The ATM Forum

Technical Committee

ATM Name System V2.0

AF-DANS-0152.000

July, 2000

The ATM Forum
Worldwide Headquarters
2570 West El Camino Real, Suite 304
Mountain View, CA 94040-1313
Tel: +1-650-949-6700
Fax: +1-650-949-6705

Preface

The editor thanks participants of the Directory and Name Services Working Group, the Routing and Addressing Working Group, the Security Working Group, and the Service Aspects and Applications Working Group for their many discussions of this specification. The editor also thanks the following people for their written contributions:

Glenn Curtis
Kim Doo-Seok
Bob Epley
Jim Harford
Joann J. Ordille
Thomas A. Shaver
Albert C. Wan

Finally, the editor thanks the working group chairs who oversaw the process of bringing this specification to completion:

Joann J. Ordille, Chair, Directory and Name Services Working Group
Ather J. Chaudhry, Chair, Routing and Addressing Working Group
Mickey Spiegel, Immediate Past Chair, Routing and Addressing Working Group
Richard Graveman, Chair, Security Working Group
Bahman Mobasser, Chair, Service Aspects and Applications Working Group

This specification uses three levels for indicating the degree of compliance necessary for specific functions, procedures, or coding. They are indicated by the use of key words as follows:

- **Requirement:** "Shall" indicates a required function, procedure, or coding necessary for compliance. The word "shall" used in text indicates a conditional requirement when the operation described is dependent on whether or not an objective or option is chosen.

- **Objective:** "Should" indicates an objective which is not required for compliance, but which is considered desirable.

- **Option:** "May" indicates an optional operation without implying a desirability of one operation over another. That is, it identifies an operation that is allowed while still maintaining compliance.

Contents

Figures

Tables

1 Introduction

Asynchronous Transfer Mode (ATM) applications require various types of directory services. Directory services can be universal, that is, used by more than one application, or they may be application specific. An example of a universal directory service is finding an ATM address corresponding to the name of an ATM end system. An example of an application specific service is finding the providers of specific types of services (for example, video on demand servers).

The ATM Name System (ANS) is a system for storing and retrieving mappings between names and a small set of defined objects. Primarily, ANS supports mapping names to ATM addresses and ATM addresses to names. It can also map an ATM address to another ATM address, names to public keys and names to text strings.

ANS is an extension to the Internet Engineering Task Force's (IETF) Domain Name System (DNS) [RFC 1034]. ANS specifies objects that can be used to translate names to and from ATM addresses whether a client submits a translation request to an ANS or DNS server. ANS specifies the native ATM application for ATM clients. ANS uses an Application Program Interface (API) whose semantics are specified in [AF-SAA-0108.000] to gain access to the ATM network via ATM Forum User Network Interface (UNI) Signaling. The ANS record formats and DNS functions are intended to be independent of the version of the underlying ATM signaling. See [AF-SAA-0108.000] for information on the versions of UNI signaling that are applicable to ANS.

ANS provides limited support for application specific services. If a group of service providers follow a naming convention for their services (for example, the video on demand servers are always named "video.service-provider.com"), then users can find these video on demand servers by translating names that follow the convention.

1.1 Capabilities

ANS V2.0 has the following major capabilities:

- Name to address translation for discovering the location of services in a Switched Virtual Circuit (SVC) environment
- ATM address to name translation for discovering the domain name that corresponds to an ATM address
- ATM address to ATM address mapping for discovering one ATM address given another (for example, for discovering an E.164 address given an ATM End System Address (AESA))
- ATM address to Internet Protocol (IP) address mapping for discovering the IP address for dual hosts
- IP to ATM address mapping for discovering the ATM address for dual hosts
- Discovery of the ATM Terminating Interface (ATI) addresses that serve a particular ATM Destination Point (ADP) in support of ATM bi-level addressing

- Secure dynamic registration and update of ATM addresses for a given host or service
- Secure dynamic registration and update of ATI addresses for a given ADP address
- Query responses that can be authenticated
- Distribution of public keys

1.2 Limitations

ANS V2.0 does not support the following:

- Storing and retrieving higher level protocol connection attributes such as the information carried in the Broadband High Layer Information (BHLI) information element
- Storing the attributes of services and querying for service information based on those attributes
- Storing service profiles for users
- Distributing X.509 style public key certificates [ITU-T X.509]

These limitations could be addressed by an adaptation of the IETF Lightweight Directory Access Protocol (LDAP) system [RFC2251] to the ATM environment.

2 Acronyms

This specification uses the following acronyms:

AAL	ATM Adaptation Layer
ADP	ATM Destination Point
AESA	ATM End System Address
AFI	Authority and Format Identifier
API	Application Program Interface
ATI	ATM Terminating Interface
ATM	Asynchronous Transfer Mode
ATMA	ATM Address
ANS	ATM Name System
BCOB	Broadband Connection Oriented Bearer
BHLI	Broadband High Layer Information
CLP	Cell Loss Priority
CPCS-SDU	Common Part Convergence Sublayer – Service Data Unit
ESI	End System Identifier
HO-DSP	High-Order Domain Specific Part
IDI	Initial Domain Identifier
IETF	Internet Engineering Task Force
ILMI	Integrated Local Management Interface
IN	Internet Address Space
IP	Internet Protocol
IPv4	Internet Protocol Version 4

LDAP	Lightweight Directory Access Protocol
MIB	Management Information Base
OUI	Organizationally Unique Identifier
PTR	Pointer
QOS	Quality of Service
RD	Recursion Desired
RR	Resource Record
SEL	Selector
SSCS	Service Specific Convergence Sublayer
SVC	Switched Virtual Circuit
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UNI	User Network Interface
VC	Virtual Circuit

3 ANS Overview

The ANS is based on the Domain Name System (DNS) specified by the Internet Engineering Task Force (IETF) and widely deployed in IP networks. The DNS includes a structure for distributed control and storage of naming data in a network, and techniques for retrieving the naming data efficiently from its storage location. ANS is an enhanced DNS that can store and retrieve mappings for names to ATM addresses and vice versa for ATM clients. This section provides a basic understanding of the ANS for first-time readers of this specification. A more complete understanding comes from reading the associated Domain Name System documents.

In ANS, names are hierarchically structured in much the same way that filenames are hierarchically structured in most operating systems. Figure 1 shows part of an ANS name space. The name space is structured as a tree. Each interior node of the tree is called a domain, and it is said to "contain" the subtree rooted at that node. The full name of any node in the tree is constructed by concatenating the names, separated by dots ".", of all the nodes on the path from that node to the root of the tree. For example, the name "MYHOST.MYCO.COM" has the name of the host, "MYHOST", followed by the name of the company "MYCO", followed by a name that groups companies "COM".

Each name (i.e. node) in the ANS can be associated with data in the form of resource records (RRs). ANS enhances DNS by supplying one new resource record, the ATM Address Resource Record (ATMA RR), which specifies how to encode AESA and E.164 addresses in the name space. Figure 1 depicts an ATMA RR for "MYHOST.MYCO.COM". The resource record says that the address is in the Internet Address Space (IN) with an ATM Address (ATMA) which is an AESA starting with "39.046F". Figure 1 also depicts the reverse mapping for translating this AESA back to the name "MYHOST.MYCO.COM". The reverse mapping uses the established DNS PTR RR for pointing to an existing name, and an ANS convention for name space structure that places reverse mappings for AESAs in a domain called "AESA.ATMA.INT".

ATM addresses are divided into components and encoded in ANS ATMA RRs according to rules specified in Section 4.3. A domain name is generated from the ATM address by reversing these components and listing them from most to least specific.

The data in the ANS name space can be distributed by "cutting" one or more links in the ANS tree. All

nodes between the cuts are called a “zone,” and are stored together on a server. Copies of a zone can be replicated on other servers, by a process called “zone transfer,” to increase performance and reliability. The new notification and incremental transfer DNS specifications [RFC1995, RFC1996] provide for more timely and efficient communication of zone changes. ANS stores information about the location of zones, and that information is used to find all the necessary components of the distributed name space for answering a query. It is typical for small organizations to maintain at least two servers for the zone that contains the organization’s naming domain. Larger organizations may divide their naming domain into several zones and use more than two servers to support their zones.

The technique of distributing parts of the name space to different servers might lead one to think that each query contacts many servers and that query processing is, therefore, expensive. This is not necessarily the case. ANS query processing software can cache information about the structure of the name space, so that a query for information about a name can often be answered by contacting directly the server that stores data about the name. The software can also cache the results of previous queries, so repeat queries for the same name can be answered from the cache without contacting any additional servers.

There are several possible configurations for query processing in DNS. ANS servers can answer queries by recursive processing, that is, by contacting all the necessary servers for their clients. ANS servers can also answer queries by providing a referral to the next server that a client must contact in processing the query. By default, ANS clients request and ANS servers perform recursive processing (see Section 7).

The following example illustrates how recursive processing and caching can be combined to reduce the expense of answering queries. Let Server A store information for the root and “.COM” nodes in Figure 1 and Server B store the subtree rooted at “MYCO.COM”. Note that any node, for example the root or “.COM”, includes information about the location of its children nodes. A client wishes to locate a user address for the name “MYHOST.MYCO.COM” in Figure 1. The client submits a query with that name to an ANS server. If the cache is empty and the name is not within the scope of the server, the server begins processing the query by contacting Server A which has information about the root of the name space. When Server A determines that the information for “MYCO.COM” is stored in Server B, it returns the information necessary for contacting Server B. The original server then contacts Server B and returns the answer. While processing the query, the original server caches the location of Server B and the answer to the query. If other queries for “MYCO.COM” are submitted to the server, it contacts Server B directly. If queries for a name are repeated, the server answers them from its cache.

The DNS specifications provide more overview [RFC1034] and configuration information [Section 2.2, RFC 1035]. The ATM Forum’s user guide on addressing provides background information on ATM addressing and useful comparisons with IP addressing [AF-RA-0105.000].

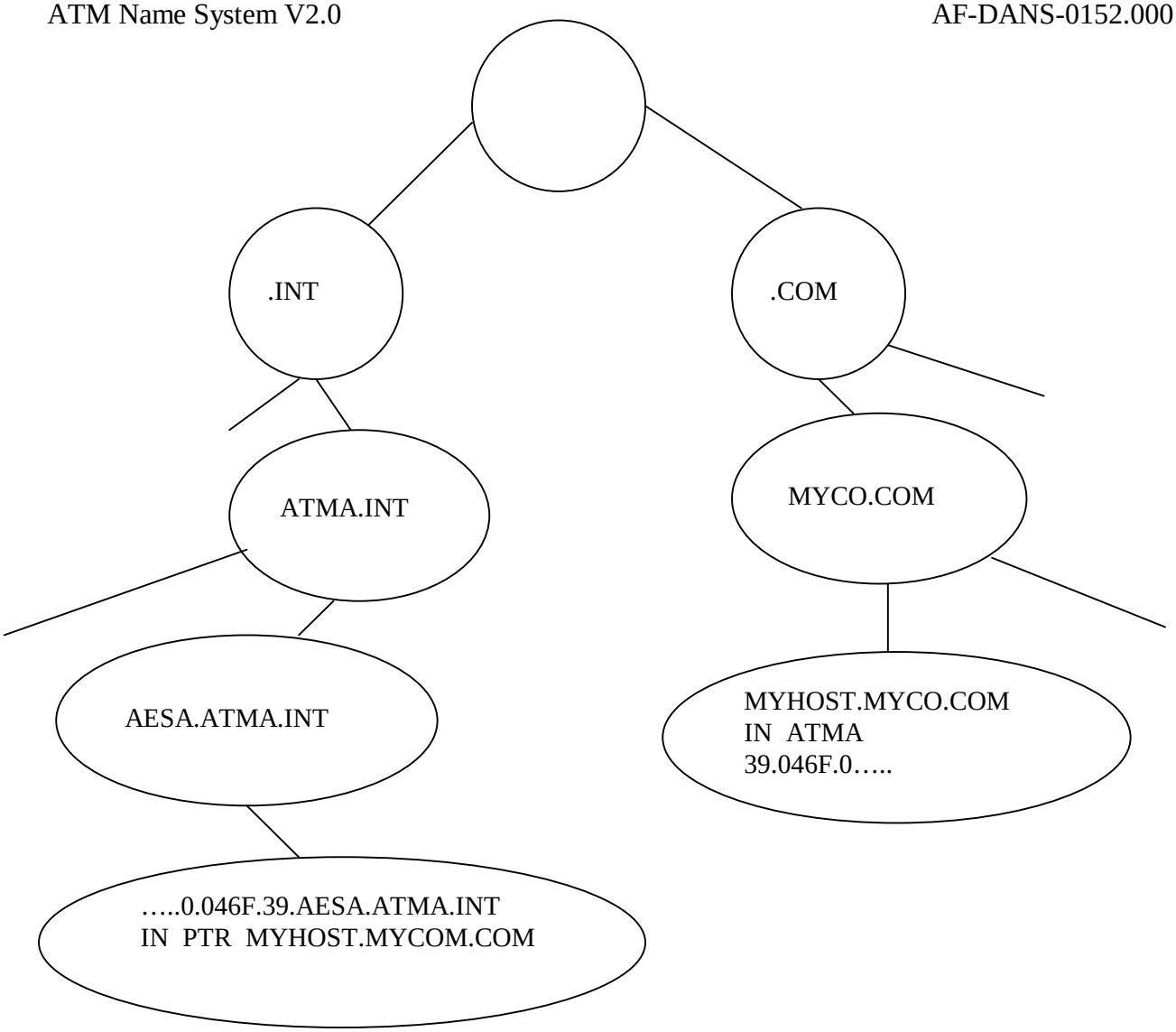

Figure 1: Partial ANS Name Space

The resource record describing the mapping of the domain name `MYHOST.MYCO.COM` to an AESA beginning with `"39.046F"` is depicted, along with the record containing the reverse mapping of the AESA to the name. The record for the reverse mapping has the AESA encoded in the domain name ending in `"0.046F.39.AESA.ATMA.INT"`.

Key

IN	Internet Address Space	***.COM**	Various domain names
ATMA	ATM Address Resource Record	***.INT**	Various domain names
PTR	Pointer to another domain name		

3.1 Relevant IETF Documents

ANS is based on the following Domain Name System documents, which are available from the Internet Engineering Task Force at http://www.ietf.org/:

[RFC 1034] Domain Names -- Concepts and Facilities

[RFC 1035] Domain Names – Implementation and Specification

[RFC 1995] Incremental Zone Transfer in DNS

[RFC 1996] A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)

[RFC 2065] Domain Name System Security Extensions

[RFC 2136] Dynamic Updates in the Domain Name System (DNS UPDATE)

[RFC 2137] Secure Domain Name System Dynamic Update

3.2 Levels of Support for ANS and Interoperability with DNS

Minimally, an ANS shall conform to [RFC 1034], [RFC 1035] and this specification. To support dynamic update, an ANS shall also support the functionality of [RFC 2065], [RFC 2136], and [RFC 2137]. An ANS supporting dynamic update should also support [RFC 1995] and [RFC 1996] to provide timely propagation of changes.

ANS is designed to co-exist in the same global name space with DNS. ANS servers shall support both the ATM and IP protocols [see Section 5]. DNS servers may not support the ANS ATMA record type. In this case, they do not store ATM addresses in resource records in their zone. However, such DNS servers will respond to requests and updates for the ATMA record type with the appropriate negative answers. A DNS server can support the processing of ATMA RRs by forwarding requests and responses related to these records even if the server itself does not store ATMA RRs.

DNS and ANS servers together form a fully connected name space. They each can supply useful addresses that allow both types of servers, ANS and DNS, to reach the other servers in the name space. While DNS servers process client requests via the IP protocols, ANS servers shall process client requests via ATM or IP protocols.

4 Database Elements

4.1 Resource Record Definitions

ATM specific resource records (RR) conform to the top level RR format and semantics as defined in

Section 3.2.1 of [RFC1035] and belong to the RR CLASS Internet (IN).

4.2 ATM Specific Resource Records

The following sections describe the database elements that allow ATM clients to:

1. Map a name to an ATM address (ATM Address Resource Record – ATMA RR)
2. Map an ATM address to a name (Pointer Resource Record – PTR RR)
3. Map an ADP address to ATI addresses (ATMA and Wildcard RRs)

Other ATM specific resource records can be added in future releases of this specification.

4.3 ATM Address (ATMA) Resource Record

The ATM Address Resource Record, ATMA RR, is used to map domain names to ATM addresses. The ATMA RR is defined with the mnemonic "ATMA" and TYPE code 34 (decimal). ATM address lookup is analogous to IP address lookup. A query is generated by the resolver requesting ATMA RRs for the provided domain name.

ATMA RR's have the RDATA format shown in Figure 2.

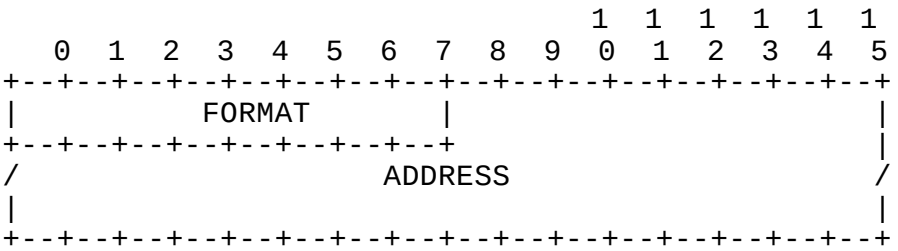

Figure 2: ATMA RDATA Format

The fields in Figure 2 have the following meaning:

- FORMAT: One octet that indicates the format of ADDRESS. The two possible values for FORMAT are value 0 indicating ATM End System Address (AESA) format and value 1 indicating E.164 format.

- ADDRESS: Variable length string of octets containing the ATM address of the node to which this RR pertains.

When the format value is 0, indicating that the address is in AESA format, the address is coded as described in ISO 8348/AD 2 using the preferred binary encoding of the ISO NSAP format. When the format value is 1, indicating that the address is in E.164 format, the Address/Number Digits appear in the order in which they would be entered on a numeric keypad. Digits are coded in IA5 characters with the leftmost bit of each digit set to 0. This ATM address appears in ATM End System Address Octets field

(AESA format) or the Address/Number Digits field (E.164 format) of the Called party number information element [AF-SIG-0061.000]. Subaddress information is intentionally not included because E.164 subaddress information is used for routing.

ATMA RRs cause no additional section processing.

4.4 ATM Address to Domain Name Mapping

The PTR RR is defined in [RFC1035]. This RR is typically used under the "IN-ADDR.ARPA" domain to map from IPv4 addresses to domain names.

Similarly, the PTR RR is used to map from ATM addresses to domain names under the "ATMA.INT" domain. A domain name is generated from the ATM address according to the rules described below. The client sends a query requesting a PTR RR for the provided domain name.

A domain name is generated from an AESA formatted ATM address by representing the components for the AESA as hexadecimal character strings, then concatenating from left to right the following substrings and separating them by a "." character from each other:

- the Selector (SEL) field
- the End System Identifier (ESI) field
- the digits of the High-Order Domain Specific Part (HO-DSP) field in reverse order and separated by a "." character from each other
- the Initial Domain Identifier (IDI) field with the following exception for the E.164 addresses
- the digits of the international E.164 number after the initial zero padding has been removed in reverse order and separated by a "." character from each other
- the hexadecimal 'f' that terminates the E.164 address in the AESA
- the Authority and Format Identifier (AFI) field
- the top level subdomain "AESA.ATMA.INT."

For example, the domain name used in the reverse lookup for the AESA

`39246f000e7c9c0312000100010000 1234567800`

would appear as

`00.000012345678.1.0.0.0.1.0.0.0.2.1.3.0.c.9.c.7.e.0.0.0.246f.39.AESA.ATMA.INT.`

For example, the following ATM E.164 AESA, which contains the international E.164 number +49 89 722 3257,

```
450000498972232 57f01020304AABBCCDDEEFF00
```

would appear as

```
00.AABBCCDDEEFF.4.0.3.0.2.0.1.0.f.7.5.2.3.2.2.7.9.8.9.4.45.AESA.ATMA.INT
```

A domain name is generated from a native E.164 formatted ATM address by converting it to an embedded E.164 AESA and following the rules described above. An E.164 is embedded in an AESA by:

- Setting the AFI to 45
- Filling the IDI field with a Binary Coded Decimal (BCD) form of the E.164 address. See [AF-RA-0106.000] for details.
- Setting the HO-DSP, ESI and SEL fields to 0

For example, the native E.164 number +49 89 722 3257 is encoded as an embedded E.164 number in the following AESA:

```
450000498972232 57f000000000000000000000
```

It would be represented for reverse lookups by the following domain name:

```
00.000000000000.0.0.0.0.0.0.0.0.f.7.5.2.3.2.2.7.9.8.9.4.45.AESA.ATMA.INT
```

Implementation note: For sanity's sake user interfaces should be designed to allow users to enter ATM addresses using their natural order, that is, as they are typically written on paper. Also, arbitrary “.”s should be allowed (and ignored) on input.

4.5 ATM Destination Point to ATM Terminating Interface Mapping

ATM bi-level addressing uses two addresses: an ATM Terminating Interface (ATI) address to select the interface from a routing network to a destination network, and an ATM Destination Point (ADP) address to select the end system in the destination network [BI-LEVEL].

ANS supports bi-level addressing by recording the set of ATM Terminating Interface (ATI) addresses associated with each ATM Destination Point (ADP) address. The set of ATI addresses for a particular ADP is the set of addresses of network interfaces through which the destination address of the user is accessible. Since groups of ADP addresses are typically on the same network and accessible through the same set of ATI addresses, wildcard resource records (wildcard RRs) are used to group related ADP addresses with their associated ATI addresses.

Wildcard RR processing is defined in [RFC1034]. Wildcard RRs are only processed when no other RRs match the query, so the parts of the “AESA.ATMA.INT” domain that are used to translate ATM addresses to host names cannot also be used to translate ATM Destination Point (ADP) addresses to ATM Terminating Interface (ATI) addresses. The domain “ATI.ATMA.INT” is used to map ADP addresses to ATI addresses using the ATMA RR.

A domain name is generated from the ADP address according to the rules described in Section 4.4, but with

a top-level domain of "ATI.ATMA.INT" instead of "AESA.ATMA.INT." A client sends a query to request an ATMA RR for the provided domain name. The results of the query are the ATI addresses associated with the ADP address.

The "ATI.ATMA.INT" domain contains wildcard RRs that group ADP addresses with the same prefix and the same ATI addresses together under one wildcarded domain name. Example wildcard RRs for ADP AESA addresses beginning with

```
39246f000e7c9c
```

and their associated ATI addresses represented by P and Q are:

```
*.c.9.c.7.e.0.0.0.246f.39.ATI.ATMA.INT. IN ATMA P
*.c.9.c.7.e.0.0.0.246f.39.ATI.ATMA.INT. IN ATMA Q
```

The following ADP address matches these wildcard RRs:

```
39246f000e7c9c031200010001000001234567800
```

A query for this address will return both P and Q as the ATI addresses.

Note that the wildcarded domain name that matches the longest prefix of the ADP address in the query is the one used to return RRs for the query. The following wildcard RR also matches the ADP address above, but it matches a shorter prefix of that ADP address and would be ignored in processing queries for that address:

```
*.0.0.0.246f.39.ATI.ATMA.INT. IN ATMA S
```

Although the ADP addresses represented by prefixes in the "ATI.ATMA.INT" domain can be associated with a host name, there is no requirement that ADP addresses be assigned host names in the ANS.

4.6 Example Master File Format

This section describes an example master file format. The master file is typically used to load database records into a master ANS server. Note that other formats are possible.

The format of ATMA RRs in Master Files conforms to Section 5, "Master Files," of [RFC1035].

The RDATA section of an ATMA RR line in a master file is expressed as follows:

- AESA formatted ATM address: a string of hexadecimal digits. A "." character can be used to separate any two digits for readability. The string is case insensitive. For example:

  ```
  39.246f.00.0e7c9c.0312.0001.0001.000012345678.00
  ```

- E.164 formatted ATM address: a "+" character followed by a string of decimal digits that form an international E.164 number. A "." character can be used to separate any two digits for readability. For example:

```
+358.400.1234567
```

In Figure 3 below are examples of the use of ATMA and PTR RRs in Master Files to support name to ATM address, ATM address to name mapping, and ATM Destination Point address to ATM Terminating Interface address mapping.

```
;;;;;;
;;;;;; Master File for domain data.example.com.
;;;;;;

$ORIGIN data.example.com.

@      IN    SOA    name1.data.example.com.  name4.data.example.com. (
                            1994041800   ; Serial  - date
                            1800         ; Refresh - 30 minutes
                            300          ; Retry   - 5 minutes
                            604800       ; Expire  - 7 days
                            3600 )       ; Minimum - 1 hour
       IN    NS     name1.data.example.com.
       IN    NS     ns.example.com.
;
salmon IN    ATMA   39.246f.000e7c9c031200010001.000012345678.00
char   IN    ATMA   39.246f.000e7c9c031200010001.000023456789.00

;;;;;;
;;;;;; Master File for reverse mapping of ATM addresses under the
;;;;;; prefix 39.246f.000e7c9c031200010001
;;;;;;

$ORIGIN 1.0.0.0.1.0.0.0.2.1.3.0.c.9.c.7.e.0.0.0.246f.39.AESA.atma.int.

@      IN    SOA    name1.data.example.com.  name4.data.example.com. (
                            1994041800   ; Serial  - date
                            1800         ; Refresh - 30 minutes
                            300          ; Retry   - 5 minutes
                            604800       ; Expire  - 7 days
                            3600 )       ; Minimum - 1 hour
      IN    NS     name1.data.example.com.
      IN    NS     ns.example.com.
;
00.000012345678      IN      PTR     salmon.data.example.com.
00.000012345679      IN      PTR     char.data.example.com.



;;;;;;
;;;;;; Master File for reverse mapping of ADP addresses under the
;;;;;; prefix 39.246f.000e7c9c031200010001 to the ATI address
;;;;;; 39.840f.8001bc72031200010001.000087654321.00
;;;;;;

$ORIGIN 1.0.0.0.1.0.0.0.2.1.3.0.c.9.c.7.e.0.0.0.246f.39.ATI.atma.int.


@      IN    SOA    name1.data.example.com.  name4.data.example.com. (
                            1994041800   ; Serial  - date
                            1800         ; Refresh - 30 minutes
                            300          ; Retry   - 5 minutes
                            604800       ; Expire  - 7 days
                            3600 )       ; Minimum - 1 hour
      IN    NS     name1.data.example.com.
      IN    NS     ns.example.com.
;
*     IN   ATMA 39.840f.8001bc72031200010001.000087654321.00
```

Figure 3: Example Master File Format

5 Protocol Transport

The DNS assumes that protocol messages will be transmitted as datagrams or in a byte stream carried by a virtual circuit. Datagrams are preferred for queries and responses due to their lower overhead. They are also preferred for any update functions that fit into a datagram.

Since ATM networks do not support datagram services, ANS queries and responses shall be carried over ATM virtual connections established between the clients and the servers. The ATM virtual connection uses AAL 5, and clients are responsible for establishing a retransmission policy for lost AAL5 messages that considers prior performance statistics to avoid overloading an already loaded server with retransmitted queries.

ANS servers may need to perform gateway functions to IP networks to obtain IP address information or to transit part of the name space. For this reason, ANS Servers shall support ATM Adaptation Layer 5 (AAL 5), User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) protocols in ANS Version 2.0. Zone transfers between two servers require TCP. Incremental zone transfers, notifications or dynamic update operations are performed according to the datagram or virtual circuit protocol requirements of the appropriate IETF standard (see Section 3.1).

After learning the ATM address of one or more ANS servers, an ANS client may setup a connection to any one of the servers for sending ANS requests and receiving ANS responses. The client sets up and releases the connection dynamically using the UNI signaling protocol accessed through an API whose semantics are specified by [AF-SAA-0108.000]. The client should time-out after some period of inactivity and release this connection. The server may release an inactive connection so that the server can provide services to other clients. Neither the server nor the client shall attribute any special significance to a disconnection indication. Similarly, ANS servers should time-out idle connections between themselves to free resources for others.

6 Client Initialization

When a client becomes operational, initially it needs to find the ATM addresses of one or more ANS servers. ANS clients use one of two mechanisms to locate the ATM address of ANS servers:

- Get the ANS server addresses via ILMI.
- Use a well-known ANS Address.

A client can use ILMI to retrieve an ANS server address from the atmfSrvcRegAns object in the Service Registry Management Information Base (MIB) [AF-ILMI-0065.000].

If an ANS server address cannot be obtained from ILMI or if the client is unable to establish a virtual circuit (VC) to any Integrated Local Management Interface (ILMI) supplied server address, then a well-known address may be used. The well-known address is

```
C50079000000000000000000.00A03E000002.00.
```

The security extensions to ANS [RFC 2065] support verification of the authorization and currency of information read from the ANS. The information that can be verified includes the public keys of other ANS clients. Security-aware clients, who conform to RFC 2065, require knowledge of the public key of at least one ANS zone to use the security extensions. Such clients can be configured with the public key of an ANS zone using the boot file format described in Section 6.2 of RFC 2065. The public key shall be obtained from a trusted source in a secure manner (for example, from an enterprise administrator by direct personal communication).

Secure dynamic update of the ANS [RFC 2137] allows security-aware clients to update entries that they own in the ANS. Clients performing updates shall be configured with a private key that corresponds to a public key recorded for the client in the ANS. The security of that private key from discovery by entities other than the authorized client is essential to the proper functioning of secure dynamic update.

7 Recursive Processing

By default, ANS clients, when they make queries, request recursive processing of the query (by setting the Recursion Desired (RD) bit to 1 in the header of the query). ANS name servers shall support recursion. If the server gives the client a referral to another server, then it can happen that the client does not have ATM connectivity to the next server. ANS clients that can communicate over ATM and IP networks, can optionally request referral processing.

8 Native ATM Services Interface

The ANS client and ANS server access the ATM network via an API whose semantics are specified by [AF-SAA-0108.000]. The Native ATM Services connection attributes used for communications between an ANS client and an ANS server are shown in the Table 1 below.

UNI IE	IE Attribute	Attribute Value
AAL Parameters	AAL Type	AAL type 5
	Forward Maximum CPCS-SDU Size	512 bytes
	Backward Maximum CPCS-SDU Size	512 bytes
	SSCS Type	Null
ATM Traffic Descriptor	Forward Peak Cell Rate (CLP=0+1)	Line rate in cells per second
	Backward Peak Cell Rate (CLP=0+1)	Line rate in cells per second
	Best Effort Indicator	Best effort requested
Broadband Bearer Capability	Bearer Class	Class X (BCOB-X)
	Traffic Type	No indication
	Timing Requirements	No indication
	Susceptibility to Clipping	Not susceptible to clipping
	User Plane Connection Configuration	Point-to-point connection
Broadband High Layer Information	High Layer Information Type	Vendor-specific application ID
	High Layer Information	The OUI (BHLI octets 6-8) is 0x00A03E. The application ID (BHLI octets 9-12) is 0x00000001.
Called Party Number	Type of Number, Addressing Plan	Either private or public address format
	Address	ATM address of ANS server
Called Party Subaddress	Type of Number, Addressing Plan	Private subaddress format, if required
	Address	ATM subaddress of ANS server, if required
Quality of Service Parameter	QoS Class Forward	QoS class 0
	QoS Class Backward	QoS class 0
N/A (Local Service)	SSCS	Null

Table 1: Native ATM Services Connection Attributes

The following elements form a SAP address for the ANS server:

- ATM address, including the selector byte
- ANS application identification (carried in the BHLI information element)

Note that both layer-2 protocol identification and layer-3 protocol identification are ABSENT.

9 ANS API

This section specifies an API that exposes ANS services on the client machine. The API herein is a *semantic* API; it is purposely specified at an abstract level that allows independence from a particular operating system or language binding. It is expected that implementations of *syntax* APIs (e.g. BSD Sockets, Winsock2, XTI) will provide more concrete details for application programmers wishing to access ANS services on a particular platform.

Note that the style of the primitives herein suggests a blocking operating system environment. That is, processor control is not returned to the requesting application until the request has been completely processed and the query results are available. This decision was based purely on ease of expression within this specification. Alternatively, a syntax API might be designed in which the results of the queries are obtained via polling by the requesting application. Also, the results of queries might be returned to the requesting application by asynchronous messages.

Define data type ATM_endstation (
ATM_address_format,
ATM_address
)

where

- *ATM_address_format* is the format of the *ATM_address* field (i.e. AESA or E.164).
- *ATM_address* specifies the address of an ATM endstation.

9.1 ANS_get_ATM_address Request

Purpose: request translation of an ATM name to one or more ATM addresses.

ANS_get_ATM_address (

IN *ATM_name,*

OUT *ATM_endstation_list*

)

where

- *ATM_name* specifies the ATM entity that is the object of the translation query.
- *ATM_endstation_list* specifies a list of ATM endstations as the answer to the translation query. Each item within the list is of data type *ATM_endstation*.

Return Values: ANS_SUCCESS
ANS_NOT_FOUND
ANS_SERVICE_DOWN
ANS_INVALID_PARAMETER

9.2 ANS_get_ATM_name Request

Purpose: request translation of an ATM address to an ATM name.

ANS_get_ATM_name (
IN *ATM_endstation,*

OUT *ATM_name*

)

where

- *ATM_endstation* specifies the ATM entity that is the object of the translation query.
- *ATM_name* specifies the answer to the translation query.

Return Values: ANS_SUCCESS
ANS_NOT_FOUND
ANS_SERVICE_DOWN
ANS_INVALID_PARAMETER

9.3 ANS_get_ATM_ati Request

Purpose: request translation of an ATM address to an ATM address, for support of bi-level addressing.

ANS_get_ATM_ati (

IN	*ATM_endstation_adp,*	// ATM destination point
OUT	*ATM_endstation_ati_list*	// ATM terminating interface

)

where

- *ATM_endstation_adp* specifies the ATM entity that is the object of the translation query. This item is of type *ATM_endstation*.
- *ATM_endstation_ati_list* specifies a list of ATM terminating interface addresses that answer the translation query. Each item within the list is of data type *ATM_endstation*.

Return Values: ANS_SUCCESS
ANS_NOT_FOUND
ANS_SERVICE_DOWN
ANS_INVALID_PARAMETER

9.4 ANS_get_public_key Request

Purpose: request translation of an ATM name to a public key.

ANS_get_public_key (

IN	*ATM_name,*
OUT	*public_key_list,*
OUT	*signature*

)

where

- *ATM_name* specifies the ATM entity that is the object of the translation query.
- *public_key_list* specifies a list of public keys that answer the translation

query.

- *signature* is the result of an algorithm applied to *public_key* by some trusted entity, whereby an ANS client can rely on the integrity of *public_key*. There shall be some defined value for the case of an ANS server not supplying a signature. The policies and mechanisms to provide security for ANS transactions are beyond the scope of this specification.

Return Values: ANS_SUCCESS
ANS_NOT_FOUND
ANS_SERVICE_DOWN
ANS_INVALID_PARAMETER

9.5 ANS_get_text_string Request

Purpose: request translation of an ATM name to a text string.

ANS_get_text_string (
IN *ATM_name***,**
OUT *text_string*
)

where

- *ATM_name* specifies the ATM entity that is the object of the translation query.
- *text_string* specifies the answer to the translation query.

Return Values: ANS_SUCCESS
ANS_NOT_FOUND
ANS_SERVICE_DOWN
ANS_INVALID_PARAMETER

---

10 References

10.1 Normative

af-ilmi-0065.000, Integrated Local Management Interface (ILMI) Specificaton, v4.0, (September 1996).

af-ra-0106.000, ATM Forum Addressing: Reference Guide, v1.0, (January 1999).

af-saa-0108-000, Native ATM Services: Semantic Description, v1.0, (February 1996).

af-sig-0061.000, ATM User-Network Interface (UNI) Specification, v4.0, (July 1996).

RFC 1034, Domain Names - Concepts and Facilities, (November 1987).

RFC 1035, Domain Names - Implementation and Specification, (November 1987).

RFC 1995, Incremental Zone Transfer in DNS, (August 1996).

RFC 1996, A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY), (August 1996).

RFC 2065, Domain Name System Security Extensions, (January 1997).

RFC 2136, Dynamic Updates in the Domain Name System (DNS UPDATE), (April 1997).

RFC 2137, Secure Domain Name System Dynamic Update, (April 1997).

10.2 Informative

ITU-T Recommendation X.509, The Directory: Authentication Framework, (1993).

RFC 2251, Lightweight Directory Access Protocol, v3, (December 1997).

af-ra-0105.000, ATM Forum Addressing: User Guide, v1.0, (January 1999).

BI-LEVEL (replace with af-ra- ????.000), ATM Bi-Level Addressing, (2000). *(Note to editor: Work on bi-level addressing is now in straw ballot. Please update this reference when that document is approved.)*

Appendix I: Requirements (Informative)

This Appendix does not form an integral part of this specification.

This informative appendix captures the requirements generated by the Service Aspects and Applications/Directory Services Working Group that created ANS V1.0. It has been updated to reflect the enhancements supplied by ANS V2.0.

I.1 Basing of ANS on DNS

There are many reasons to use DNS as the basis for ANS.

- DNS is widely implemented in end systems.
- DNS domain registration procedures have been implemented in most countries.
- DNS domain names are well known by the user community.
- DNS can scale well if the name space is properly assigned. This satisfies the scalability requirement given in I.2.
- DNS implementations are efficient in terms of CPU and memory usage. This satisfies the efficiency requirement given in I.2.
- DNS is resilient via the secondary server mechanism. This satisfies the no single point of failure requirement given in I.2.
- DNS has low latency of queries. This satisfies the low latency requirement given in I.2
- DNS information can be queried and updated securely.
- DNS can be easily augmented with new database objects for ATM.
- DNS allows dual hosts to choose between IP and native ATM.

I.2 ANS requirements

Several requirements for the directory service function that apply to ANS are:

- No single point of failure (resilience)
- Efficient implementation including storage efficiency
- Hierarchical naming structure (scalability)
- Low latency of queries
- Tolerance for temporary inconsistency, that is, that such inconsistency does not damage the information in the ANS
- The service may be transiently unavailable, for example, “a busy signal”

Other ANS requirements include:

- Client independence from non-ATM network and transport layer protocols

  ANS is to be used by ATM applications. These applications may not necessarily implement non-ATM

network or transport protocols. This document specifies the operation of DNS protocols using native ATM services.

- Use of an existing naming scheme

  ANS should exploit an already existing naming scheme in the same way as ATM addressing utilizes existing addressing schemes. Otherwise the ATM Forum would need to set up a new, worldwide name registration service, which would be a difficult task to manage and operate.

- Dynamic updating of the directory data base

  When an ATM end system address changes, it is important that the ANS database need not be manually reconfigured. The end system should be able to dynamically check and update its address information in the ANS database. This is especially important when the end user does not even necessarily know that an ATM address of the end system has changed. (For example, a manager of the switch could change the address prefix without notifying the users.)

- Security of the dynamic updating process

  Dynamic updating of the ANS database by end systems usually requires that the updates can be authenticated. ANS must therefore include an option to sign the ANS database entries with cryptographic digital signatures. It is also desirable that the ANS store authenticated public keys in its database in order to make the ANS protocol independent of yet another public key management protocol.

- Automatic configuration of ANS clients

  Finally, automatic configuration of ANS clients is an important requirement. In order to use directory services, the clients need to know one or more ATM addresses of one or more ANS servers. If security is implemented, an ANS client needs a correct public key of at least one zone of the hierarchical name space. A natural solution is that ANS clients learn this information dynamically from an ATM configuration server.

Appendix II: Usage Scenarios (Informative)

This Appendix does not form an integral part of this specification.

This informative appendix provides examples of ANS operation. An ANS query specifies a target domain name, the IN class and the type of resource records to return. In this informative appendix we specify such a query by:

```
QUERY   domain-name   record-type
```

`QUERY` returns the resource records with type `record-type` for the domain entry called `domain-name`.

An ANS update specifies a target zone of the ANS, a set of prerequisite resource records whose existence is tested before the update occurs, a set of resource records to add to the zone, and a set of resource records to delete from the zone. Security is provided by signing the entire update request, including the DNS header but excluding other request signatures, with the private key of the owner of the records updated. If there is more than one owner for the records, all must sign the update request.

A simplified description of some possible operations follows:

```
ADD   zone-name  exists   rr   sig
```

If `exits` is true, `ADD` adds the resource record `rr` to the zone `zone-name` provided at least one resource record for the domain name specified in `rr` exists. If `exists` is false, the `rr` is only added if no resource records for the domain name in `rr` exist.

```
DELETE zone-name rr sig
```

`DELETE` deletes the resource record `rr` from the zone `zone-name`.

```
MODIFY   zone-name   old-rr   new-rr   sig
```

`MODIFY` modifies the resource record `old-rr` in the zone `zone-name` to have the values in the resource record `new-rr`. This is done by adding `new-rr` and deleting `old-rr` in one transaction.

In each case, the primary ANS server for the zone verifies the digital signature `sig` on the update before making any changes. See the relevant RFCs for more information on ANS operations [RFC1034, RFC2065, RFC2136, RFC2137].

The following usage scenarios are based on the partial ANS name space shown in Figure II-1. This ANS name space includes the domain called `MYCO.COM` for a company called "MYCO." It includes the domain called `AESA.ATMA.INT` for mapping ATM addresses to domain names, and the domain called `ATI.ATMA.INT` for mapping ADP addresses to ATI addresses. It also includes the domain called `IN-ADDR.ARPA` for mapping IP addresses to domain names. The resource records for a system called `SONGS.MYCO.COM` are given. These include an AESA in an `ATMA RR`, an E.164 address in an

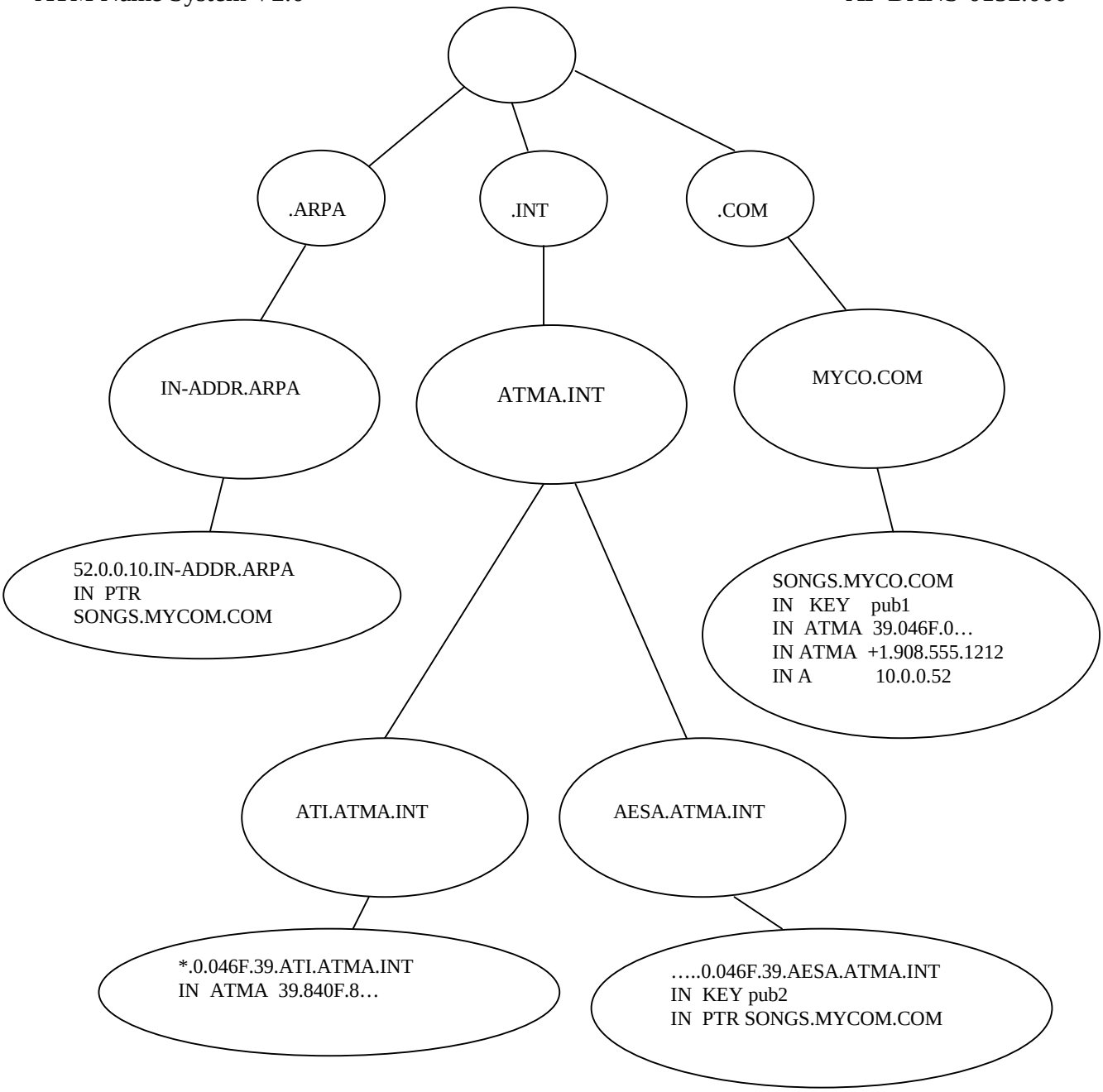

Figure II-1: Partial ANS Name Space for ATM and IP Addresses

The resource records for `SONGS.MYCO.COM` are depicted. These include an AESA beginning with `"39.046F"`, the E.164 address `"+1.908.555.1212"`, the IP address `"10.0.0.52"`, and the public key `"pub1"`. The records containing the reverse mapping of the AESA to the name in the `AESA.ATMA.INT` domain and of the IP address to the name in `IN-ADDR.ARPA` domain are also depicted. The record beginning with a `"*"` is a wildcarded `RR` for mapping the ADP addresses beginning with `"39.046F.0"` to their associated ATI address.

Key

IN	Internet Address Space	**A**	IP Address Resource Record
ATMA	ATM Address Resource Record	**PTR**	Pointer to another domain name
KEY	Public Key Resource Record		

ATMA RR, an IP address in an A RR, and a public key in a KEY RR. The domain name "…0.046F.39.AESA.ATMA.INT" has resource records for the reverse mapping from the AESA for SONGS.MYCO.COM to the domain name for SONGS.MYCO.COM. It also has a resource record for the public key that enables users to change the resource records for this domain name. The resource record for the reverse mapping from the IP address for SONGS.MYCO.COM to the domain name for SONGS.MYCO.COM is supplied. Finally, the wildcarded ATMA resource record for mapping ADP addresses with the same prefix as the AESA for SONGS.MYCO.COM to the AESA for its associated ATI is supplied. In the following discussion, the AESA abbreviated by "39.046F.0…" and the abbreviated AESA encoded in "…0.046F.39.AESA.ATMA.INT" are the same AESA.

II.1 Translating a Name to an ATM Address

If a client wants to obtain the ATM address for SONGS.MYCO.COM, the client submits the following query to the ANS:

QUERY SONGS.MYCO.COM ATMA

The client receives two ATM addresses as answers: the AESA "39.046F.0…" and the E.164 address "+1.908.555.1212".

II.2 Translating a Given ATM End System Address to Another ATM Address

A client knows the AESA "39.046F.0…" for an end system, and wants to obtain an E.164 address for the end system if one exists. The client constructs the domain name for the AESA in the AESA.ATMA.INT domain. This domain name is "…0.046F.39.AESA.ATMA.INT". The client submits the following query to the ANS:

QUERY …0.046F.39.AESA.ATMA.INT PTR

The client receives SONGS.MYCO.COM as an answer. The client can then enter the query in Scenario 1 above to obtain the list of ATM addresses for the end system. In the list, the client will find an E.164 address for the end system.

II.3 Translating an IP Address to an ATM Address

A client knows the IP address "10.0.0.52" for an end system, and wants to obtain an ATM address for the end system if one exists. The client constructs the domain name for the IP address in the IN-ADDR.ARPA domain. This domain name is "52.0.0.10.IN-ADDR.ARPA". The client submits the following query to the ANS:

QUERY 52.0.0.10.IN-ADDR.ARPA PTR

The client receives SONGS.MYCO.COM as an answer. The client can then enter the query in Scenario 1 above to obtain the list of ATM addresses for the end system.

II.4 Translating an ATM Destination Point Address to an Associated ATM Terminating Interface Address

A client knows the AESA "39.046F.0…" for a destination point, and wants to obtain addresses for one or more terminating interfaces on the destination ATM network. The client constructs the domain name for the AESA in the ATI.ATMA.INT domain. This domain name is "…0.046F.39.ATI.ATMA.INT". The client submits the following query to the ANS:

QUERY …0.046F.39.ATI.ATMA.INT ATMA

The client receives the AESA "39.840F.8…" as an answer, because the wildcarded resource record in the ATI.ATMA.INT domain matches the prefix of the destination point address and no more specific matching resource record exists in that domain.

II.5 Registering and Removing Dynamic Addresses

Dynamic registration and removal of information from the ANS is performed according to the security policy of the zone being updated. In this usage scenario, we consider two zones. These zones are the subtree rooted at MYCO.COM and the subtree rooted at AESA.ATMA.INT. The security policy of these zones allows clients to update information about their ATM addresses. They can use the private key corresponding to the public key registered for their domain name to sign update requests for resource records for their domain name. The clients are also given a private key when they are assigned an ATM address that allows them to register and remove a PTR RR for that address in the AESA.ATMA.INT subtree. Many other security policies are possible and readers are urged to consult the appropriate IETF documents for more information [RFC2065, RFC2136, RFC2137].

SONGS.MYCO.COM dynamically joins and leaves the ATM network that last gave it the AESA "39.046F.0…". It has the private key priv1 corresponding to the public key pub1 recorded for its domain name. It also has the private key priv2 corresponding to the public key pub2 recorded for the domain for its AESA in AESA.ATMA.INT.

In preparation for leaving the ATM network, SONGS.MYCO.COM submits the following updates to the ANS:

DELETE MYCO.COM {SONGS.MYCO.COM IN ATMA 39.046F.O…} sig1

DELETE AESA.ATMA.INT {…0.046F.39.AESA.ATMA.INT IN PTR SONGS.MYCO.COM} sig2

where braces delineate individual resource records, sig1 is the digital signature for the first delete request using the private key priv1, and sig2 is the digital signature for the second delete request using the private key priv2.

When SONGS.MYCO.COM rejoins this ATM network, it is assigned an AESA of "39.046F.1…" and given the key priv3 for the reverse mapping of this AESA. It submits the following updates to ANS:

ADD MYCO.COM TRUE {SONGS.MYCO.COM IN ATMA 39.046F.1…} sig1a

```
ADD AESA.ATMA.INT TRUE {…1.046F.39.AESA.ATMA.INT IN PTR SONGS.MYCO.COM} sig3
```

where braces delineate individual resource records, `sig1a` is the digital signature for the first add request using the private key `priv1`, and `sig3` is the digital signature for the second add request using the private key `priv3`. The second add assumes that a key record for `…1.046F.39.AESA.ATMA.INT` with a public key corresponding to `priv3` already exists in the ANS (but is not shown in Figure 2).

Note that another common scenario would entrust the private keys for updating the `AESA.ATMA.INT` zone to a configuration server that allocates and de-allocates the addresses. In this case, it would be the configuration server, and not `SONGS.MYCO.COM`, that updates the reverse mapping from the AESA to the `SONGS.MYCO.COM` domain name. In this case, one public key for all records in the zone would be recorded and the configuration server would use the corresponding private key. Also note that, for simplicity, these scenarios have used `AESA.ATMA.INT` as the target zone to be updated for AESA to domain name mappings. In reality, a subtree of `AESA.ATMA.INT` that corresponded to the range of addresses to be allocated would be the target zone.

II.6 Retrieving a Public Key

The ANS can provide digital signatures on responses that allow the client to verify the validity of the information in the response. A user wishes to obtain the public key for `SONGS.MYCO.COM`. The user submits the following query to a security-aware client of the ANS:

```
QUERY    SONGS.MYCO.COM    KEY
```

The security-aware client has been configured with an initial public key for an ANS zone. For simplicity, assume that this zone is `MYCO.COM` and that the private key for the zone is used to sign both the dynamic and static resource records for the zone. See [RFC2137] for other policies for placing signatures on resource records. The client requests a digital signature on its response to the query for the public key of `SONGS.MYCO.COM.` The response, signed by `MYCO.COM`, is returned with additional information from the signature resource record (`SIG RR`) for the public key. In particular, the expiration time of the signature indicates when the public key should be retrieved again to check its validity. The client checks the signature and expiration time before accepting the public key. More details about verifying information in the ANS is described in the relevant IETF documents [RFC2065, RFC2137].

II.7 Selecting a Service

Scenarios 2 and 3 assume that the different addresses recorded for `SONGS.MYCO.COM` are addresses for the same end system. This interpretation is the most frequent case. It is possible, however, to store the addresses of multiple locations of a service called `SONGS.MYCO.COM` that delivers the audio for the most recent hits of popular singers. In this case, the addresses in the resource records for `SONGS.MYCO.COM` are interpreted as sources of a service and may actually belong to different servers.

A client, who wishes to obtain songs from the MYCO company, can enter the query in Scenario 1 and use one of the ATM addresses returned to contact a server for the songs. The procedures described in Scenarios 2 and 3 would allow the client to locate other servers for the songs given the address of one of the servers.

II.8 Registering and Removing a Server

An administrator for the songs service described in Scenario 7 is informed that the server at

"+1.908.555.1212" is temporarily out of service, but that a new server is available at "+1.555.555.1212". The administrator uses the private key priv1 to sign the following ANS update request with sig1b:

```
MODIFY   MYCO.COM   {SONGS.MYCO.COM IN ATMA +1.908.555.1212}
{SONGS.MYCO.COM IN ATMA +1.555.555.5555}   sig1b
```

The administrator would also verify that a reverse mapping for "+1.555.555.1212" exists in the AESA.ATMA.INT domain.

Appendix III: Appendix C: Special Considerations When Using Secure Dynamic Update (Informative)

This Appendix does not form an integral part of this specification.

The secure dynamic update specification [RFC2137] describes two different modes for secure operation. In one mode, the zone key is kept offline. Owners of a domain name create signatures for resource records that they update using their private keys. In the other mode (the mode assumed by the scenarios in "Appendix B"), the zone key is kept online. The zone verifies updates from the owners of the domain name, and then signs the resource records with the zone key. Performance issues arise, because generating a digital signature is typically much more expensive than verifying one. High rates of change in ATM addresses can result in high load for an ANS server, for example when changing the prefix for a substantial set of ATM addresses or when high rates of dynamic allocation of addresses to end systems or services occurs. This load is increased if the zone must sign every change.

Other issues arise depending on the configuration, for example the ability to verify the authenticity of data in a zone transfer and the vulnerability of the zone key to compromise. These issues are important to all users of dynamic update. Users should closely consider the issues described in [RFC2137] in light of the load they anticipate for their server and the security they require.

The ANS achieves higher performance by caching the results of queries. There may be a delay in retrieving changes to the ANS until old values in a local cache time out. If more timely information is required, an enhanced ANS client can be created that locates the ANS server that is the authority for the information requested. The client can then directly retrieve the information from that server and refresh its cache. The client would do this in response to a request from a local user for more current information. Other enhancements are possible to refresh both client and server caches, but they require additional, future specification effort.