Network Working Group   T. Johnson
Request for Comments: 3944   U. of North Carolina
Category: Informational   S. Okubo
                                                   Waseda University
                                                         S. Campos
                                                             ITU-T
                                                     December 2004

                        H.350 Directory Services

Status of this Memo

   This memo provides information for the Internet community. It does
   not specify an Internet standard of any kind. Distribution of this
   memo is unlimited.

Abstract

   The International Telecommunications Union Standardization Sector
   (ITU-T) has created the H.350 series of Recommendations that specify
   directory services architectures in support of multimedia
   conferencing protocols. The goal of the architecture is to
   'directory enable' multimedia conferencing so that these services can
   leverage existing identity management and enterprise directories. A
   particular goal is to enable an enterprise or service provider to
   maintain a canonical source of users and their multimedia
   conferencing systems, so that multiple call servers from multiple
   vendors, supporting multiple protocols, can all access the same data
   store.

   Because SIP is an IETF standard, the contents of H.350 and H.350.4
   are made available via this document to the IETF community. This
   document contains the entire normative text of ITU-T Recommendations
   H.350 and H.350.4 in sections 4 and 5, respectively. The remaining
   sections are included only in this document, not in the ITU-T
   version.

Table of Contents

1. Scope

   The International Telecommunications Union Standardization Sector
   (ITU-T) has created the H.350 series of Recommendations that specify
   directory services architectures in support of multimedia
   conferencing protocols. The goal of the architecture is to
   'directory enable' multimedia conferencing so that these services can
   leverage existing identity management and enterprise directories. A
   particular goal is to enable an enterprise or service provider to
   maintain a canonical source of users and their multimedia
   conferencing systems, so that multiple call servers from multiple
   vendors, supporting multiple protocols, can all access the same data
   store.

   H.350 architectures are not intended to change the operation of
   multimedia conferencing protocols in any way. Rather, they are meant
   to standardize the way the already defined protocol elements are
   stored in a directory, so that they can be accessed in a standardized
   manner.

   In the H.350 series, Recommendation H.350 specifies the base
   architecture and object classes, while subordinate Recommendations
   specify elements that are specific to individual protocols.
   Currently, the Recommendations include:

   H.350   - Directory Services Architecture for Multimedia Conferencing
   H.350.1 - Directory Services Architecture for H.323
   H.350.2 - Directory Services Architecture for H.235
   H.350.3 - Directory Services Architecture for H.320
   H.350.4 - Directory Services Architecture for SIP
   H.350.5 - Directory Services Architecture for Non-Standard Protocols

   Because SIP is an IETF standard, the contents of H.350 and H.350.4
   are made available via this document to the IETF community.

2. Terminology

   The following terms are used throughout the document:

   * call server: a protocol-specific signalling engine that routes
      video or voice calls on the network. In H.323 this entity is a
      gatekeeper. In SIP, this entity is a SIP Proxy Server. Note that
      not all signalling protocols use a call server.

   * endpoint: a logical device that provides video and/or voice media
      encoding/decoding, and signalling functions. Examples include:

      * a group teleconferencing appliance that is located in a
        conference room

      * an IP telephone.

      * a software program that takes video and voice from a camera and
        microphone and encodes it and applies signalling using a host
        computer.

  * enterprise directory: A canonical collection of information about
    users in an organization. Typically this information is collected
    from a variety of organizational units to create a whole. For
    example, Human Resources may provide name and address,
    Telecommunications may provide the telephone number, Information
    Technology may provide the email address, etc. For the purposes
    of this architecture, it is assumed that an enterprise directory
    is accessible via LDAP.

  * White Pages: An application that allows end users to look up the
    address of another user. This may be web-based or use some other
    user interface.

3. Conventions used in this document

   Conventions in this document conform to ITU-T guidelines. In this
   Recommendation, the following conventions are used:

   "Shall" indicates a mandatory requirement.

   "Should" indicates a suggested but optional course of action.

   "May" indicates an optional course of action rather than a
   recommendation that something take place.

   References to clauses, sub clauses, annexes and appendices refer to
   those items within this Recommendation unless another specification
   is explicitly listed.

4. H.350

   The normative text of H.350 is reproduced in this section.

4.1. Scope

   This Recommendation describes a directory services architecture for
   multimedia conferencing using LDAP. Standardized directory services
   can support association of persons with endpoints, searchable white
   pages, and clickable dialling. Directory services can also assist in

the configuration of endpoints, and user authentication based on
authoritative data sources. This document describes a standardized
LDAP schema to represent endpoints on the network and associate those
endpoints with users. It discusses design and implementation
considerations for the inter-relation of video and voice-specific
directories, enterprise directories, call servers and endpoints.

The use of a common, authoritative data source for call server,
endpoint, user, authentication and white pages information is an
important aspect of large scale multimedia conferencing environments.
Without a common data source, service providers must create separate
processes to manage each of these functions. By standardizing the
LDAP schema used to represent the underlying data, products from
different system vendors can be deployed together to create an
overall application environment. For example, a white pages search
engine developed by one provider could serve directory information to
IP telephones produced by a second provider, with signalling managed
by a call server produced by yet a third provider. Each of these
disparate systems can access the same underlying data source,
reducing or eliminating the need to coordinate separate management of
each system. A significant benefit to the user is that the
management of this data can be incorporated into existing customer
management tools, allowing for quick and flexible scaling up of
applications. Indeed, many technology providers have already
incorporate LDAP into their products, but have been forced to do so
without benefit of a standardized schema. This Recommendation
represents an effort to standardize those representations to improve
interoperability and performance.

While URLs are already standardized for several conferencing
protocols, their representation in a directory is not. This
Recommendation supports a standardized way for URLs to be searched
and located. This is a necessary step to support 'clickable
dialling'.

Management of endpoint configurations can be improved if the correct
settings are stored by the service provider in a location that is
accessible to both service provider and endpoint. LDAP provides a
convenient storage location that can be accessed by both call server
and endpoint; thus it is possible to use the directory to support
endpoint configuration, which is important for simplified operation
and supporting user mobility. Note that other technologies also
support endpoint configuration, notably the use of SNMP for complete
configuration and SRV records for obtaining registration server
addresses. Therefore, H.350 should be viewed not as an authoritative
endpoint configuration architecture, but rather one tool that can

assist with this task. Note that the use of H.350 has as a feature
endpoint specific configuration, where it is desirable that each
endpoint has a unique configuration.

This architecture uses a generic object class, called commObject, to
represent attributes common to any video or voice protocol. Auxiliary
classes represent specific protocols, such as H.323, H.235, or H.320,
as described in the H.350.x series of Recommendations. Multiple
H.350.x classes can be combined to represent endpoints that support
more than one protocol. For example, endpoints that support H.323,
H.235 and H.320 would include H.350, H.350.1, H.350.2, and H.350.3 in
their LDAP representations. Further, each entry should contain
commObject to serve as the entry's structural object class.

There are two basic components in the architecture. The commURI
object is a class whose only purpose is to link a person or resource
to a commObject. By placing a commURI 'pointer' in an individual's
directory entry, that individual becomes associated with the
particular targeted commObject. Similarly, commObject contains a
pointer, called commOwner, which points to the individual or resource
that is associated with the commObject. In this way, people or
resources can be associated with endpoints. The only change required
in the enterprise directory is the addition of the simple object
class commURI. CommObject data may be instantiated in the same or in
entirely separate directories, thus allowing flexibility in
implementation.

4.1.1. Design Goals

Large-scale deployments of IP video and voice services have
demonstrated the need for complementary directory services
middleware. Service administrators need call servers that are aware
of enterprise directories to avoid duplication of account management
processes. Users need 'white pages' to locate other users with whom
they wish to communicate. All of these processes should pull their
information from canonical data sources in order to reduce redundant
administrative processes and ensure information accuracy. The
following design criteria are established for this architecture. The
architecture will:

1)   enable endpoint information to be associated with people.
     Alternately it enables endpoint information to be associated
     with resources such as conference rooms or classrooms;

2)   enable online searchable "white pages" where dialling
     information (e.g., endpoint addresses) can be found, along with
     other "traditional" directory information about a user, such as
     name, address, telephone, email, etc.;

    3)   enable all endpoint information to be stored in a canonical data
         source (the Directory), rather than local to the call server, so
         that endpoints can be managed through manipulations of an
         enterprise directory, rather than by direct entry into the call
         server;

    4)   support the creation of very large-scale distributed
         directories. These include white pages "portals" that allow
         searching for users across multiple institutional directories.
         In this application, each enterprise directory registers itself
         with (or is unknowingly discovered by) a directory of
         directories that is capable of searching across multiple LDAP
         directories;

    5)   be able to support multiple instances of endpoints per user or
         resource;

    6)   represent endpoints that support more than one protocol, for
         example, endpoints that are both H.320 and H.323;

    7)   store enough information about endpoint configuration so that
         correct configuration settings can be documented to end users on
         a per-endpoint basis, as a support tool, or loaded automatically
         into the endpoint;

    8)   be extendible as necessary to allow implementation-specific
         attributes to be included;

    9)   be non-invasive to the enterprise directory, so that support for
         multimedia conferencing can be added in a modular fashion
         without significant changes to the enterprise directory.

   The scope of this Recommendation does not include extensions of
   functionality to protocols as defined within the protocols
   themselves. It is not the intent of the Recommendation to add
   features, but merely to represent existing protocol attributes. The
   exception to this case is when functionality is implied by the
   directory itself, such as the commPrivate attribute.

4.1.2. Extending the Schema

   H.350 object classes may be extended as necessary for specific
   implementations. For example, a class may be extended to support
   billing reference codes. Extensions to the schema are not considered
   as part of the Recommendation and do not signify compliance.

In some cases it may be necessary to extend the H.350 schemas in
order to represent more information than is supported by the
Recommendations. This may be important for developers that implement
proprietary endpoint functionality that needs to be represented by
attributes in the directory. It may also be important for enterprise
applications. For example 'modelNumber', and 'accountNumber' are
examples of attributes that are not defined in the Recommendation but
may be useful if implemented. Adding attributes to this architecture
must be done in a way that does not break compatibility with this
Recommendation.

A full discussion of schema design and extension is beyond the scope
of this Recommendation. See IETF RFC 2252 for details. Two basic
approaches to schema extension that do not break compatibility with
this Recommendation, are extension through subclass and extension
through the use of auxiliary classes.

4.1.2.1. Extension Through Subclass

It is possible to create a subclass of an existing predefined object
class in order to add new attributes to it. To create a subclass, a
new object class must be defined, that is a subclass of the existing
one, by indicating in the definition of the new class that the
existing class is its superior. Once the subclass is created, new
attributes can be defined within it.

The following example shows how the commObject class can be
subclassed in order to add an attribute to represent a billing
account and a billing manager.

```
objectclass ( BillingInfo-OID
NAME 'BillingInfo'
DESC 'Billing Reference Information'
SUP commObject STRUCTURAL
MAY ( BillingAccount $ BillingManager $ )
)
```

Note that BillingInfo-OID must be replaced by an actual OID. Also
note that, whenever a structural class is extended, its subclass must
also be structural.

The following sample entry shows the newly created attributes. This
example also uses ITU-T Rec. H.350.1 for h323Identity.

```
dn: commUniqueId=2000,ou=h323identity, dc=company, dc=com
objectclass: top
objectclass: commObject
objectclass: h323Identity
```

```
objectclass: BillingInfo
commUniqueId: 2000
BillingAccount: 0023456
BillingManager: John Smith
```

Note that this example and approach demonstrate extension of the
general commObject object class, and not any individual H.350.x
classes. If it is desired to extend an H.350.x auxiliary class, then
that should be accomplished through the definition of additional
auxiliary classes that support the desired attributes, as described
in section 4.1.2.2.

4.1.2.2. Extension Through The Use Of Auxiliary Classes

It is possible to add attributes to an LDAP entry by defining an
auxiliary class containing the new attributes and applying those
attributes to instantiated values in the directory. The auxiliary
class will not be subclassed from any existing object class. Note
that it should have the special class top as its superior. The
following example creates the same billing account and billing
manager attributes as the previous example, but does so by defining
them in their own auxiliary class.

```
objectclass ( BillingInfo-OID
NAME 'BillingInfo'
DESC 'Billing Reference Information'
SUP top AUXILIARY
MAY ( BillingAccount $ BillingManager $ )
)
```

Note how the superior was changed from commObject to top and the
object class changed from being a structural to auxiliary.

It is recommended that all attributes in the auxiliary class be
optional rather than mandatory. In this way, the auxiliary object
class itself can be associated with an entry regardless of whether
any values for its attributes are present.

The following example shows a sample endpoint that utilizes the new
auxiliary class and attributes. This example also uses H.350.1 for
h323Identity.

```
dn: commUniqueId=2000,ou=h323identity, dc=company, dc=com
objectclass: top
objectclass: commObject
objectclass: BillingInfo
```

      commUniqueId: 2000
      BillingAccount: 0023456
      BillingManager: John Smith

4.1.2.3. Object Identifiers

   An attribute's Object Identifier (OID) is a unique numerical
   identifier usually written as a sequence of integers separated by
   dots. For example, the OID for the commUniqueId is
   0.0.8.350.1.1.2.1.1. All attributes must have an OID. OIDs can be
   obtained from anyone who has one and is willing to delegate a portion
   of it as an arc, keeping a record of the arc to avoid duplication.
   Further, the Internet Assigned Numbers Authority (IANA) gives out
   OIDs to any organization that asks.

4.2. commURIObject Definition

   Auxiliary object class that contains the commURI attribute. This
   attribute is added to a person or resource object to associate one or
   more commObject instances with that object. Its values are LDAP URIs
   that point to the associated commObjects, for example, to a user's
   H.323 conferencing station and SIP IP phone. Note that multiple
   instances of commURI need not point to the same commObject directory.
   In fact, each commURI instance could point to an endpoint managed by
   a different service provider.

4.2.1. commURIObject

   OID: 0.0.8.350.1.1.1.2.1
   objectclasses: (0.0.8.350.1.1.1.2.1
   NAME 'commURIObject'
   DESC 'object that contains the URI attribute type'
   SUP top AUXILIARY
   MAY (commURI)
)

4.2.2. commURI

   OID: 0.0.8.350.1.1.1.1.1
   attributetypes:(0.0.8.350.1.1.1.1.1
   NAME 'commURI'
   DESC 'Labeled URI format to point to the distinguished name of the
   commUniqueId'
   EQUALITY caseExactMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15)
   Application utility class
        Standard

      Number of values
           multi
      Definition
           Labelled URI containing an LDAP URL identifying the directory
      containing the referenced commObject instance. The search filter
      specified by this LDAP URL shall specify an equality search of the
      commUniqueId attribute of the commObject class.
      Permissible values (if controlled)
      Notes
           Used to find the endpoint of the user in question. The label
      field may be used to represent the function of the endpoint, such as
      'home IP phone' or 'desktop video' for user interface display
      purposes.
           Note that the label portion of the field may contain spaces as
      in the example below showing 'desktop video'.
      Semantics
      Example applications for which this attribute would be useful
      Example (LDIF fragment)
      commURI:
      ldap://directory.acme.com/dc=acme,dc=com??sub?(commUniqueId=bob)
      desktop video

4.3. CommObject Definition

   Abstraction of video or voice over IP device. The commObject class
   permits an endpoint (H.323 endpoint or SIP user agent or other
   protocol endpoint) and all their aliases to be represented by a
   single entry in a directory. Note that every directory entry should
   contain commObject as the entry's structural object class. That
   entry may also contain H.350.x auxiliary classes.

4.3.1. commObject

   OID: 0.0.8.350.1.1.2.2.1
   objectclasses: (0.0.8.350.1.1.2.2.1
   NAME 'commObject'
   DESC 'object that contains the Communication attributes'
   SUP top STRUCTURAL
   MUST commUniqueId
   MAY (commOwner $ commPrivate)
)

4.3.2. commUniqueId

   OID: 0.0.8.350.1.1.2.1.1
   attributetypes: (0.0.8.350.1.1.2.1.1
   NAME 'commUniqueId'
   DESC 'To hold the endpoints unique Id'

```
     EQUALITY caseIgnoreIA5Match
     SUBSTR caseIgnoreIA5SubstringsMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
     Application utility class
          standard
     Number of values
          multi
     Definition
          The endpoint's unique ID.
     Permissible values (if controlled)
     Notes
          This is the RDN of this object.  In practice, there will always
     be one and only one commUniqueId for every endpoint.  This attribute
     uniquely identifies an endpoint in the commObject directory.  It must
     be unique within that directory, but need not be unique globally.
     This attribute has no relationship to the enterprise directory.
     Semantics
     Example applications for which this attribute would be useful
     Example (LDIF fragment)
     commUniqueId: bob
```

4.3.3. commOwner

```
     OID: 0.0.8.350.1.1.2.1.2
     attributetypes: 0.0.8.350.1.1.2.1.2
     NAME 'commOwner'
     DESC 'Labeled URI to point back to the original owner'
     EQUALITY caseExactMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
     Application utility class
          Standard
     Number of values
          multi
     Definition
          Labelled URI format to point back to the person or resource
     object associated with this entry.
     Permissible values (if controlled)
     Notes
          Used as a reverse entry finder of the owner(s).  This attribute
     may point to groups.  Note that this URI can point to a cn, but in
     applications where it is desired to bind authentication information
     across both the commObject and enterprise directories, it may be
     desirable that commOwner points to a dn rather than a cn, thus
     uniquely identifying the owner of the commObject.
     Semantics
     Example applications for which this attribute would be useful
     Example (LDIF fragment)
```

```
commOwner:
ldap://directory.acme.com/dc=acme,dc=com??sub?(cn=bob%20smith)
commOwner: uid=bob,ou=people,dc=acme,dc=com
```

4.3.4. commPrivate

```
OID: 0.0.8.350.1.1.2.1.3
attributetypes: (0.0.8.350.1.1.2.1.3
NAME 'commPrivate'
DESC 'To decide whether the entry is visible to world or not'
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
Application utility class
    Standard
Number of values
    multi
Definition
    To be used by the user and indicate privacy options for an
endpoint, i.e., unlisted number.
Permissible values (if controlled)
Notes
    This attribute is defined as Boolean.  Future version of this
Recommendation may develop a controlled vocabulary for this
attribute to accommodate multiple types of privacy.
Semantics
Example applications for which this attribute would be useful
Example (LDIF fragment)
commPrivate: true
```

4.4. CommObject LDIF Files

This section contains a schema configuration file for commURIObject
and commObject that can be used to configure an LDAP server to
support these classes.

4.4.1. LDIF for commURIObject

```
# Communication Object Schema
#
# Schema for Representing Communication Objects in an LDAP Directory
#
# Abstract
#
# This document defines the schema for representing Communication
# objects in an LDAP directory [LDAPv3].  It defines schema elements
# to represent a communication object URI [commURIObject].
#
#
#
```

```
#                          .1 = Communication related work
#                          .1.1 = commURIObject
#                          .1.1.1 = attributes
#                          .1.1.2 = objectclass
#                          .1.1.3 = syntax
#
# Attribute Type Definitions
#
#    The following attribute types are defined in this document:
#
#        commURI
dn: cn=schema
changetype: modify
#
# if you need to change the definition of an attribute,
#          then first delete and re-add in one step
#
# if this is the first time you are adding the commObject
# objectclass using this LDIF file, then you should comment
# out the delete attributetypes modification since this will
# fail.  Alternatively, if your ldapmodify has a switch to continue
# on errors, then just use that switch -- if you're careful
#
delete: attributetypes
attributetypes: (0.0.8.350.1.1.1.1.1 NAME 'commURI' )
-
#
# re-add the attributes -- in case there is a change of definition
#
#
add: attributetypes
attributetypes: (0.0.8.350.1.1.1.1.1
     NAME 'commURI'
     DESC 'Labeled URI format to point to the distinguished name of
the commUniqueId'
     EQUALITY caseExactMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
-
# Object Class Definitions
#
#    The following object classes are defined in this document:
#
#        commURIObject
#
# commURIObject
#
#    This auxiliary object class represents a URI attribute type
#
```

```
#
delete: objectclasses
objectclasses: (0.0.8.350.1.1.1.2.1 NAME 'commURIObject' )
-
add: objectclasses
objectclasses: (0.0.8.350.1.1.1.2.1
     NAME 'commURIObject'
     DESC 'object that contains the URI attribute type'
     SUP top AUXILIARY
     MAY ( commURI )
        )
-
#
# end of LDIF
#
```

4.4.2. LDIF for commObject

```
# Communication Object Schema
#
# Schema for Representing Communication Objects in an LDAP Directory
#
# Abstract
#
# This document defines the schema for representing Communication
# objects in an LDAP directory [LDAPv3].  It defines schema elements
# to represent a communication object [commObject].
#
#
#                       .1 = Communication related work
#                       .1.2 = commObject
#                       .1.2.1 = attributes
#                       .1.2.2 = objectclass
#                       .1.2.3 = syntax
#
#
# Attribute Type Definitions
#
#     The following attribute types are defined in this document:
#
#         commUniqueId
#         commOwner
#         commPrivate
dn: cn=schema
changetype: modify
#
# if you need to change the definition of an attribute,
#            then first delete and re-add in one step
```

```
#
# if this is the first time you are adding the commObject
# objectclass using this LDIF file, then you should comment
# out the delete attributetypes modification since this will
# fail. Alternatively, if your ldapmodify has a switch to continue
# on errors, then just use that switch -- if you're careful
#
delete: attributetypes
attributetypes: (0.0.8.350.1.1.2.1.1 NAME 'commUniqueId' )
attributetypes: (0.0.8.350.1.1.2.1.2 NAME 'commOwner' )
attributetypes: (0.0.8.350.1.1.2.1.3 NAME 'commPrivate' )
-
#
# re-add the attributes -- in case there is a change of definition
#
#
add: attributetypes
attributetypes: (0.0.8.350.1.1.2.1.1
     NAME 'commUniqueId'
     DESC 'To hold the endpoints unique Id'
     EQUALITY caseIgnoreIA5Match
     SUBSTR caseIgnoreIA5SubstringsMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
attributetypes: (0.0.8.350.1.1.2.1.2
     NAME 'commOwner'
     DESC 'Labeled URI to point back to the original owner'
     EQUALITY caseExactMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
attributetypes: (0.0.8.350.1.1.2.1.3
     NAME 'commPrivate'
     DESC 'To decide whether the entry is visible to world or not'
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
-
# Object Class Definitions
#
#    The following object classes are defined in this document:
#
#        commObject
#
# commObject
#
#
delete: objectclasses
objectclasses: (0.0.8.350.1.1.2.2.1 NAME 'commObject' )
-
add: objectclasses
objectclasses: (0.0.8.350.1.1.2.2.1
     NAME 'commObject'
```

```
      DESC 'object that contains the Communication attributes'
      SUP top STRUCTURAL
      MUST commUniqueId
      MAY ( commOwner $ commPrivate )
      )
-
#
# end of LDIF
#
```

4.5. H.350 Annex A Indexing Profile

   Indexing of attributes is an implementation-specific activity and
   depends upon the desired application. Non-indexed attributes can
   result in search times sufficiently long to render some applications
   unusable. Notably, user and alias lookup should be fast. The Annex
   A Indexing Profile describes an indexing configuration for commObject
   directories that will be optimized for use in directory of
   directories applications. Use of this profile is optional.

   commURI: no recommendation

   commUniqueId: equality

   commOwner: presence

   commPrivate: presence

5. H.350.4

   The normative text of H.350 is reproduced in this section.

5.1. Scope

   This Recommendation describes an LDAP directory services architecture
   for multimedia conferencing using SIP. In particular, it defines an
   LDAP schema to represent SIP User Agents (UAs) on the network and
   associate those endpoints with users.

   This Recommendation is intended to supplement the CommObject
   directory architecture as discussed in ITU-T Rec. H.350, and not
   intended to be used as a stand-alone architecture. The
   implementation of this LDAP schema, together with the use of the
   H.350 CommObject architecture, facilitates the integration of SIP
   User Agents and conferencing devices into existing Enterprise
   Directories, thus allowing the user to perform white page lookups and
   access clickable dialling supported by SIP devices. The primary
   reasons for implementing this schema include those listed in ITU-T

Rec. H.350 (the CommObject class definition) as they apply
specifically to the use of SIP UAs, and to facilitate vendors making
SIP services more readily available to their users.

The scope of this Recommendation includes recommendations for the
architecture to integrate endpoint information for endpoints using
SIP into existing enterprise directories and white pages.

The scope of this Recommendation does not include normative methods
for the use of the LDAP directory itself or the data it contains. The
purpose of the schema is not to represent all possible data elements
in the SIP protocol, but rather to represent the minimal set required
to accomplish the design goals enumerated in ITU-T Rec. H.350.

Note that SIP provides well-defined methods for discovering registrar
addresses and locating users on the network. Some of the attributes
defined here are intended for more trivial or manual implementations
and may not be needed for all applications. For example,
SIPIdentityRegistrarAddress and SIPIdentityAddress may not be needed
for many applications, but are included here for completeness. Thus,
SIPIdentitySIPURI is the primary attribute of interest that will be
served out, especially for white page directory applications.

5.1.1. Extending the schema

The SIPIdentity classes may be extended as necessary for specific
implementations. See the base of ITU-T Rec. H.350 for a discussion
on schema extension.

5.2. Object class definitions

The SIPIdentity object class represents SIP User Agents (UAs). It is
an auxiliary class and is derived from the commObject class, which is
defined in the ITU-T Rec. H.350.

5.2.1. SIPIdentity

OID: 0.0.8.350.1.1.6.2.1
objectclasses: (0.0.8.350.1.1.6.2.1
NAME 'SIPIdentity'
DESC 'SIPIdentity object'
SUP top AUXILIARY
MAY (SIPIdentitySIPURI $ SIPIdentityRegistrarAddress $
   SIPIdentityProxyAddress $ SIPIdentityUserName $
   SIPIdentityPassword $ SIPIdentityServiceLevel $
   userSMIMECertificate)
)

5.2.2. SIPIdentitySIPURI

    OID: 0.0.8.350.1.1.6.1.1
    attributetypes: (0.0.8.350.1.1.6.1.1
    NAME 'SIPIdentitySIPURI'
    DESC 'Universal Resource Indicator of the SIP UA'
    EQUALITY caseExactMatch
    SUBSTR caseExactSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15)
    Application utility class
        standard
    Number of values
        multi
    Definition
        Uniform Resource Identifier that identifies a communication
    resource in SIP. Usually contains a user name and a host name and is
    often similar in format to an email address.
    Permissible values (if controlled)
    Notes
        This URI may institute SIP or SIPS (secure). In the event that
    SIPS is instituted, the URI must reflect that it is using SIPS as
    opposed to SIP. See Examples below.
    Semantics
    Example applications for which this attribute would be useful
        Online representation of most current listing of a user's
    SIP(S) UA.
    Example
    SIPIdentitySIPURI: sip:alice@foo.com   // SIP example
    SIPIdentitySIPURI: sip:alice@152.2.158.212   // SIP example
    SIPIdentitySIPURI: sips:bob@birmingham.edu   // SIPS example

5.2.3. SIPIdentityRegistrarAddress

    OID: 0.0.8.350.1.1.6.1.2
    attributetypes: (0.0.8.350.1.1.6.1.2
    NAME 'SIPIdentityRegistrarAddress'
    DESC 'specifies the location of the registrar'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26)
    Application utility class
        Standard
    Number of values
        multi
    Definition
        Address for the domain to which the server that handles
    REGISTER requests and forwarding to the location server for a
    particular domain belongs.
    Permissible values (if controlled)

    Notes
        Note that RFC 3261 states that user agents can discover their
    registrar address by configuration, using the address-of-record, or
    by multicast. The first scenario, by configuration, is noted as out
    of scope for RFC 3261. This attribute may be used for the first
    scenario. It can be accomplished manually, (e.g., a web page that
    displays a user's correct registrar address) or automatically with
    an H.350.4 aware user agent.
    Semantics
    Example applications for which this attribute would be useful
        white pages, a web page that displays a user's correct
    configuration information.
    Example (LDIF fragment)
    SIPIdentityRegistrarAddress: 152.2.15.22   //IP address example
    SIPIdentityRegistrarAddress: sipregistrar.unc.edu //FQDN example

5.2.4. SIPIdentityProxyAddress

    OID: 0.0.8.350.1.1.6.1.3
    attributetypes: (0.0.8.350.1.1.6.1.3
    NAME 'SIPIdentityProxyAddress'
    DESC 'Specifies the location of the SIP Proxy'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26)
    Application utility class
        Standard
    Number of values
        multi
    Definition
        Address which specifies the domain location of SIP proxy within
    a domain. RFC 3261 defines the role of the SIP proxy.
    Permissible values (if controlled)
    Notes
        SIP User Agents are not REQUIRED to use a proxy, but will in
    many cases.
    Semantics
    Example applications for which this attribute would be useful
        white pages, a web page that displays a user's correct
    configuration information.
    Example (LDIF fragment)
    SIPIdentityProxyAddress: 172.2.13.234   //IP address example
    SIPIdentityProxyAddress: sipproxy.unc.edu //FQDN example

5.2.5. SIPIdentityAddress

   OID: 0.0.8.350.1.1.6.1.4
   attributetypes: (0.0.8.350.1.1.6.1.4
   NAME 'SIPIdentityAddress'
   DESC 'IP address or FQDN of the UA'
   EQUALITY caseIgnoreIA5Match
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.26)
   Application utility class
        standard
   Number of values
        multi
   Definition
        Specifies the IP address or fully qualified domain name of the
   UA.
   Permissible values (if controlled)
   Notes
        This attribute may be useful for applications in which UA to UA
   communication is direct, not involving a proxy or registrar.
   Example applications for which this attribute would be useful
        A web page that displays a user's proper user agent
   configuration information.
   Example (LDIF fragment)
   SIPIdentityAddress: 152.2.121.36   // IP address example
   SIPIdentityAddress: ipPhone.foo.org   // FQDN example

5.2.6. SIPIdentityPassword

   OID: 0.0.8.350.1.1.6.1.5
   attributetypes: (0.0.8.350.1.1.6.1.5
   NAME 'SIPIdentityPassword'
   DESC 'The user agent SIP password '
   EQUALITY octetStringMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.40)
   Application utility class
        Standard
   Number of values
        multi
   Definition
        The SIP user agent's password, used for the HTTP digest
   authentication scheme as defined in RFC 2617.
   Permissible values (if controlled)
   Notes
        Because RFC 2069, which was made obsolete by RFC 2617, was used
   as the basis for HTTP Digest in RFC 2543, any SIP servers supporting
   RFC 2617 must ensure backward compatibility with RFC 2069.
        This SIPIdentityUserName, together with SIPIdentityPassword,
   are reserved for the purpose of use with Digest Access

Authentication, and not intended for use with Basic Authentication
methods.
     LDAP provides one method to store user passwords for reference.
If passwords are stored in LDAP it makes the LDAP server a
particularly valuable target for attack. Implementors are encouraged
to exercise caution and implement appropriate security procedures
such as encryption, access control, and transport layer security for
access to this attribute.
Semantics
Example applications for which this attribute would be useful
Example (LDIF fragment)
SIPIdentityPassword: 36zxJmCIB18dM0FVAj

5.2.7. SIPIdentityUserName

OID: 0.0.8.350.1.1.6.1.6
attributetypes: (0.0.8.350.1.1.6.1.6
NAME 'SIPIdentityUserName'
DESC 'The user agent user name.'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15)
Application utility class
     Standard
Number of values
     multi
Definition
     The SIP user agent's user name, used for the HTTP digest
authentication scheme as defined in RFC 2617.
Permissible values (if controlled)
Notes
     Because RFC 2069, which was made obsolete by RFC 2617, was used
as the basis for HTTP Digest Authentication in RFC 2543, any SIP
servers supporting HTTP Digest Authentication as defined in RFC 2617
must ensure backward compatibility with RFC 2069.
     This SIPIdentityUserName, together with SIPIdentityPassword,
are reserved for the purpose of use with Digest Access
Authentication, and not intended for use with Basic Authentication
methods.
     Note that in many cases the user name will be parsed from the
user@proxy.domain portion of the SIP URI. In that case it may not be
necessary to populate this attribute.
Semantics
Example applications for which this attribute would be useful
Example (LDIF fragment)
SIPIdentityUserName: nelkhour

5.2.8. SIPIdentityServiceLevel

   OID: 0.0.8.350.1.1.6.1.7
   attributetypes: (0.0.8.350.1.1.6.1.7
   NAME 'SIPIdentityServiceLevel'
   DESC 'To define services that a user can belong to.'
   EQUALITY caseIgnoreIA5Match
   SUBSTR caseIgnoreIA5SubstringsMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.26)
   Application utility class
        Standard
   Number of values
        multi
   Definition
        This describes the level of services a user can belong to.
   Permissible values (if controlled)
   Notes
        This attribute does not represent a data element found in SIP.
   SIP itself does not support distinctions in service levels. Instead,
   this attribute provides a mechanism for the storage of service level
   information directly in LDAP. This mapping allows service providers
   to adapt to an existing LDAP directory without changing the values
   of the SIPIdentityServiceLevel instances in the directory.
   Semantics
   Example applications for which this attribute would be useful
   Example (LDIF fragment)
   SIPIdentityServiceLevel: premium

5.3. SIPIdentity LDIF Files

   This clause contains a schema configuration file for SIPIdentity
   that can be used to configure an LDAP server to support this class.

SIPIdentity Object Schema
#
Schema for representing SIPIdentity Object in an LDAP Directory
#
Abstract
#
This Recommendation defines the schema for representing
SIPIdentity
object in an LDAP directory [LDAPv3]. It defines schema elements
to represent an SIPIdentity object [SIPIdentity].
#
.1 = Communication related work
.1.6 = SIPIdentity
.1.6.1 = attributes
.1.6.2 = objectclass

```
#                       .1.6.3 = syntax
#
#
#
# Attribute Type Definitions
#
#     The following attribute types are defined in this
Recommendation:
#
#     SIPIdentitySIPURI
#     SIPIdentityRegistrarAddress
#     SIPIdentityProxyAddress
#     SIPIdentityAddress
#     SIPIdentityPassword
#     SIPIdentityUserName
#     SIPIdentityServiceLevel
dn: cn=schema
changetype: modify
#
# if you need to change the definition of an attribute,
#           then first delete and re-add in one step
#
# if this is the first time you are adding the SIPIdentity
# objectclass using this LDIF file, then you should comment
# out the delete attributetypes modification since this will
# fail.  Alternatively, if your ldapmodify has a switch to continue
# on errors, then just use that switch -- if you are careful
#
delete: attributetypes
attributetypes: (0.0.8.350.1.1.6.1.1 NAME 'SIPIdentitySIPURI' )
attributetypes: (0.0.8.350.1.1.6.1.2 NAME 'SIPIdentityRegistrarAddress')
attributetypes: (0.0.8.350.1.1.6.1.3 NAME 'SIPIdentityProxyAddress')
attributetypes: (0.0.8.350.1.1.6.1.4 NAME 'SIPIdentityAddress' )
attributetypes: (0.0.8.350.1.1.6.1.5 NAME 'SIPIdentityPassword' )
attributetypes: (0.0.8.350.1.1.6.1.6 NAME 'SIPIdentityUserName' )
attributetypes: (0.0.8.350.1.1.6.1.7 NAME 'SIPIdentityServiceLevel')
-
#
# re-add the attributes -- in case there is a change of definition
#
#
add: attributetypes
attributetypes: (0.0.8.350.1.1.6.1.1
     NAME 'SIPIdentitySIPURI'
     DESC 'Universal Resource Indicator of the SIP UA'
     EQUALITY caseExactMatch
     SUBSTR caseExactSubstringsMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

```
attributetypes: (0.0.8.350.1.1.6.1.2
     NAME 'SIPIdentityRegistrarAddress'
     DESC 'specifies the location of the registrar'
     EQUALITY caseIgnoreIA5Match
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
attributetypes: (0.0.8.350.1.1.6.1.3
     NAME 'SIPIdentityProxyAddress'
     DESC 'Specifies the location of the SIP Proxy'
     EQUALITY caseIgnoreIA5Match
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
attributetypes: (0.0.8.350.1.1.6.1.4
     NAME 'SIPIdentityAddress'
     DESC 'IP address of the UA'
     EQUALITY caseIgnoreIA5Match
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
attributetypes: (0.0.8.350.1.1.6.1.5
     NAME 'SIPIdentityPassword'
     DESC 'The user agent SIP password '
     EQUALITY octetStringMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 )
attributetypes: (0.0.8.350.1.1.6.1.6
     NAME 'SIPIdentityUserName'
     DESC 'The user agent user name.'
     EQUALITY caseIgnoreMatch
     SUBSTR caseIgnoreSubstringsMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
attributetypes: (0.0.8.350.1.1.6.1.7
     NAME 'SIPIdentityServiceLevel'
     DESC 'To define services that a user can belong to.'
     EQUALITY caseIgnoreIA5Match
     SUBSTR caseIgnoreIA5SubstringsMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
-
# Object Class Definitions
#
#     The following object class is defined in this Recommendation:
#
#         SIPIdentity
#
# SIPIdentity
#
#
delete: objectclasses
objectclasses: (0.0.8.350.1.1.6.2.1 NAME 'SIPIdentity' )
-
add: objectclasses
objectclasses: (0.0.8.350.1.1.6.2.1
     NAME 'SIPIdentity'
```

```
      DESC 'SIPIdentity object'
      SUP top AUXILIARY
      MAY ( SIPIdentitySIPURI $ SIPIdentityRegistrarAddress $
           SIPIdentityProxyAddress $ SIPIdentityAddress $
           SIPIdentityPassword $ SIPIdentityUserName $
           SIPIdentityServiceLevel $ userSMIMECertificate )
      )
-
#
# end of LDIF
#
```

5.4. H.350.4 Annex A Indexing profile

   Indexing of attributes is an implementation-specific activity and
   depends upon the desired application. Non-indexed attributes can
   result in search times sufficiently long to render some applications
   unusable. Notably, user and alias lookup should be fast. The Annex
   A Indexing Profile describes an indexing configuration for
   SIPIdentity directories that will be optimized for use in directory
   of directories applications. Use of this profile is optional.

   SIPIdentitySIPURI: equality

   SIPIdentityRegistrarAddress: no recommendation

   SIPIdentityProxyAddress: no recommendation

   SIPIdentityAddress: equality

   SIPIdentityUserName: equality

   SIPIdentityPassword: no recommendation

   SIPIdentityServiceLevel: equality

6. Acknowledgments

   We are grateful to numerous colleagues for reaching across multiple
   boundaries of standards bodies, research networks, academia and
   private industry in order to produce an architecture that works
   toward integrating multimedia conferencing deployments. In
   particular, standards from both IETF and ITU-T were drawn from
   extensively, and the architecture is meant to serve all communities.

7. Security Considerations

   This section is not present in the ITU-T standard, but gives
   information for the IETF community. Its content has the consensus of
   the ITU-T Study Group 16.

   H.350 does not alter the security architectures of any particular
   protocol. However, it does offer a standardized place to store
   authentication credentials where appropriate. It should be noted
   that both H.323 and SIP support shared secret authentication (H.235
   Annex D and HTTP Digest, respectively). These approaches require
   that the call server have access to the password. Thus, if the call
   server or H.350 directory is compromised, passwords also may become
   compromised. These weaknesses may be due to weaknesses in the
   systems (H.350 directory or call servers) and their operation rather
   than in H.350 per se.

   The userSMIMECertificate attribute is defined in RFC 2798 (section
   2.8) as a part of inetOrgPerson. The SIP user agent's X.509
   certificate can be stored in this attribute. When the certificate is
   present, it can be employed with S/MIME to provide authentication,
   integrity, and confidentiality as specified in RFC 3261 [5].

   It is strongly encouraged that call servers and an H.350 directory
   mutually authenticate each other before sharing information.
   Further, it is strongly encouraged that communications between H.350
   directories and call servers or endpoints happen over secure
   communication channels such as SSL or TLS.

   Finally, access control lists on LDAP servers are a matter of policy
   and are not a part of the standard. System administrators are
   advised to use common sense when setting access control on H.350
   attributes. For example, password attributes should only be
   accessible by the authenticated user, while address attributes might
   be publicly available.

8. References

8.1. Normative References

   [1] Hodges, J. and R. Morgan, "Lightweight Directory Access Protocol
        (v3): Technical Specification", RFC 3377, September 2002.

   [2] ITU-T Recommendation H.350, "Directory services architecture for
        multimedia conferencing", 2003.

   [3] ITU-T Recommendation H.350.4, "Directory services architecture
        for SIP", 2003.

   [4] Franks, J., Hallam-Baker P., Hostetler, J., Lawrence, S., Leach,
        P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic
        and Digest Access Authentication", RFC 2617, June 1999.

   [5] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A.,
        Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP:
        Session Initiation Protocol", RFC 3261, June 2002.

   [6] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol
        (SIP): Locating SIP Servers", RFC 3263, June 2002.

   [7] Smith, M., "Definition of the inetOrgPerson LDAP Object Class",
        RFC 2798, April 2000.

8.2. Informative References

   [8] ITU-T Recommendation H.350.1, "Directory services architecture
        for H.323", 2003.

   [9] ITU-T Recommendation H.350.2, "Directory services architecture
        for H.235", 2003.

   [10] ITU-T Recommendation H.350.3, "Directory services architecture
        for H.320", 2003.

   [11] ITU-T Recommendation H.350.5, "Directory services architecture
        for Non-Standard Protocols", 2003.

   [12] ITU-T Recommendation H.350.6, "Directory services architecture
        for Call Forwarding and Preferences", 2004.

   [13] Howes T. and M. Smith, "Understanding And Deploying LDAP
        Directory Services", New Riders Publishing, ISBN: 1578700701,
        1999.

   [14] Howes T. and M. Smith, "LDAP Programming Directory-Enabled
        Applications with Lightweight Directory Access Protocol", New
        Riders Publishing, ISBN: 1578700000, 1997.

9. Relationship to Other Specifications

   This specification is an RFC publication of an ITU-T publication [4],
   without textual changes within the standard itself (Section 4). The
   present section appears in the RFC publication only. In order for
   this specification to be implemented properly, a number of standards
   pertaining to LDAP [1], [7], H.350 [2],[3], and SIP [4], [5], [6],
   [7], need to be implemented in whole or in part by the implementor.

   For some background information on the ITU and IETF directory service
   protocols, reading [8], [9], [10], [11], and [12] is valuable, and
   [13] and [14] are recommended books.

10. Authors' Addresses

   Tyler Johnson
   Editor, H.350
   University of North Carolina
   Chapel Hill, NC 27599

   Phone: +1.919.843.7004
   EMail: Tyler_Johnson@unc.edu

   Sakae Okubo
   Rapporteur for Q.4/16, ITU-T SG16
   Waseda University
   YRP Ichibankan, 3-4 Hikarinooka
   Yokosuka-shi, 239-0847 Japan

   Phone: +81 46 847 5406
   EMail: sokubo@waseda.jp

   Simao Ferraz de Campos Neto
   Counsellor, ITU-T SG 16
   International Telecommunication Union
   Place des Nations
   Geneva CH1211 - Switzerland

   Phone: +41-22-730-6805
   EMail: simao.campos@itu.int

Full Copyright Statement

Intellectual Property

Acknowledgement