information. For autoconfiguring clients this is accomplished by utilizing a LIS-specific anycast address and following the procedures outlined in [MAR95] and [LAUB95]. A non-autoconfiguring client supporting this scheme must follow the procedures outlined in [LAUB95]. RFC 1577 clients follow the procedures outlined in [LAUB94]. Note that clients based on [LAUB95] will use an ATMARP_Request with both the source and target addresses set to its own address resolution information to register itself, while RFC 1577 client registration will now occur implicitly.

If a non-RFC 1577 client's initial attempt to connect to its designated server fails, it is suggested that the client use a simple exponential backoff algorithm before retrying the setup. This may be necessary to avoid a "setup" storm that might occur during periodic client refresh operations or perhaps during a server failure. In no case, however, should a client allow the cumulative time spent trying to establish a connection to its designated server to exceed TBD seconds. This is required to support an upper bound on the amount of time that occurs before a client initiates recovery operations (see Section 6).

It is necessary for a client to wait until its registration information has been reliably propagated to all the cooperating servers in the LIS (that are currently operable) before it receives a reply. Non-RFC 1577 clients must allow, at a minimum, TBD seconds for this to be accomplished. If this period of time elapses before a reply is received, the client should retry the registration. If the registration is still not successful after TBD attempts, the client initiates recovery operations.

4.4 ATMARP Server Registration Procedures

4.4.1 Considerations

The main goal of the server registration procedures is to reliably distribute a client's address resolution information to all the cooperating servers in the LIS so that such information may be cached. Among the issues that must be considered in accomplishing this goal is how (if at all) the client's address resolution information is related to other information already cached throughout the LIS. It is imperative that, as a result of the server registration procedure, inconsistencies not be introduced into the servers' caches. In particular, three cases stand out as areas of concern:

1. A client is attempting to register an IP address that is already cached in the LIS, but with a different ATM address and an open connection is associated with the cache entry

2. A client is attempting to register an IP address that is already cached in the LIS, but with a different ATM address and no open connection is associated with the cache entry

3. A client is attempting to register an IP address not currently cached in the LIS, however, the client's ATM address is associated with another IP address that is already cached

The first case will normally occur as the result of a pair of mis-configured hosts. In a single server environment, this situation is easily detected by performing a "duplicate IP address" test. The results of this test could be used to deny registration to the requesting client, if the client's IP address proved to be a duplicate. However, performing this test, and denying service if it fails, is no easy task in a distributed environment. This would seem to require a two phase "lock and commit" database type of operation to be performed across all the servers in the LIS. Since we obviously don't know ahead of time if a particular client registration is a duplicate or not, we would need to check every client registration. Since a client is required to re-register periodically, the more clients we add to the LIS, the more often we would be performing this check. This would lead to a great deal of overhead. Another consideration here is, although it is bound to happen, the occurrence of this event is expected to be very low.

The second case would be most likely to occur as a result of a client's physical connection being moved to another UNI, probably located on a switch other than it was connected to when it last registered. The action of moving the physical cable would have released the connection to the ATMARP server and caused the mapping between the IP address/ATM address to change. This action would have

also necessitated that the client reconnect and re-register with the ARP server. It is claimed that this case is most likely to explain the second case because a connection is not currently associated with the cache entry. However, this case could be equivalent to the first case if the previously registered client did not maintain a connection to the ATMARP server. Nevertheless, it is felt that more times than not, a "change" operation has occurred in this case. Therefore, rather than deny registration we would want this new mapping to take effect throughout the LIS as rapidly as possible.

The third case is not very likely to happen given the wide acceptance and implementation of the ILMI address registration procedure [ATM95]. Most ATM address configuration is removed from human hands and so we are unlikely to end up with this case.

Given the above considerations, it seems that the server registration procedure should be optimized for the second case. While the first case is cause for concern, it is not expected to occur as often as the second. We propose to adopt a policy of "last registration always overrides an earlier registration" as opposed to that supported by the policy of the "detection and denial" model which is "first registration always wins." In the absence of any specific security mechanisms, the "last registration overrides" policy is just as valid as denying a registration. Furthermore, this policy is much simpler to implement in a distributed scheme and simplifies the processing when a duplicate entry is discovered at a server other than the one where a registration originated. If the first case were to occur and persist, then it is likely that this scheme will thrash about waiting for something or someone to correct the problem. However, this behavior is not so much different than what has been observed with other address resolution protocols.

4.4.2 Procedures

Server registration procedures are initiated upon receiving an ATMARP_Request packet from a client with the source and target IP addresses equal ("explicit" case) or an ATMARP_Request packet is received from a client and the server does not have a corresponding cache entry (implicit case).

After updating (or creating) its local cache entry according to the policy described above, the server must forward the client's address resolution reliably to each of the other servers in the LIS. This is accomplished by forwarding a Server ATMARP_Request, containing the client's mapping information, over the sever's outgoing point-to-multipoint VC and awaiting for a Server ARP_Reply reply from each server. Note that if a timeout occurs while waiting for a reply, resending the request will cause all servers to receive the request again (unless perhaps one or more of the servers missed the previous one).

When a server receives a Server ATMARP_Request, it creates or updates its local cache entry according to the policy described above. After the entry is updated, the server forwards an ATMARP_Reply to the server from which it was received.

After the initiating server receives ATMARP_Replys from all the servers, it sends an ATMARP_Reply to the client (explicitly or implicitly) initiating the registration.

4.5 Client Refresh Procedures

TBD

5. Address Resolution Procedures

5.1 Client Address Resolution Procedures

ATMARP clients attempt to resolve IP addresses to ATM addresses by examining their local address resolution cache. If a client's address resolution cache does not contain an entry matching the sought after IP address, it must send an ATMARP_Request to its designated server in order to obtain the desired mapping information as outlined in the applicable procedures [LAUB94, LAUB95] and as augmented by this memo. The scheme presented here is completely backward compatible with RFC 1577 clients. There are only minor differences for non-RFC 1577 clients.

After sending an ATMARP_Request to its designated server, the client awaits for a corresponding ATMARP_Reply. An ATMARP_Reply returned to an RFC 1577 client is handled as outlined in [LAUB94]. However, an ATMARP_Reply returned to a non-RFC 1577 client may require slightly different handling. This is due to the possibility that the mapping information returned in the reply may have been obtained second hand. In other words, the client's designated server may have not been able to resolve the request from its own cache and therefore would have had to obtain the sought after information from another server. Depending on the strategy employed, the information returned to the requesting server (i.e., the requesting client's designated server) may have been non-authoritative. If this is were the case, then a more restrictive limit must be used on the lifetime of the mapping information to be cached by the client. This is necessary to avoid having stale cache information persist in the client's cache thus reducing that the possibility that the client will experience any period of non-connectivity due to such stale cache information. ([LAUB95] requires a client to purge a cache entry when a failure to connect to another client occurs. Following this, the client sends an ATMARP_Request to try to obtain fresh cache information. The caching scheme employed in this memo attempts to limit this type of activity.)

For non-RFC 1577 clients, the NON-AUTHORITATIVE bit (see section X)

is set when second hand cache information is returned to the client. This is an indication to the client that it needs to use the restricted cache timeout value when completing the cache entry. This value is defined as TBD seconds. If the NON-AUTHORITATIVE bit is clear in a reply received by a non-RFC 1577 client, then normal packet handling procedures apply. For backward compatibility with RFC 1577 clients, the NON-AUTHORITATIVE bit is never set so such clients are more susceptible to the issues discussed above. [Most RFC 1577 implementations were modified to accommodate the "non-connectivity because of stale cache information" problem so most of these implementations should eventually recover from this condition.]

Because of the differences between the distributed ATMARP scheme presented in this memo and the approach outlined in [LAUB95], the handling of an ARP_Nak is slightly different and actually results in significantly simpler client behavior. In [LAUB95] the receipt of an ARP_Nak requires the client to query another ATMARP server in an attempt to resolve its request. In fact, in the worst case, the client would need to query all the address resolution servers in the LIS. This approach complicates the client behavior and significantly increases the delay required to resolve a request (in the worst case). The approach taken in this memo essentially retains the original ARP_Nak semantics as defined in RFC 1577 [LAUB94]. Because the scheme presented here ensures that all servers in the LIS are queried (simultaneously) if a designated server does not contain a cache entry matching the target IP address in a request, it can provide the answer in a definitive manner as to whether the information exists anywhere in the LIS. This approach, therefore, maintains characteristics similar to a single ATMARP server for this case as well as the normal case.

Non-RFC 1577 clients must allow at least TBD seconds for the ATMARP server to return a reply to its request. If a timeout occurs, the client should retry the request a small number of times. If a reply is still not obtained, the client may initiate recovery procedures (see Section 6). RFC 1577 behavior remains the same for this case.

5.2 Server Address Resolution Procedures

Server address resolution operations are initiated upon the receipt of an ATMARP_Request from a client or from another server. In the case of an ATMARP_Request received from a client, a (designated) server attempts to resolve the request from entries contained in its local address resolution cache. If an entry is found, the sought after information is extracted and returned to the client in an ATMARP_Reply. Before returning a reply, however, the server must perform a series of checks to determine if it should set the NON-AUTHORITATIVE bit in the ATMARP_Reply

If the client request can be satisfied with cache information from (another) directly-registered client, then no further considerations

are necessary and the ATMARP_Reply is returned to requesting client with the NON-AUTHORITATIVE bit clear. However, if the server is holding a cache entry that can satisfy the request, but the entry does not correspond to a directly-registered client, then the server must determine how this cache information was obtained. If the cache entry was created from an authoritative reply, or from a relayed client registration, then the ATMARP_Reply is returned with the NON-AUTHORITATIVE bit clear. If, however, the cache entry was created from a non-authoritative reply, then the type of requesting client must be determined, if known. This information is normally recorded at the time the client registered. If requesting client registered implicitly, then for backward compatibility reasons the NON-AUTHORITATIVE bit cannot be set. However, if the client is known to be a non-RFC 1577 client then the NON-AUTHORITATIVE bit must be set so that the appropriate client handling procedures can be performed.

If the server is not holding a cache entry matching the requested IP address, then it must query all the other servers in the LIS for the answer. The sequence used to perform this query is somewhat dependent on the strategy employed by the server. Two interoperable strategies were outlined in section 2. For the sake of brevity only the first strategy will be detailed here.

A server sends a Server ATMARP_Request (see section 7) over its outgoing point-to-multipoint VC to forward the request to all other servers in the LIS. The Server ATMARP_Request carries the server's address information in the source address fields, and the IP address to be resolved in the target protocol address field. To indicate a request for authoritative cache information (in keeping with the goals of the first strategy), both the REGISTRATION and the NON-AUTHORITATIVE bits are clear.

A server receiving a Server ATMARP_Request for authoritative cache information examines its cache to determine if it has a matching entry. If a cache entry was found, and if the entry corresponds to a directly-registered client, the server forms a Server ATMARP_Reply using the information found in the entry and returns the reply to the requesting server. Otherwise, the server discards the request. [Would it be better to have servers send NAKs if they do not find authoritative information?] When the requesting server receives the Server ARP_Reply, it completes its cache entry and forwards the reply on to the requesting client.

6. Server Failure Detection and Recovery

Depending on how an ATMARP server failure occurs, the client may or may not immediately detect its failure. If a client maintains a connection to the ARP server then it probably has a better chance of detecting a problem sooner (again, depending on the failure). In any event, the client will probably initiate recovery operations when it repeatedly fails to connect to its designated server or when the

client fails to receive an expected reply from the server after several retry attempts.

Note, if the client still has a connection established to the server (e.g., the timeout case), then it must release it. When its connection to the ATMARP server is released, the client must attempt to reconnect and re-register its ATMARP information. If the client is non- autoconfiguring then a new server address should be selected at random (ensuring that a different address is indeed selected). For autoconfiguring clients, anycast will "search" for the new server (presumably, the failed server's anycast address will eventually be de-registered by an ILMI connectivity timeout if' nothing else). During the period of time that it is setting up a call, the client may continue to use its cache to resolve addresses. When the client succeeds in establishing a new connection to its (new) server, and registers its address resolution information, it may then resolve any pending requests.

Notice that when a client re-registers (as above) the (new) server will record the client as being "directly-registered." Once a failed server is returned to service, it is necessary that the clients that have failed over to the (new) server, return to the original server. This happens automatically in this scheme when the clients refresh their address resolution information with the server. As part of a modified refresh operation, the clients must release their current connection and setup a new call. For autoconfiguring clients this is done using the appropriate anycast address; non-autoconfiguring clients choose a new server address at random from those it has configured.

7. Packet Formats

TBD

8. VC Resource Requirements

One consideration for this scheme is its use of VC resources. The requirements in terms of VCs for the server-server case is N (N - 1) incoming VCs [the leafs associated with the (N - 1) incoming PMP connections per server, where N = the number of servers in a LIS] and N outgoing VCs for the root of each PMP call to each of the (N - 1) other servers. So the total VC requirements for server-server interaction are N**2 VCs per LIS. Each server would as well need to support an additional M/N (with M = the number of clients in the LIS) bidirectional VCs for client-server interaction (assuming clients are distributed uniformly across multiple servers). So the total VC requirements for each server would then be N + M/N VCs and N**2 + N (M/N) = N**2 + M VCs for all servers in a LIS.

9. Acknowledgments

The authors would like to thank Mike Kazar (FORE Systems) for useful feedback concerning some of the cache issues discussed in this memo.

10. References

[LAUB94] Laubach, M., "Classical IP and ARP over ATM", RFC 1577, Hewlett-Packard Laboratories, January 1994.

[LAUB95] Laubach, M., "Classical IP and ARP over ATM Update", draft-ietf-ipatm-classic2-00.txt, Internet Draft (work in progress), August 1995.

[ATMF95] ATM Forum, "ATM User-Network Interface (UNI) Signalling Specification Version 4.0", ATM Forum, December 1995.

[MAR95] Marcinik, C., Liaw, F., "ATMARP Client Autoconfiguration", draft-marcinik-ipatm-auto-arp-00.txt, Internet Draft (work in progress), November 1995.

[PEREZ95] Perez, M., Liaw, F., Mankin, A., Hoffman, E., Grossman, D. and A. Malis, "ATM Signaling Support for IP over ATM", RFC 1755, ISI, Fore, Motorola Codex, Ascom Timeplex, February 1995.

11. Security Considerations

Security issues are not addressed in this memo.

12. Authors' Addresses

Carl Marcinik
FORE Systems, Inc.
Pittsburgh Office and Research Park
Pittsburgh, PA 15237-5829

Phone: (412) 625-9051
Email: carlm@fore.com

Maryann Perez Maher
USC/Information Sciences Institute
4350 N. Fairfax Drive Suite 400
Arlington, VA 22203

Phone: 703-807-0132
EMail: perez@isi.edu