Internet Architecture Board   P. Faltstrom, Ed.
Internet-Draft   G. Huston, Eds., Ed.
Expires: June 14, 2005   IAB
                                                        December 14, 2004

A Survey of Internet Identities
draft-iab-identities-02.txt

Status of this Memo

Copyright Notice

Abstract

This memo provides an overview of the various realms of identification used within the Internet protocol suite, with a view to noting the interdependencies of the different identifiers and consequent implications for updating their specifications or changing their infrastructures' operations.

Table of Contents

1. Introduction

In any communications domain where two parties wish to conduct a conversation across a network each party must specify to the network sufficient information for the network to identify the other party. When the conversation refers to a resource or service that is accessible through the network, the only effective way to refer to such a resource of service is to use an identifier that can subsequently be passed to the network to perform the access.

Some networks use a single identifier domain to identity all parties and services. Other networks use a collection of discrete identifier domains, where each identifier domain has a specific realm of discourse or application. The Internet is an example of a multiple-identifier domain network, where there are a number of identity domains, each referring to a particular function or area of application. In terms of routing and forwarding IP packets the identity domain used is that of IP addresses, while in terms of identifying particular services or resources the URI form of identity is commonly used. In terms of human use of identities, the most common form of identity in the Internet is based upon the domain name.

This memo examines the role of identities and identifiers, together with an overview of the various realms of identity that are used in the Internet. The document then looks in more detail at the Domain Name System (DNS) and examines its role in relation to these identity realms.

One of the characteristics of the Internet's multiple identifier systems is their heavy interdependence. This memo notes those interrelationships and provides some observations on implications for technical or operational evolution of their specifications.

1.1 Desirable properties of Internet Identities

Before exploring the set of Internet-based identity realms, its useful to enumerate a set of desirable characteristics of any useful identity system. The following list is of characteristics and some related questions related to properties of the identifier is proposed as a useful, although not comprehensive, collection of identity attributes:

Uniqueness:

In what realm is the identifier unique?

Can the same identifier be associated with two or more distinct

objects within the domain of a single realm?

Can multiple identifiers in a single realm be associated with the same object?

An identifier can only be used reliably and deterministically when there is a unique association with an object. An identity realm is generally useful when the association between identities and objects is a relationship where each unique identifier references a single unique object. Note that there is no requirement in the reverse direction, in that it typically makes little difference to the utility of an identity realm if a unique object is associated with multiple identities. In other words it can lead to ambiguities in identity resolution if an identity is associated with two or more distinct objects, but it generally is not as critical if an object is associated with multiple identities (i.e. multiple identity aliases for a unique object).

Consistency:

Is the identity asserted within a consistent identifier space?

This avoids an assertion of identity being interpreted by another party in an unintended manner.

Persistence:

Does the identity remain constant, or are gratuitous changes in the mapping from the identifier to the referenced object avoided?

Constantly changing identities are, at the very least, difficult to track.

Trust:

Can a particular identity withstand a challenge as to its validity?

Other parties who would like to use this identity would like to be reassured that they are not being deceived. 'Use' in this context is a generic term that includes actions such as resolution to the object identified by the identity value, storage of the identity value for subsequent 'use', referral, where the token is passed to another party for their 'use'.

Robustness:

Is the identity realm capable of withstanding deliberate or unintentional attempts to corrupt it in various ways?

A robust identity system should be able to withstand efforts to undertake identity theft or identity fraud.

Withholding:

If the identity is composed of a number of components, are only those components of the identity that are essential to support the communication exposed to other parties?

Compound identity systems should not reveal those components of the identity structure that are not relevant to the identity operation being performed.

Referential Consistency:

If the identity is used in the context of a reference, then when the referenced object is altered or relocated, does the identifier remain a valid reference to the object?

Referential consistency refers not only to the constancy of the reference in the face of changes to the referenced object, but also consistemncy when one entity passed the identity value to another. The identity resolution should remain constant in such cases.

Structure:

Is the token space from which identity values are drawn structured or unstructured?

Structured token spaces allow various forms of retrieval operations based on the identity value to be undertaken efficiently, while unstructured token spaces allow for more flexible generation and use of identities within more restrictive realms of discourse.

This list is not attempting to be a complete enumeration of required identifier properties, but instead list the most important desireable properties of identifier realms in the context of the Internet.

2. A Hierarchy of Identities

In networking models there is a conceptual layering of functionality, starting at the layer of bits on the wire at the media access level and moving up a stack of layers through internetworking, end-to-end transport and application levels. Each one of these layers creates

at least one context in which identifiers are used for the communication. It would appear that from this perspective an identity within the Internet is not just a single identity, but an collection of various identities, used in a variety of contexts.

2.1 Media Access Addresses

There are two generic types of base media in this realm. One is a point-to-point medium, a bilateral communications system where all Protocol Data Units (PDUs) generated by one party are passed to the other party. In such environments use of media access addresses are not strictly required. The other form of environment is a multi-access environment, where a number of parties can communicate directly using a common medium. In this environment the sender must specify the intended recipient of the PDU, and to achieve this all connected entities must use a unique media access address. The most common of these multi-access media are encompassed within the IEEE 802 collection of media standards.

These IEEE 802 technologies share a common structure of Media Access Control layer address (MAC address) to uniquely identity devices connected within a LAN. There are two forms of this identity space, one using a 48 bit identity space (EUI-48 [21]), and the other a 64 bit space (EUI-64 [22]). Both identity spaces can be considered as partially-structured identity spaces, where a number of bits within this MAC address determines whether the address has been globally or locally assigned. Globally assigned values are globally unique, but are structured in such a way that there is no imposed hierarchy within the address that could be used for efficient searching, in contexts such as, for example, a routing or forwarding application.

A global MAC address identity certainly passes one of the more basic tests of an identity domain, that of uniqueness. Two parties cannot assume the same MAC address value and use this same value as a unique identity. So in a LAN context, a collection of devices can distinguish between each other by virtue of this unique MAC address. A manufacturer of Ethernet devices is assigned a manufacturer's block of Ethernet MAC addresses and uniquely places one address in each device. The end consumer has no need to reconfigure the device with a new address, nor is there any need to alter existing MAC addresses each time the LAN changes with new devices being added, and the address is intended to be globally unique.

Beyond these attributes there are some real weaknesses in using a MAC address as an identity outside the context of a LAN environment. The identity space, while globally unique, has few other distinguishing properties. The structure of the identity space does not reflect its current location within a particular network topology, so its of no

assistance as a location token. In the context of equating a device identity to this network interface identity, the identity has limited persistence, in that it follows the interface hardware, not the host computer or its use. For example, switching a device from a wireless to a wired connection changes its MAC identity. The identity has no capability to express any linkage to any other identity domain. It has no internal structure of sub-fields that could be interpreted as pointers into other identity fields. Its precise semantics are to define an interface to a network rather than the device itself. This issue of identifier scope comes up in link layer security discussions where it may not be the best possible approach to bind master session keys to MAC addresses, rather than some other identity. Another example, in IEEE 802.11i it is possible for a host to have multiple interfaces and therefore there is a significant difference between binding an Master Session Key to a MAC address and binding to a host identity.

This lack of a direct association between an interface's MAC address and a host device has undesirable effects when it has been assumed that a MAC address equates to a host identity. In "Authentication for DHCP Messages " [13] where the MAC address takes on the role of the DHCP client-identifier, or in the administrative model of IEEE 802.11-1999 Wired Equivalent Privacy (WEP) [23] it can be an administrative burden to keep track of all the network interface cards, their MAC addresses and their associated secrets.

Even despite these limitations, the MAC address is regarded as a useful identity mechanism in the context of an identity space. The original 48 bit identity specification has been augmented with 16 padding bits in order to be incorporated into the IEEE 64-bit EUI-64 global identifier structure, which in turn has been incorporated into the IPv6 address architecture as the interface identifier component of the unicast address [9].

It should be noted that this latter action of embedding one identity (a MAC address) in another (the IPv6 address) lifts the original identity outside its original context. There have been some concerns noted where public disclosure of the MAC address within every IPv6 address also discloses both the unique identifier and, potentially, the role of the device. For example, a device manufactured by a specialized storage manufacturer is more likely to be a very expensive storage subsystem housing mission-critical data. This may not be information that is intended to be made public, and a follow-up proposal advocated the ability for the interface identifier within an IPv6 address to be a temporary randomized value [12]. This is illustrative of one of the side-effects of identity interdependence where one identity realm is embedded in another.

2.1.1 Summary

Uniqueness:
MAC addresses are globally unique

Consistency:
Mac addresses are intended to be a unique token for a network interface. Within this limited scope they are consistent.

Persistence:
MAC addresses are associated with an interface, in the hardware, and, as such, are persistent. There are uses of MAC addresses that do not assume permanence, but that has more to do with the impermanence of binding that address to some other identifier (e.g., IP address) than anything else.

Trust, Robustness:
Both Trust and Robustness seem to be tied to the use of the MAC address, as there is no Internet infrastructure for assigning or reporting them. This raises the question of whether the process that asserts the MAC address of a hardware element and the transmission of that assertion is trustable, and whether the the manufacturing process that embeds MAC addresses into hardware is always reliable.

Withholding:
MAC addresses are not decomposed, and are completely exposed.

Referential Consistency:
The MAC address refers to an interface, rather than a device or an endpoint of an application. If the hardware component is moved then the the MAC address is still a valid identifier for the interface. However, from the perspective of dependent identifier systems, there may be some consistency issues, in that another system may no longer be able to see the interface identified by the MAC address.

Structure:
MAC addresses are unstructured.

2.2 IP Addresses

Moving up one level in the protocol stack model provides an identity based on the internetworking layer, namely the IP address. The IPv4 address is a 32 bit field providing each Internet-connected interface with a unique value. IPv6 uses effectively the same construct, using a 128 bit identity domain rather than a 32 bit domain. In both cases the IP address is a structured identity space where there is a

globally significant prefix that is used in the context of routing and forwarding outside of a particular local domain, and a local part that is used to deliver the packet to the correct interface of the associated device within the local network. The fact that the structure of the address is based on the requirements of routing, and is therefore topologically sensitive, implies that the underlying semantics of the IP identity can be most reasonably assumed to be temporal rather than persistent.

As an identity token, an IP address should be unique. It is structured to be useful to forward packets to the addressed device, and it's well known, in that it's not a secret value.

An IP address is not everything one could hope for in an identity. The IP address identifies an interface, not a device or its user. A device with multiple active interfaces has multiple IP addresses, and while it's obvious to the device itself that it has multiple identities, no remote party can tell that the multiple identities are in fact pseudonyms, and that the multiple addresses simply reflect the potential for multiple paths to reach the same endpoint.

Furthermore, the IP address is an information-bearing identifier, which is structured in such a way that it can be used in routing and forwarding. This is helpful in the sense that there is no need to deploy a second identity system that refers only to locality within a network, however it compromises the use of the address as an identity, since in some circumstances a change in the connectivity of a local network will require a renumbering of that network, such that the address of each individual device will change.

This is a specific example of the more generic observation about IP addresses, namely that the IP address carries both the identity of the endpoint in the IP realm and the location of the endpoint in the IP network. It is a matter of longstanding study that continues today as to the merits of delineating these two roles of identity at the IP level, creating one identity realm as a means of uniquely identifying an instance of a protocol stack within an end device (variously called a " stack identifier" or "endpoint identifier" in previous studies) and a second identity realm that is used to identify the current location of the identity element within the network (typically called a "locator" identity) [1][25][5].

2.2.1 Summary

Uniqueness:
With the exception of certain identified special cases, such as private addresses [4], IP unicast addresses are globally unique. In the context of anycast use of IP addresses, an IP anycast

address represents a collection of individual entities that undertake an equivalent function. In the context of multicast IP addresses, an IP multicast address represents a set of many independent hosts.

Consistency:

Mostly consistent; private addresses are known by convention, not by any internal identifier structure.

Persistence:

IP addresses are intended to be persistent. Becuase of the duality of the address as representing both identity and location, a change in location, such as in a mobile device, often triggers a change in IP address.

Trust:

There is no systematic method of validating an assertion of an IP address.

Robustness:

Attempting to hijack an IP address also requires some form of corruption of the routing system on order for other system to be informed of the updated location of the corrupted address.

Withholding:

IP addresses cannot be decomposed, and are completely exposed.

Referential Consistency:

It is normally the case that IP addresses are referentially consistent, and one party can pass a reference to a correspondant party to any other party by means of passing the IP address. One caveat is that where the IP address is deliberately corrupted, by viture of the use of a NAT device, or in the case of dynamically addigned addresses, then IP addresses lose their referential consistency. As noted above, anycast addresses may not preserve precise referential integrity.

Structure:

The structure of an IP address refers to a structure of topology in a locational context. In order to provide effective summarization tools in the context of routing, the IP address is structured such that adjacent devices use adjacent address, such that a common address prefix can be used to summarize the location of a local network of addressed devices.

2.3 Service and Session Identities

In the TCP/IP protocol suite the next level of identity is that of

the transport session. In order for a system to advertise a particular service that is a point of attachment for clients it combines three fields: IP server address, transport protocol identity, and the address of the local service identity (port number) into a compound identity that describes a particular service port on a particular device.

The port address concept, used in TCP and UDP, represent generic identities for service rendezvous points. When combined with an IP address they become particular service points, or, identified service points, and these compound identification objects (IP address, Transport Protocol, Port) are service identifiers.

The identity concept for transport is further extended by including the sender's IP address and port address. The corresponding 5-tuple of (Source IP address, Destination IP address, Transport Protocol, Source Port, Destination Port) is an identifier for a particular instance of a session. Not only is this 5-tuple used at the destination point to correctly de-multiplex an incoming packet stream and send each packet to the correct local instance of the application, the session identity can also be used within the network to recognize a 'flow' of packets that require identical forwarding treatment and may require identical service treatment, if so configured. In the latter case the session identity is being used to trigger a particular service response within the network, and the assumption being made within such contexts is that this 5-tuple is sufficiently unique to identify particular sessions to the relevant network elements. (SCTP also has a port address, but uses a set of IP addresses to identify the remote end. At the network level a 'flow' or 'stream' is identified as a collection of 5-tuples, rather than as a single 5-tuple.)

There are circumstances where the complete 5-tuple is not visible to the network, such as in the use of IPSEC [8]. It is an objective when using the Encapsulating Security Payload (ESP) protocol when confidentiality is enabled to hide session information. The objective of the deliberate attempt to occlude these details is in order to impede traffic analysis or greatly reduce the information obtainable via traffic analysis. When using IPSEC with ESP the user has choices about how ESP is deployed. One choice is to use a separate Security Association for each flow, while another choice is to use a single Security Association for multiple flows to hide that flow information. It is not uncommon to use multiple already encrypted flows and re-encrypt them together using a common Security Association. This technique is very effective in impeding or preventing traffic analysis. The triple (source IP, dest IP, Security Parameter Index (SPI)) will identify the full flow granularity that the user intended to reveal. Of course, the SPI

value will change at key rollover time, but usually the packet patterns (size, frequency of transmission, etc) will reveal which new SPI value corresponds with which previous SPI value. So if an entity is trying to identify flows, it is best to use that natural triple in the case of IPSEC with ESP.

Session identities are intended to be unique at any single point in time, in that two distinct sessions will not share a common session identity. However, this identity is temporal, in that once the session is finished the identifier is no longer of direct relevance, and at a subsequent time a different session may use the same 5-tuple. As well as impermanence, session level identifiers exhibit a very fine level of granularity, and as such are often at a level of detail which is too fine to be a useful general identity token across the entire Internet realm. One use is to allow a session to construct an identity that refers to itself or its correspondent that can then be handed into a quality of service policy controller to request a specialized service response for the session. Other uses of session identities can be found in filters, firewalls and network address translators, as well as various forms of middleware applications.

2.3.1 Summary

Uniqueness:
A session identity is unique in a very limited context, such that the session identity is only unique between the communicating endpoints, and only unique for the lifetime of the session.

Consistency:
A session identity is intended to be consistent within the scope of the IP-level multiplexing and demultiplexing function performed in the endpoints of the session. Some forms of active middleware attempt to use this session level identity as a means of session identification. This use out of intended context is not always reliable.

Persistence:
Session identities are not persistent

Trust:
Session identities are not necessarily trustable. Additional mechanisms would be required to improve the trust attributes of session identities.

Robustness:

Session identities are not robust, and some other form of session context is required to minimize the risk of session hijacking.

Withholding:

Session identities are an instance of withholding, in that an end point session state includes a number of additional information items relating to packet sequence numbers and endpoint protocol state. These items are withheld from the explicit protocol exchange and are inferred at each end from the protocol exchange.

Referential Consistency:

Session identities are not referentially consistent.

Structure:

Session identities have a number of components, but each component is is not internally structured.

2.4 Routing and Forwarding Identities

As mentioned above, IP addresses provide information required by routing and forwarding systems. Forwarding is undertaken using the entire address as the lookup function into a forwarding table, using the best match of the address against a table entry as the basis of the forwarding decision (where 'best' refers to a precise match across the longest sequence of leading bits). Routing within the Internet uses a hierarchy of environments, ranging from a non-routed multi-access local network, through a set of locally routed networks where routing is based on comprehensive knowledge of local network topology, through to the interdomain routing environment, where routing is based on a sequence of edge-to-edge transits across domains.

This hierarchy of routing is reflected in the structure of addresses. At any point in the routing hierarchy an address is divided into two parts, a routing network part and a subnet address part. Early definitions of this address structure used a fixed division, while later refinements of classless IPv4 addresses and IPv6 both use an explicit prefix length value that is combined with an IP address prefix to form the routing identifier.

Interdomain IP routing incorporates both routing identifiers and routing domains, or "autonomous systems". Within a given routing domain, IP routing is performed using only routing identifiers. However for routing between domains, IP routing is performed using a new identity, the Autonomous System number. The most common implementation of inter-domain routing is a distance vector distributed computation of inter-domain topology using vectors of AS

numbers as both a loop detection and a path preference mechanism. The AS identity space is an unstructured space of numeric values, allocated from a single 16-bit identifier space.

An IP address is located within a routing system by identifying the most specific enclosing routing identifier. Forwarding a packet to a specific IP address involves an algorithm of locating the associated routing identifier and undertaking the forwarding action associated with that object. Coherency of the routing system demands that routing identifiers are managed in a consistent fashion. The overloading of an IP address as both an IP identity and a component of a routing identifier implies that a device's location is implicitly described by its IP address. As noted earlier, relocating a device to a new network location, or relocating a network to a different point in the overall Internet topology necessarily implies associating a new IP address with the device. In the absence of any other mechanisms, this new IP address replaces the previous IP address, changing the device's IP identity, the device's service identities and the device's session identities.

2.4.1 Summary

Uniqueness:
Routing identities are intended to be unique, deriving their uniqueness from the underlying properties of the IP address space.

Consistency:
Routing identities are intended to be a unique token within the context of a routing realm.

Persistence:
Routing identities are not persistent, and have a liketime associated with connectivity of the described entity within the routing realm.

Trust:
There is no systematic method of validating an assertion of a routing identity.

Robustness:
The identity structure does not inherantly prevent various forms of corruption of routing identities

Withholding:
An inter-domain routing identity is a compund entiy consisting of an address prefix and an autonomous system number. There is no withholding of elements of this identity.

Referential Consistency:

Routing identities are intended to be consistent within a routing realm, and the operation of routing protocols rely on this referential consistency of routing identities.

Structure:

Routing identities are not internally structured.

2.5 Mobile Identities

Device and network mobility adds an additional dimension to identity, in that mobility implies some level of decoupling of the notion of location with that of identity. In one form of approach to this generic space, that of device mobility, a device has an additional IP address that acts as a 'current locator' that describes the device's current location within the network, while the device also retains a constant 'home address' that in effect acts as the device's constant identity and also acts as the discovery service point for its current location. With this approach a 'home agent' acts as a proxy agent for the device when it is roaming beyond the confines of its local network. The home agent tunnels traffic sent to the home address to an address at the host's current topological location, called the 'care of' address in Mobile IP. The host is responsible for updating the binding between the home address and the care of address in the home agent, by sending a binding update message when the care of address changes. The mechanism involved in mapping between the home address and care of address is very similar to the mechanism used on the local link for the ARP neighbour cache, except IP addresses are involved for both.

This approach raises a critical issue for identities, namely that of robustness. Approaches to mobility need to be aware of a potentially hostile environment where third parties may attempt to subvert the implicit redirection of traffic by assuming the identity of the mobile element through the generation of false updates of the current location.

2.5.1 Summary

Uniqueness:

A mobile identity is a compound entity using two IP addresses: a 'home' address that functions as an endpoint identifier and a 'core of' address, which functions as a current endpoint location identifier. A mobile identity is intended to be unique.

Consistency:
A mobile identity is consistent within the realm of mobile IP protocols.

Persistence:
A mobile identity is not persistent,. The endpoint identity value is intended to be persistent, while the endpoint location identifier is only intended to be valid while the mobile entity is located at the specified 'care of' location.

Trust:
Of itself a mobile identity is not trustable. Mobile IP protocols add additional communication exchnages between the mobile entity and 'agent' entities in order to create trust in a mobile identity.

Robustness:
A mobile identity is not intrinsically robust. The protocol exchanges between the mobile entity and its agents can create a robust mobile identities.

Withholding:
Manipulation of mobile identities can include deliberate withholding of the current location information, or the persistent identity information.

Referential Consistency:
Becuase of the temporal nature of the location identifier, mobile identities are not consistent over time.

Structure:
The structure of a mobile identity is derived fromt he structure of the underlying IP address space.

2.6 Opportunistic Identities

This concept of maintaining some form of identity association in the face of a communicating within a potentially hostile environment has lead to a proposal for an identity token that has its roots in the public / private key pairs. In this approach the identity token is associated with the public key value of a public / private key pair. A message encrypted with a private key can be passed to the other party where only the originating party's publicly asserted identity (or public key) can decrypt the message.

Such identity realms can serve to support a reliable assertion that the received message originated from the same party that originated the communication and that the message has not been tampered with

while in transit. The identity systems are opportunistic in that they are self- generated identities, and have no external structure. The implication is that such identities have no particular structure and may not be completely unique. For this reason their utility in other identity applications where persistence or referential integrity is required, such as acting as a persistent reference to other attributes of a named object, is limited.

2.6.1 Summary

Uniqueness:
Opportunistic identities are not unique.

Consistency:
Opportunistic identities may not be consistent.

Persistence:
Opportunistic identities are not necessarily persistent.

Trust:
Opportunistic identities are not trustable in general. There may be limited contexts in which an opportunistic identity may be considered trustable..

Robustness:
Oppostunistic identities are normally robust in the sense that they are not generally divulged, are generated in a manner that is systematically predictable by a third party, and are often drawn from a sufficient large space that they are resilient to guessing techniques. In this sense an opportunistic identity can be considered to be robust.

Withholding:
Opportunistic identities can be simple or compound tokens. Withholding is possible in the case of compound identity realms.

Referential Consistency:
Opportunistic identities are normally bounded by a particular context of use, and are not referentially consistent outside of this context.

Structure:
Opportunistic identities may not necessarily be structured.

2.7 Domain Names

The set of identities described so far have no particular human-visible aspects of their function. The identity tokens are

structured to meet a particular purpose, and are not intended, as their primary purpose, to be manipulated by humans nor are they intended to be used primarily within the realm of human discourse. By contrast, the Domain Name System (DNS) was specifically intended to be a name realm that is suitable to be included in human discourse, yet at the same time to admit enough structure to be manipulated by computer applications in a deterministic fashion.

The DNS is essentially a hierarchical name space, where the hierarchical name structure allows the space to be efficiently searched and managed in a distributed fashion, but also supports one of the most desirable attributes for an identity space, namely uniqueness. The explicit hierarchy also assists in ensuring uniqueness, as DNS names are intended to be unique across the entire name string rather than just at the first component, so that "a.b.c" is a different identifier to "a.d.e " even though the first token in the domain names, "a", is the same in both cases.

The most common use of the DNS is to map domain names to IP addresses, but other uses are possible via mapping a name to a number of other defined 'resources'. The core functionality of the DNS is that of a unique, structured, name space and a mapping capability that allows a query to be performed to retrieve the mapping information for a DNS name for a particular class of resource mapping.

The Domain Name System is more than a set of syntactic rules for constructing a well-formed DNS name. The resultant name, if well constructed and properly implemented, can be used as a referral token to a service environment. In this fashion the DNS encompasses a translation service that maps from domain names to defined resources, including IP addresses. For example, given a well formed DNS name, a DNS lookup can query for a corresponding IP address. The DNS describes a data model, a set of relationships between data objects as well as a protocol used to send queries and receive answers.

As DNS names provide a mapping from a name to a resource, the name does not need to change when the resource changes location or some other identifying attribute. The mapping changes, but the name remains constant, and for this reason domain names can be considered to be stable unique identifiers, residing within a structured space that can be efficiently searched and managed in a highly distributed manner.

2.7.1 Summary

Uniqueness:
DNS identifiers are unique.

Consistency:
DNS identifiers are intended to be consistent. There are a number of issues relating to character equivalence within various languages that impinge upon consistency of interpretation in some contexts.

Persistence:
DNS identities are intended to be persistent.

Trust:
The trustability of a DNS identity is based on the integrity of delegation within the hierarchy of the DNS identity.

Robustness:
The DNS is implemented as a distributed name database, and the robustness of the DNS is based on the robustness of this database.

Withholding:
DNS identities are capable of wothholding. A DNS identity can be regarded as a DNS name, and an associated set of resource records. Resource record values are withheld unless explicitly requested as part of a resolution query.

Referential Consistency:
DNS identities are intended to the referentially consistent.

Structure:
The DNS name space uses a hierarchical structure.

2.8 Uniform Resource Identifiers

When communicating, applications often need more information than a domain name. For electronic mail, for example, the sending application must use a combination of the domain name, the TCP protocol, the mail delivery or mail agent's service port and the mailbox name of the recipient. Other applications require different compound identification objects, in accordance with their characteristics. This compound identity may be specified in the format of a Uniform Resource Locator, or URL.

Uniform Resource Locators (URLs) are a subset of a more generic form of resource identification, Uniform Resource Identifiers (URIs). As an identity space, the URI space is very loosely defined, and it's quite remarkable as to the extent to which it has spread across the world as a form of object identifier, or identity token. URLs refer

to the subset of URIs that identify resources via a representation of their primary access mechanism. Other forms of URIs provide resource identification through a name scheme or by other attributes of the resource.

There are few syntax rules to the Universal Resource Identifier space, and only a small amount of common semantic structure. The original IETF documentation, RFC 1630 [2], refers quite simply to a syntax of a prefix word, a colon, and a following string. Where there is hierarchy in the following string, slashes are used to delineate the hierarchical levels, and the hierarchy runs from left to right. The current generic syntax of URIs is described in RFC 2396 [7], and the only change to this generic syntax is to refer to 'schemes', as in "<scheme>:<scheme-specific-part>".

The common usage of URIs has been more structured than this general specification, and most URI schemes do not provide a single string that is an alias for an identity, but instead form an identity from the instructions that specify how to access the resource, in the same way as a postal address is often constructed from the instructions as to how to deliver a postal letter to you. This form of a URI, which can be viewed as a location specification, is the basis of the URL scheme. In other words such protocol-scheme URLs consist of what could be interpreted as a device selector, an application selector and an application-specific string that acts as an object reference.

Within such protocol-scheme URLs the scheme prefix is an identifier that uniquely identifies the service being referenced, or in terms of access it references the protocol and port address to be used. The first, or top, level of the hierarchical following string is either the DNS name of the server, or the DNS name coupled with some specific qualification, such as a mail address. Any subsequent hierarchical components represent service-specific instructions to be specified that lead you to the referenced object. Thus we have "mailto:user@domain.example.com" for a mail specification, where the scheme prefix "mailto" identifies the use of the TCP transport protocol, a port address of 25 and a protocol of SMTP. The following string, "user@domain.example.com" references the mail agent (a DNS lookup of "domain.example.com" for an MX resource record) and a value to be used in the protocol exchange (delivery to the mailbox "user@domain. example.com"). Similarly, "http://www.example.com/directory/hierarchy/index.html" for a specific web page uses "http" as a scheme identifier for TCP, port 80, protocol HTTP, the initial part of the following string to reference the server (a DNS lookup for an A or AAAA resource record for "www.example.com") and an HTTP protocol request for "www.example.com/directory/hierarchy/index.html".

In this form of the URL identity system uniqueness is keyed from the general use of a DNS name within the URL, and the wrapping around the DNS string is taking the general form of the DNS as an alias for an IP address, and, additionally, specifying a service point, and then arguments that are needed to provide to this service point in order to retrieve the referenced resource. In that way a protocol-scheme URL is closer to a description of an algorithm than to an identifier whose structure of the identifier is adapted to tasks such as sorting, searching or equivalence operations. There are issues with consistency here in that while the hierarchically structured string set makes sense to one application it may not make any sense in the context of a different application.

The persistence of protocol-scheme URLs is also an issue, in that the resource may change location over time, and the corresponding algorithm to locate the resource, or URL, must necessarily change as well. The other major difference between a structured identifier space and the protocol-scheme URL approach is that the structured identifier space requires some form of lookup to apply the identity into a retrieval system. By changing the outcomes from the lookup operation, the identity owner can track changes in the location of the resource. In the protocol-scheme URL approach there is no way to understand how widely the identity has circulated, and it is not possible to update the in-circulation copies of the URL. The property of the DNS is that in itself, the DNS identities are simple structured tokens, and they require a lookup operation to be performed in order to produce an algorithm that allows an application to refer to a particular object. While such protocol-scheme URLs are widely used as service and resource identities, they pale in significance, persistence and utility when compared with DNS names. In other words URLs specify "how" to access a service, while generic DNS names can be interpreted as identity tokens that can be used to identify a resource that may host a service (or "who").

It is also not surprising from this perspective to see the emergence of DNS resource records that refer to URLs, as in NAPTR records [10]. In this approach the first DNS lookup retrieves one or more URLs that have been associated with the DNS name, and a second lookup is used to resolve any DNS names as may be referenced in the URL strings. In this framework a service may change its location, or the access algorithm may be altered (and by necessity, the URL changed), but the DNS identity that maps to this URL remains constant. This is one of the clearer forms of delineating identity from access mechanisms.

This mapping can also be used for service discovery. Given the name of a domain it is possible to look up NAPTR records to discover what URLs can be used for communication with that domain. This is for example used in the ENUM specification [11]. In ENUM a lookup in DNS

of NAPTR records for a domain name created from an E.164 number is via transformation turned into a list of URLs. This give an ability to know what URLs one can use in order to contact the entity referred to by a given E.164 number. The more general form of this approach can use NAPTR resource records to associate a DNS name with one or more resources. The name that has the NAPTR records can be considered as an identity token, while the associated NAPTR records provide a mapping from this identity to the instantiation of the identified service. This approach has been used in the Archive Resource Key (ARK) proposal [26].

Of course not all URIs are protocol-scheme URLs of the form outlined above. URIs are a very general construct where the initial "scheme" part of the URI determines the structure and semantics of the remainder of the URI string. The next section examines that class of URIs where persistence of the identity is a specific feature of the identity realm, the Uniform Resource Name.

2.8.1 Summary for URLs

This summary section explicity refers to URLs rather than URIs. The more general case of URIs is one that, in the general case, is unclear on all these desireable identity attributes.

Uniqueness:
URLs are intended to be unique.

Consistency:
URLs are not necessarily consistent. A URL typically includes the specification of an application, an enpoint and some additional arguments for the application to apply to the application instance on the nominated endpoint. Inconsistent interpretation of URL components by other applications is possible.

Persistence:
URLs are not necessarily persistent, as they implicitly identify how to access a resource or service, rather than identifying the resource or service per se. If the service or resource changes location, a new URL is required to reference the new location. In the case where URIs use a DNS identifier as part of the URI scheme, as in URLs, for example, such URIs also depend on the persistence of the underlying DNS identity for persistence of the URL.

Trust:
URLs are not necessarily trustable.

Robustness:
URLs are not necessarily robust.

Withholding:
URLs are not capable of withholding elements of the URL identity.

Referential Consistency:
URLs are intended to be referentially consistent, but are limited in terms of their persistence.

Structure:
URLs are a structured identity space.

2.9 Uniform Resource Names

To solve the problem of lack of long term stability for references, URNs can be used as an alternative to recursive references into the DNS. URNs are generally considered not to be entirely within a human realm as they often include what would appear to be long random combination of characters. URNs are intended to be globally unique, and never reused. As long as a named object exists, it retains that name. An object can have many names. The object may cease to exist, in which case the URN can no longer be resolved, because the resolution service (from URN to URI) is no longer working, but, as the name exists (virtually), a new service can be created and the object re-established if there is need for it. RFC 3305 [14] describes in more detail the different views that exist on the relationship between URIs, URLs and URNs.

2.9.1 Summary

Uniqueness:
URNs are intended to be unique.

Consistency:
URNs are indended to be consistent.

Persistence:
URNs are intended to be persistent.

Trust:
It is unclear how trust relationships are formed with URNs.

Robustness:
As with trust relationships, the robustness properties of URNs are unclear.

Withholding:
It is unlikely that URNs can withold parts of the URN.

Referential Consistency:
URNs are intended to be referentially consistent.

Structure:
URNs included unstructured components.

2.10 Human Friendly Strings

URIs have a problem that URNs didn't solve, and that is the ability for humans to remember them. Humans act in a context, so global uniqueness is not important at this level of abstraction. Instead, when a human uses a name, they normally want a resolution service that "does what they want". In this realm the context of the name is an important factor in resolving the name to an object, and global uniqueness is neither necessary nor assumed.

This area of human friendly strings is a topic of ongoing work. One possible goal for a working system is to be able to handle the so-called "side of the bus" problem. A human sees something in an advertisement on the side of a bus, remembers it (or remembers part of it), and when they come to a computer they try to get more information about what they have seen. This involves complex language and localization (and internationalization) problems.

There has been various ideas connected to "layers above DNS", for example mentioned in RFC 3467 [19] (subject of the SIREN Research Group in the IRTF). This topic encompasses an effort to decouple the naming realms that makes sense to humans, with their various forms of implied context for resolution, from the naming realms that work for computers, with the implication of explicit specification of resolution, and define a mapping between them. The DNS can't handle the types of names that often make sense to people, because people always work in a context (such as a geographical context of 'locality'), and it's no longer sufficient for people to fit their needs into what DNS can handle. For a some time it was considered possible to overload the semantics of the DNS label (machine-parseable, vaguely human-recognizable) but it is becoming evident that this is not a tenable approach, and some distinction needs to be drawn between DNS names and context-sensitive human-friendly strings.

No real human friendly naming system exists today on the Internet.

2.10.1 Summary

Uniqueness:
HFS are intended to be unique within a context of discourse.

Consistency:
Unclear.

Persistence:
Unclear, although it would appear to be a desireable attribute in this context.

Trust:
Unclear.

Robustness:
Unclear.

Withholding:
Unclear.

Referential Consistency:
Desireable.

Structure:
Unclear.

3. Issues with Identities

3.1 Overloading the IP Address

An IP address suffers from semantic overload in attempting to carry both location and some form of constant identity. If a network or individual device changes access providers then this is, in effect, a change in network location, and if provider-based address aggregation is being used, then the local IP address will change. The same issue applies with mobile devices. This implies that an IP address is not necessarily a permanent or truly persistent association with a device, and such impermanence is a weakness in any persistent identity system.

Another issue with IP addresses, at least in version 4 of the protocol, is that of their total span. While 32 bits is still a large size, encompassing some 4.4 billion unique addresses, there is an inevitable level of wastage in deployment, and a completely exhausted 32 bit address space may only encompass a connectivity realm of perhaps only 1 or 2 billion IP devices. When this is coupled with a world of embedded IP devices in all kinds of

industrial and consumer applications, 1 or 2 billion addresses is insufficient to provide unique addressing to every possible device.

In response to these address pressures there has been the introduction of a number of technologies that dilute the strong binding of IP address with identity. Such approaches tend to treat the IP address purely as a routing and forwarding token without any of the other attributes of identity, including persistence and public association. For example, DHCP, or address-lending, is a commonly used method of extending a fixed pool of IP addresses over a domain where not every device is connected to the network at any given time, or when devices enter and leave a local network over time and need addresses only for the time they are within the local network's domain. In this form of identity, the association of the device with a particular IP address is temporary, and hence there is some weakening of the identity concept, as the dynamically-assigned IP address is being used primarily for routing and forwarding. This has been taken a further step with the use of dynamic Network Address Translation (NAT) approaches, where a single device has a pool of public addresses to use, and maps a privately used address to one of its public addresses when the private device initiates a session with a remote public device. The private-side device has no direct knowledge of the public address that the NAT edge will use for the session, nor does the corresponding public-side device necessarily know that it is using a temporary identity association to address the private device.

These forms of changes to the original semantics of an IP address are significant architectural changes to the concept of identity at the level of IP, particularly in the presence of NATs. The widespread deployment of such approaches continues to underline the concept that as an identity token there is a lack of persistence in an IP address, and the various forms of aliasing weaken its utility as an identity system. The conclusion drawn from these observations is that, increasingly, an IP address, in the world of the IPv4 Internet, is being seen primarily as a locality token with a very weak association with some form of identity.

Version 6 of IP represents an effort to restructure the address field, and the 128 bits of address space represents a very large space in which to attempt to place structure. One of the more innovative concepts that was discussed within the development of IPv6 was extending the concept of the IPv6 interface identifier field of the address to be a globally unique identifier. This had some obvious connotations in being able to identify when the connectivity for a device has changed, as in such cases the globally unique interface identifier could remain constant while the routing prefix may have changed. There was also some potential applications in the

area of supporting multi-homed networks, where a local network could be seen via different routing prefixes. At present these aspects of IPv6 address architecture are the topic of ongoing work in the IETF. One of the fundamental issues with this form of approach is management of an interface identifier space that is globally unique and persistent, as well as being adequately robust. Current directions of activity in this area indicate that the self-assertion of identity using this field within IPv6 are insufficiently robust to prevent various forms of redirection attacks. Approaches currently being investigated are looking deeper into various aspects of mechanisms that are intended to provide corroboration of identity assertion in the face of locator change and additional protocol mechanisms appear to be a common feature of the current proposals relating to multi-homing and aspects of mobility.

3.2 Dynamic DNS Updates and Nomadism

An alternative mechanism to revising the semantics of the IP address is looking at the concept of moving the role of completing the transition of persistent identity into the DNS. Here the constant identity of the device is its DNS name. In a mobile context, as the device or network it roams across the network, and by using a sequence of secure dynamic incremental updates to the DNS, update the association of the constant DNS name to the new local IP address. This approach has possible applications in various multi-homing scenarios.

However, this approach is not without attendant considerations. Much of the leverage of the DNS as an efficient lookup mechanism is based on extensive use of local caching of DNS information. Increasing the responsiveness of the DNS to dynamic updates implies that the extent to which cached information can be retained is compromised, and any cache has to refer more frequently to the primary source to refresh the currency of the local cached copy. The tradeoff here is greater DNS traffic loads and increased DNS server query loads in order to get a more responsive name system. Such a mechanism also requires an "always available" primary DNS server to accept the incremental updates, so that the failure backup mechanism of the DNS with primary and secondary servers is compromised in this nomadic model with the requirement for primary server availability in order to undertake an authoritative update to the DNS.

An alternative approach is to equip the DNS with an additional resource record that contains an identity value in addition to the current A or AAAA address values. This approach can be used in conjunction with an additional element within the protocol stack that could allow the transport layer to operate using this identity field, and a new stack element provides a dynamic mapping between this

identity and a 'current' locator value, where the equivalent current locator is passed into the IP protocol element.

An alternative to this approach of changing mappings is to place the responsibility for the redirection into the application protocol. For example, with SIP, the mobile node could use the REGISTER method to change its registration for session setup. This may not be fast, but may be faster than dynamic DNS updates and perhaps even fast enough for handling initiating new sessions. A mobile HTTP server, on the other hand, would have to use HTTP Redirect from a fixed server whose address was in the DNS.

3.3 URLs and Persistent Identifiers

URLs are, as their name suggests, locators rather than location independent identifiers. When the resource is relocated, or when multiple copies of the same resource exist, the URL scheme cannot persist across the change. Despite the almost universal use of the URL within web browsers, URLs are not an ideal candidate for a persistent identity.

This weakness in the URL scheme has lead to the consideration of many alternate naming schemes, although the underlying requirements for any candidate naming scheme is that it is cleanly mappable into a URI-styled format and that there is a robust resolution system associated with the name scheme. Resolution is a critical factor here, as without the ability operate in a predictable, robust, scalable, trustable and reliable manner when translating an identifier into a resource, entity or service access description, the identifier scheme is of dubious value.

The requirement for persistent identifiers is not intended to dispense with URLs, or similar forms of locators and service descriptors, but to separate the notions of identification and location, and to use distinct label space for each concept, and to use a resolution mechanism to map from the identifier to the location descriptor.

Work on the development of a unique permanent identifier space has proceeded concurrently with the formalization of URL schemes, using the name of URN (Uniform Resource Name) schemes. A specification outlining the minimum requirements of the URN can be found at [3]. The syntax of the URN as expressed in [6] is as follows:

```
   urn:<Namespace Identifier (NID)>:<Namespace Specific String (NSS)>
```

> The NID ensures the global uniqueness of the identifier. The NSS can take any form specified by the naming authority provided that it is unique within that namespace.

The simple structure of the identifier reflects recognition of the need to accommodate different requirements and different schemes. Because the local, or namespace specific, string can be in any form, the identifier structure allows maximum flexibility in the identifier while providing a mechanism to assure global uniqueness and facilitating interoperability between discrete systems.

There is a need to distinguish between naming schemes and resolution systems. A naming scheme, as a procedure for creating unique URNs that conform to a specific syntax, is independent of the resolution service which resolves the URN to locate the resource. Ideally, a naming scheme should not be tied to any specific resolution system and a resolution service should be capable of resolving a URN from any given name scheme.

This objective is consistent with the intentions behind the development of the URN. A persistent identifier, especially when used for archival data must of necessity be capable of outlasting any systems and protocols that are currently in use. However the lack of a commonly agreed upon resolution system is also a major obstacle to the wide deployment of URNs.

A variety of solutions have been proposed, including the NAPTR (Naming Authority PoinTeR) DNS resource record [10], that provides rules for mapping parts of URIs to domain names and then using these domain names as DNS lookup queries to find mapped URIs. This was specification has been further refined as the Dynamic Delegation Discovery System (DDDS) [15][16][17][18]. As noted in RFC3404 [18]:

> "For the short term, the Domain Name System (DNS) is the obvious candidate for the resolution framework, since it is widely deployed and understood. However, it is not appropriate to use DNS to maintain information on a per-resource basis. First of all, DNS was never intended to handle that many records. Second, the limited record size is inappropriate for catalogue information. Third, domain names are not appropriate as URNs. Therefore our approach is to use the DDDS to locate "resolvers" that can provide information on individual resources, potentially including the resource itself."

There appears to be some residual issues over the status of URNs. For URNs to achieve widespread deployment, not only is consensus on functional requirements and syntax needed, but the ability to

recognize and resolve URNs should be incorporated into the application realm. For example, it would be a reasonable objective to incorporate URN support in standard Web browsers. However a pre-requisite for this step is the definition and construction of the necessary resolving infrastructure, developed either by leveraging off the existing Domain Name System or by some other route. As long as application developers are uncertain of what is to be accepted as a standard resolution mechanism, and while naming scheme developers are uncertain of how to register their name and resolution schemes these issues will not be fully resolved.

Until the resolution issues are clarified and there is clear consensus to adopt a particular specification, implementation of URN systems will require some form of application level assistance by way of proxy servers. The implication is that use of URNs will require encapsulation in a URL in order to specify the appropriate proxy server address.

This approach has already been undertaken in the specification of PURLS [24], which is a naming scheme that incorporates within the PURL a conventional URL reference to a resolver to specify a PURL resolution service and a name part of the URL that the resolution service translates to the resource URL. In a web-based context this is handed back to the client as an HTTP redirect. The dependency of the identifier scheme on the behavior of a particular application (namely HTTP in this case) is not the most desireable of attributes for an identity scheme. If the PURL was to be used in a different context by a different application, a comparable redirection mechanism would be required to support the desired outcome.

In comparison, the Handle system [20] uses a non-URL name scheme, and resolution in applications requires modification of the application. The 'handle' itself is a persistent identifier consisting of two parts. The syntax is a two part identifier of "<naming authority>/< name>" where the naming authority is an administrative unit authorized to create and maintain handles and the name of the resource is a string which must be unique to that authority but which has no prescribed syntax. Use of handles can be through standard web browsers using a plug-in, or through unmodified web clients using proxy servers and embedding the handle within a URL that specifies a handle resolver in a manner similar to the PURL approach. The specification of a distinct handle syntax allows handles to be used in a broader set of contexts than web browsing as there is independence of the identifier to a particular access protocol and server location.

The issue of resolution of the compound identifiers remains problematic, and the use of embedding the URN into a proxy URL to

undertake redirection can be argued as defeating the purpose of having location and protocol independent identifiers, since the resultant identifier includes the location of the proxy agent. The full value of persistent identifiers to ensure persistence in citations can only be realized if they are actually useful when citing documents and objects. In order to use them, the user must know that there is a persistent identifier and must be able to discover what it is and how to resolve the identity. At present this is difficult because of the nature of the redirects used in most existing systems.

4. The DNS in Identity Spaces

How good are any of these identities? Which one should be used in which context?

Each of these digital identities have a context of usage, or realm of discourse, and outside of that realm they tend to break down as a cohesive and useful identity tool. Offering a MAC address as an email point of contact makes little sense, even though it could conceivably be used to form a unique identity in the mail realm. Offering an identification at the appropriate level of abstraction that provides a description of the mode of contact and identity in a form that matches the actions at this level is often used to distinguish between identities. At the level of human interaction we commonly identify email addresses using a domain and user name part. We do this because this is what you need to enter into your mail application in order to send me a message.

There are considerations when generating identity spaces based on generic descriptions of algorithms of how to access the specific resource, trigger the particular application or contact a particular individual or role's network point of presence. These considerations, commonly found in conjunction with URI's, raise consideration of maintaining referential integrity, allowing efficient searching and persistence of the identity. The human world, and its digital counterpart, is far from static. Any identity system that aspires to be useful in a human space needs to be able to support a maintenance function that allows any implicit reference that is contained in an identity space to be updated and refreshed in a reliable, trustable and timely manner. Knowing who you were is a less important piece of information as compared to knowing who you are right now. That leads to consideration of structured identity spaces whose two major attributes are:

- sufficient structure to ensure that specific instances of the identity are unique, and

- appropriate structure to allow rapid lookup of the identity to be able to retrieve the current set of associated pointers within various specified realms.

There is a good match between these desired attributes and those of the DNS, and one perspective to be drawn from this is that the major underpinning of useful and lasting digital identities rests within the framework of the DNS. In other words any useful identity space is highly likely to have managerial and operational characteristics that would largely parallel that of the DNS.

4.1 The role of the DNS

Different identities are used in the Internet for different purposes. IP addresses are essential at the level of forwarding protocol data units across the network, but are unwieldy to use in the context of naming resources and services at the level of human operation of applications. In the context of URIs, the use of a DNS identity within a URI ensures that the identity of a service doesn't have to be changed when the IP address changes. The domain name creates an abstraction layer above the IP addresses that allows a service to be identified without particular reference to its current location within the Internet, and using a name realm that has better properties for human use.

We could use something else, like static tables, databases or more similar systems like X.500. But, none of these alternatives have been able to prove they scale as well as DNS. Both the protocol itself and the data model with the distributed delegation has proven to be extremely efficient (even though some things could be "better").

The perspective being espoused here is that we don't have any current viable alternative to the DNS in terms of a structured identity space that supports mapping across identity realms. Even if we stop using domain names in URI's and instead using something else, deploying a translation service from this other name to IP addresses would inevitably involve recreating much of the functionality of the DNS.

4.2 Changing the DNS

Because DNS is the service we use for mapping between many of the namespaces we use on the Internet, it is extremely important it works. Because of this, changes to DNS must be made with care. This refers to both changes to the protocol as well as the DNS data model.

Example of changes to the protocol include the need for DNSSEC

(signed record sets) which makes it possible for a recipient of a DNS response to verify whether it comes from an authoritative source. This has been discussed in the IETF for some years, and is illustrative of the required level of care in the design of changes to the DNS.

Example of a change to the database structure include a move from an hierarchical to a flatter namespace. The result might be a disruptive change of DNS traffic on the global Internet which in turn might make further scaling difficult. Another similar change is allocation of names which are not registered properly. Especially in the root zone, this leads to problems such as the inability to later allocate and set a policy for the domain, and increased number of queries for non-existing names in the root zone when leakage of names happens from presumed to be closed networks. Example of the former are the very large TLDs like .com and .de. Example of the latter is the use of the pseudo TLDs '.local' and '.gprs' which are being used in private or enterprise contexts without any proper definition or registration and their consequent leakage of queries into the 'public' DNS.

4.3 The DNS is a strict lookup service

When sending a query to a server, the server is to send the same data back regardless of context. Further, the server should send either a "match" which consists of one or more resource records, or a "failure" which include the special response "no such domain".

This implies that two users sending the same query from two different locations at the same time should receive the same data in response. Or, the same user using two different computers with different operating system should receive the same data.

Having the DNS server doing a "search", undertaking "fuzzy matching" or inferring some additional context to a query that guides the server to choose a particular response is ill-advised. The DNS server can not know the context of the query, nor should it guess what the DNS response is to be used for. It is always tempting to assume that the response is to be used by the most popular operating system for the most popular application of the day. It must though be remembered that other operating systems and other applications might break when fuzzy matching happens. For example, instead of giving back a "no such response" it is conceivable to give back something which pushes a potential error to the application layer by returning a synthesized answer that has resource records pointing to some form of application- level service. This implies the DNS server must know what application layer protocol is in use, and that a "no" at the application layer has the same semantics as a "no" on the DNS

(naming) layer. Often TCP is used at the application layer which implies a "no" might only be signalled to the other end by not accepting the connection, which means the querying client cannot differentiate between "no such (dns) name" and "no response in application protocol".

4.4 Coherency of the DNS

DNS is a bootstrap mechanism that publishes your data in a manner that allows queries from others to be answered. If you make mistakes in your local DNS configuration then you don't destroy the utility of the DNS for yourself, but you destroy the ability for others to contact you. Someone trying to reach your webpage might not be able to do so as they can not find the proper translation from your domain name to the IP address of the web server. It is also the case that mis-configurations most often happen in the glue between parent zone and child zone, and not in the child zone itself. Because of this, if you know where your nameserver is, you might not see the errors, as they have to do with finding the nameserver, and not the content of it.

As mentioned before, it is very important the same response is sent back regardless of from where it is sent. The assumption within the DNS is that you should be able to pass a URI with an embedded domain name in it to all of your friends, and they all should be able to resolve it in an identical fashion. It is extremely important the domain names are globally unique, and lead to the same result every time, and from every location.

Part of the coherence requirement is that the servers must be able to give back the same response to the same query. The implication is that all servers have to use the same matching algorithms when attempting to locate a match between a query and the local data used to form a response. What matching algorithm is used when looking in the data cannot change between servers because then they will give back different results for the same query.

Complications arise when considering this in the context of use of various character sets within the DNS. Having each server use a local set of rules that defines equivalence of characters generates the situation of the same query generating different responses. The implication is that the consideration of different matching/equality rules can be solved by creating "bundles" of characters which are to be treated as equal, and solving the problem at the time of registration. This gives a greater choice for the registrant, and it can also give a higher freedom regarding context, as the bundles possibly look differently depending on such things like (parent) domain and language.

4.5 The DNS as an Identity Glue

When comparing the desired attributes of a useful identity system to the properties of the DNS it is evident that there is a reasonable level of fit between the DNS and a generic identity realm. The DNS provides a namespace that ensures uniqueness, is consistent, can support persistence, and referential consistency. The space is structured in a manner that supports relatively efficient lookup over a large name space that has both hierarchical structuring and within that some areas of large flat name spaces. The DNS can support trust models in terms of being able to validate the authenticity of responses. The DNS can support a variety of resource records that allow a DNS name token to be used as a search object that can map to related values drawn from other identifier realms, as well as supporting indirect self-reference through the use of NAPTR records and URIs.

There are obvious trade-offs in the design, protocol and deployment of the DNS in terms of resiliency, dynamic behaviours and scalability. While it is not argued here that the DNS represents the only optimal trade-off between these properties, it is argued that any other identity space with similar properties will be faced with precisely the same set of trade-offs. It is also probable that any similar identity space faced with the same requirements of scalability, operational performance, accuracy and validity of responses and flexibility of mapping the identity space to related objects in other identity realms would find a resolution between these requirements in a manner that would not differ markedly from the DNS.

The salient observation here is that an identity system acts generically as a reference to an initial point of rendezvous in a communication transaction. In this vein the role of the identity system is to identify how other parties in the network can refer to the identified element using an identity token that is persistent, with associated referential mappings into other identity realms that reflect the current status of the element. Once a communications state has been established using the rendezvous points, if there are characteristics of the application that require the subsequent exchange of information (such as location changes in a mobility environment, or a server hand-over at the application level) this is generally the task of components within the protocol stack, using a trust relationship between the communicating parties to alter the identity elements used within the stack to match the changing characteristics.

5. Security Considerations

Any identity system that provides a mapping from an identity value within one realm to an identity value (or set of values) within another realm will present a number of considerations with respect to security. The trust model for an identity system is that the mapping supported by the identity system is authentic, and that when the identity value is used as a key in a query operation, the response should be an accurate response that correctly represents the mapping originally provided by the assigned holder of that identity value.

Equally, it is necessary to correctly report responses where an invalid or unassigned identity value is used, providing the query agent with a clear indication that the identity value is not assigned.

In a hierarchically structured identity space there are a number of potential weak points in the identity space, where vulnerabilities exist for third parties to intercept queries and substitute a non-authentic response. This could involve misrepresentation of the of the root servers for the hierarchy, or misrepresentation of delegation points, as well as misrepresentation of responses for particular mapping queries.

Any design of an identity space resolution service should be resilient to these forms of attack, by using appropriate mechanisms to reduce the risks of interception and misrepresentation in identity resolution operations. However, recognizing the lack of absolute assurances that a resolution system is resilient to all forms of attack, a resolution services should also be capable of exposing the trust model that exists within the identity space, and allow a user of the resolution service the ability to validate the response against the trust model. In other words authenticity should be a verifiable quality of the identity realm, rather than simply being an assertion that is interpretable only as a article of faith.

6. Acknowledgements

The editors acknowledge the contributions made by Ran Atkinson, Brian Carpenter, Vint Cerf, Leslie Daigle, Joel Halpern and James Kempf in the preparation of this document.

7 Informative References

[1] Saltzer, J., "On the Naming and Binding of Network Destinations", RFC 1498, August 1993.

[2] Berners-Lee, T., "Universal Resource Identifiers in WWW: A

Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web", RFC 1630, June 1994.

[3] Sollins, K. and L. Masinter, "Functional Requirements for Uniform Resource Names", RFC 1737, December 1994.

[4] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G. and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[5] Carpenter, B., Crowcroft, J. and Y. Rekhter, "IPv4 Address Behaviour Today", RFC 2101, February 1997.

[6] Moats, R., "URN Syntax", RFC 2141, May 1997.

[7] Berners-Lee, T., Fielding, R. and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[8] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[9] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[10] Mealling, M. and R. Daniel, "The Naming Authority Pointer (NAPTR) DNS Resource Record", RFC 2915, September 2000.

[11] Faltstrom, P., "E.164 number and DNS", RFC 2916, September 2000.

[12] Narten, T. and R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 3041, January 2001.

[13] Droms, R. and W. Arbaugh, "Authentication for DHCP Messages", RFC 3118, June 2001.

[14] Mealling, M. and R. Denenberg, "Report from the Joint W3C/IETF URI Planning Interest Group: Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations", RFC 3305, August 2002.

[15] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part One: The Comprehensive DDDS", RFC 3401, October 2002.

[16] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Two: The Algorithm", RFC 3402, October 2002.

[17] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database", RFC 3403, October 2002.

[18] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Four: The Uniform Resource Identifiers (URI)", RFC 3404, October 2002.

[19] Klensin, J., "Role of the Domain Name System (DNS)", RFC 3467, February 2003.

[20] Sun, S., Lannom, L. and B. Boesch, "Handle System Overview", RFC 3650, November 2003.

[21] IEEE, "Guidelines for use of a 48-bit Global Identifier (EUI-48)", December 2003, <http://standards.ieee.org/regauth/oui/tutorials/EUI48.html>.

[22] IEEE, "Guidelines for 64-bit Global Identifier (EUI-64) Registration Authority", December 2003, <http://standards.ieee.org/db/oui/tutorials/EUI64.html>.

[23] IEEE, "802.11 Wireless", December 2003, <http://standards.ieee.org/getieee802/802.11.html>.

[24] OCLC, "PURLS: Persistent Uniform Resource Locators", December 1995, <http://purl.oclc.org/docs/new_purl_summary.html>.

[25] Shoch, J., "Internetwork Naming, Addressing, and Routing", Proceedings of the 17th IEEE Computer Society International Conference pp. 72-79, December 1978.

[26] Kunze, J. and R. Rodgers, "The ARK Persistent Identifier Scheme", draft-kunze-ark-08 (work in progress), July 2004.

Authors' Addresses

Patrik Faltstrom (editor)
Internet Architecture Board

EMail: paf@cisco.com

Geoff Huston (editor)
Internet Architecture Board

EMail: gih@telstra.net

Appendix A. IAB Members

Internet Architecture Board Members at the time this document was drafted were:

Bernard Aboba
Harald Alvestrand
Rob Austein
Leslie Daigle
Patrik Faltstrom
Sally Floyd
Jun-ichiro Itojun Hagino
Mark Handley
Geoff Huston
Pete Resnick
Bob Hinden
Eric Rescorla
Jonathan Rosenberg

Intellectual Property Statement

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.