IPS
Internet Draft
Draft-ietf-ips-iscsi-name-disc-01.txt
Draft Title: iSCSI Naming and Discovery

Mark Bakke
Cisco

Joe Czap
IBM

Jim Hafner
IBM

Howard Hall
Pirus

Jack Harwood
EMC

John Hufferd
IBM

Yaron Klein
Sanrad

Lawrence Lamers
San Valley Systems

Todd Sperry
Adaptec

Joshua Tseng
Nishan

Kaladhar Voruganti
IBM

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026 except that the right to

produce derivative works is not granted. Internet-Drafts are working documents of the Internet Engineering.Task Force (IETF), its areas,

and its working groups. Note that other groups may also distribute working documents as Internet-Drafts. Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet- Drafts as reference material or to cite them other than as "work in progress." The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Comments
Comments should be sent to the ips mailing list (ips@ece.cmu.edu) or to kaladhar@us.ibm.com

1. Abstract

This document describes iSCSI [7] naming and discovery details. This document complements the iSCSI IETF draft. Flexibility is the key guiding principle behind this requirements document. That is, an effort has been made to satisfy the needs of both small isolated environments, as well as large environments requiring secure/scalable solutions.

This document has been organized into the following sections:
a) Section 3 presents the naming requirements. It discusses the concept of an iSCSI Name.
b) Section 4 discusses the discovery requirements.
c) Section 5 presents Storage Name Server (SNS) requirements.
d) Section 6 presents the details of iSNS protocol. iSNS meets the requirements of SNS. The protocols identified in section 6, which are used by iSNS, MUST also be supported by any iSCSI compliant SNS protocol.
e) Section 7 briefly lists the other existing discovery protocols.
f) Section 8 briefly discusses the security implications on the discovery mechanism.

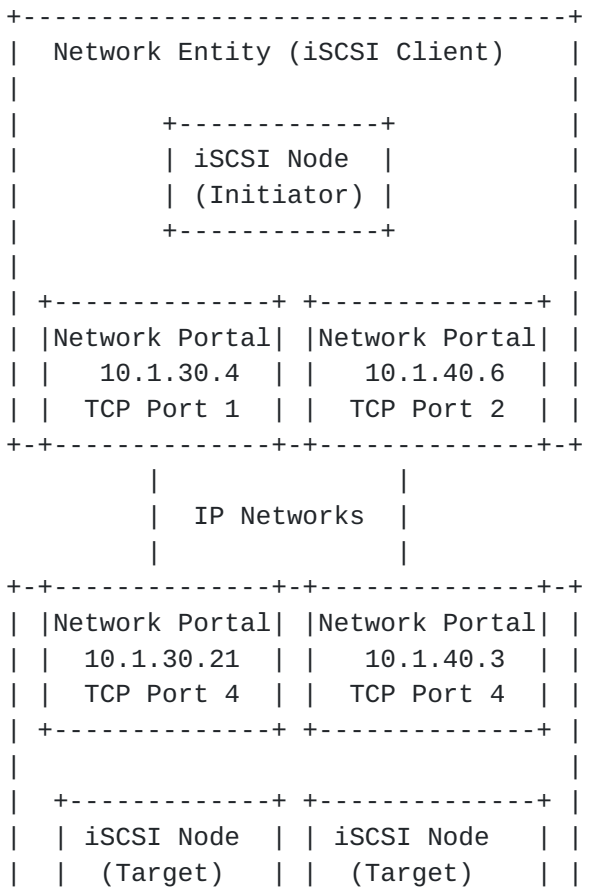
g) Appendix A describes the different hardware and software components with whom the initiator and target iSCSI Names can be associated.
h) Appendix B contains examples on how the iSCSI Names are to be used in iSCSI Login commands.
i) Appendix C contains a taxonomy of iSCSI proxy and firewall concepts. This taxonomy helps to evaluate the behavior of the discovery mechanism when dealing with proxies and firewalls.

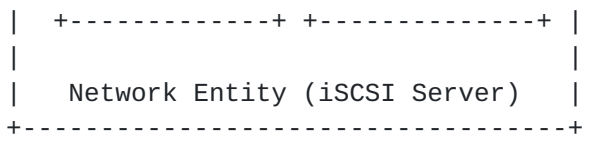
2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119.

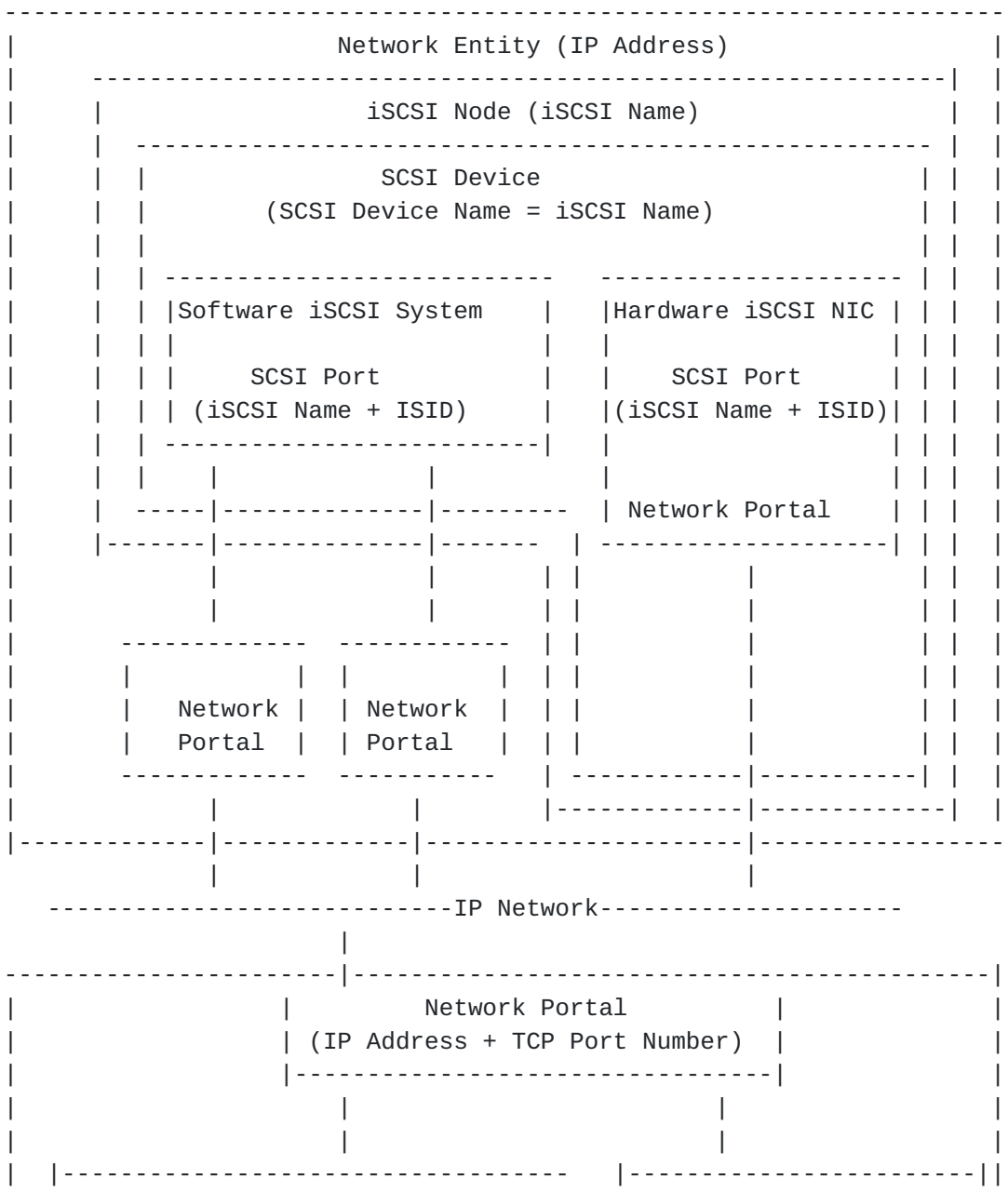
3. Naming Requirements

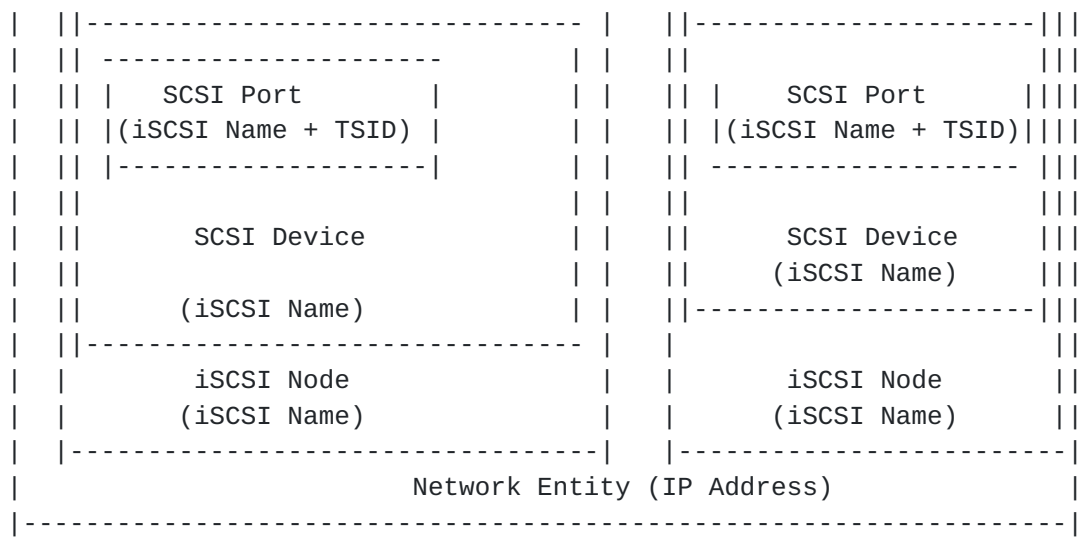
The iSCSI protocol exists to allow iSCSI initiators to connect to iSCSI targets. It is essentially a client-server model, with the initiators represented by the clients, and the targets represented by servers.





A Network Entity object can contain multiple iSCSI Node and Network Portal objects. An iSCSI initiator is represented by an iSCSI node object and an iSCSI target is also represented by an iSCSI node object. In order for an iSCSI initiator to connect to an iSCSI target, the initiator needs to provide information about the target iSCSI node object and the target network portal object. The details of these iSCSI objects and the relationship of these iSCSI objects with the SAM-2 concepts is provided below:





a) Network Entity Object
The Network Entity object represents a device or gateway that is accessible from the IP network. This device or gateway may support one or more iSCSI node objects. The iSCSI node object is accessed via a network portal object.

b) iSCSI Node Object
The iSCSI Node object defines an individual iSCSI initiator or target. In addition to iSCSI functionality, the iSCSI Node provides SCSI device functionality. There may be one or more iSCSI Storage Node objects within the Network Entity object. An iSCSI Node object is identified by its iSCSI name. There is a requirement for iSCSI names to be unique. However, it is not mandatory for the initiators and targets to use the iSCSI names because a unique identifier might not be required in some simple, isolated iSCSI configurations. iSCSI names are useful because in some cases (e.g. when DHCP services [6] are used etc), the combination of IP address and port number [6] cannot uniquely identify an initiator or a target.

There is a default iSCSI Node object present at every target network entity that can be accessed without specifying the iSCSI name. However, if there are multiple iSCSI target Nodes that are serviced by a single Network Entity and Network Portal objects, then it is necessary for the initiator to specify the target iSCSI name to uniquely identify the target iSCSI node. An alias string could also be associated with an iSCSI target node. The target alias helps an organization to associate their own semantic meaning with the target alias string. However, the target alias string is not a substitute for the target iSCSI name.

c) SCSI Device Object
This is the SAM-2 term for an entity that contains other SCSI entities. For example, a SCSI Initiator Device contains one or more SCSI Initiator Ports and zero or more application clients; a SCSI

Target Device contains one or more SCSI Target Ports and one or more logical units. For iSCSI, an iSCSI Node provides the functionality of a single SCSI Device. The SCSI Device Name is the same as the iSCSI Node name

d) SCSI Port Object
This is the SAM-2 term for an entity in a SCSI device that provides the SCSI functionality to interface with a service delivery subsystem or transport. For iSCSI, the SCSI Port Object maps to the endpoint of a session. An iSCSI session is negotiated through the login process between an iSCSI Intiator Node and an iSCSI Target Node. At successful completion of this process, a SCSI Initiator Port is created within the iSCSI Initiator Node and similarly, a SCSI Target Port is created within the iSCSI Target Node. The session is the SCSI I_T nexus. The SCSI Initiator Port Name and SCSI Initiator Port Identifier are the iSCSI Initiator Node name together with the ISID of the session identifier. Similarly, the SCSI Target Port name and SCSI Target Port Identifier are the iSCSI Target Node name together with the TSID of the session identifer. There can be only one iSCSI session with a given ISID between an iSCSI Intiator Node and an iSCSI Target Node. There can be multiple SCSI Port objects present in an iSCSI Storage Node object (one for each session created). In software iSCSI representations, the iSCSI Storage Node object creates a session process which, in turn, represents the SCSI Port. By making the SCSI Port be a separate object from the iSCSI Node object, it allows one to use different combinations of software and hardware iSCSI implementations within the same iSCSI node umbrella. Moreover, this also allows the iSCSI Node name at the initiators to be

associated with the operating system footprint and not with any network card hardware (such as the iSCSI offload network card). In hardware iSCSI offload card implementations, the session process is present in the iSCSI network card. The iSCSI Node object passes the unique iSCSI Node name and the ISID or the TSID to the session process. A target uses the initiator iSCSI Node name and the ISID combination to implement part of the persisent feature of SCSI reservations. Therefore, if the initiator wants to maintain the state from its previous session, the iSCSI node at the initiator needs to ensure that the same ISID is assigned (after an iSCSI session logout and during an iSCSI session re-login attempt) when connecting to the same target iSCSI Node object, and the target iSCSI node needs to ensure that during the Login phase, it assigns the same TSID to the login attempt with the particular ISID (if the state from the previous session between the SCSI ports is present at the target). Thus, the target iSCSI Node needs to maintain the state of the pending initiator SCSI reservations.

e) Network Portal Object
The Network Portal object is a port through which access to any iSCSI Node object within the Network Entity object can be obtained. A Network Entity object must have one or more Network Portal objects,

each of which is usable by iSCSI Node objects contained in that Network Entity object to gain access to the IP network. The Network Portal object is identified by its IP address and Port number. A SCSI Port object (an iSCSI session) can use multiple Network Portal objects and a Network Portal object can be used by multiple SCSI port objects. A hardware iSCSI offload card contains at least a SCSI Port object and a Network Portal object. Moreover, the Network Portal contained within a hardware iSCSI NIC can only be used by the SCSI Port(s) contained within the hardware NIC.

3. Naming Requirements

The concepts of names and addresses have been carefully separated In iSCSI. A name is an identifier that specifies an end-point, such as an initiator or target. The name must be location-independent; the target can be moved to another address, or have multiple addresses, but it is still the same target. An address is an identifier that specifies a path to an end-point at which the target may currently be found. Names are forever; addresses may change at any time.

This means that two types of identifiers are needed for iSCSI initiators and targets:

- An iSCSI Node Name, or iSCSI Name, is a location-independent, permanent identifier for an iSCSI initiator or target. It stays constant for the life of the target. An iSCSI Name specifies an iSCSI Node.

- An iSCSI Address specifies the location of an iSCSI initiator or target. An iSCSI address consists of a host name or IP address, a TCP port number, and the iSCSI Name of the host or target. An iSCSI Address specifies an iSCSI Portal plus

  an iSCSI Node.

- An iSCSI Identifier is either an iSCSI Name or an iSCSI Address. Some specifications of iSCSI targets, such as a configuration File or the iSCSI Boot mechanism [ref], require either a name or an address to be stored in the same field. In this case, the term "iSCSI Identifier" is specified to indicate that either the name or the address can be used.

This set of structures has already been defined by the WWW and internet folks. An iSCSI Identifier is functionally equivalent to a Uniform Resource Identifier (URI), an iSCSI Name is functionally equivalent to a Uniform Resource Name (URN), and an iSCSI Address fulfils the same function as a Uniform Resource Locator (URL).

In addition to the iSCSI Name, an iSCSI Node may have an iSCSI Alias. This alias is not a unique identifier, but is an administratively-assigned, human-readable text string that assists a user in identifying a target or initiator.

A similar analogy exists for people. A person might be:

Robert Smith
SSN: 333-44-5555
Phone: +1 (763) 555.1212
Home Address: 555 Big Road, Minneapolis, MN 55444
Work Address: 222 Freeway Blvd, St. Paul, MN 55333

In this case, Robert's globally unique name is really his Social Security Number (perhaps it should have a "us." in front of it); his common name, "Robert Smith", is not guaranteed to be unique. Robert has three locations at which he may be reached; two Physical addresses, and a phone number. In this example, Robert's SSN is like the iSCSI Name, his phone number, and addresses are analogous to the iSCSI Address, and "Robert Smith" would be the iSCSI Alias.

A default target is present on each Network Entity that contains targets. This default target has the iSCSI Name "iscsi", which is not globally unique. An iSCSI initiator, given only the IP address and TCP port of an iSCSI portal, can connect to this default target to request a list of targets within the network entity.

The default target need not be present within a network entity that contains only initiators.

3.1 iSCSI Names

An iSCSI Node Name, or iSCSI name, is used to identify each iSCSI initiator and target. The terms "initiator name" and "target name", when used in this document, refer to iSCSI Node Names. This name is designed to be globally unique, and is independent of the location of the initiator or target. iSCSI names are formatted as UTF-8 text

strings.

iSCSI names may be assigned by a hardware manufacturer, software manufacturer, or the end user. A naming authority scheme is provided to ensure that each of these can confidently generate unique names.

The iSCSI Name uniquely identifies iSCSI initiators and targets. The Initiator Name corresponds to the logical operating system on which the initiator is running, and the Target Name corresponds to the target Storage Node entity.

iSCSI names are designed to fulfill the following requirements:

1. iSCSI names are globally unique. No two initiators or targets should have the same name.

2. iSCSI names are permanent. An iSCSI initiator or target has the same name for its lifetime.

3. iSCSI names do not imply a location or address. An iSCSI initiator or target can move, or have multiple addresses. A change of address does not cause a change of name.

4. iSCSI names must not rely on a central name broker; the naming authority must be distributed.

5. iSCSI names must support integration with existing unique naming schemes.

6. iSCSI names must rely on existing naming authorities. iSCSI must not create its own naming authority.

The encoding of an iSCSI name also has some requirements:

1. iSCSI names have one single encoding method when transmitted over various protocols.

2. iSCSI names must be relatively simple to compare. The algorithm for comparing two iSCSI names for equivalence must not rely on any external server.

3. iSCSI names must be transcribable by humans. iSCSI names should be kept as simple as possible, and should not use more than a few special characters. They must also be case-insensitive, and cannot include white space.

4. iSCSI names must be transport-friendly. They must be transported using both binary and ASCII-based protocols, as well as on paper.

An iSCSI Name really names a logical software entity, and is not tied to a port or other hardware that can be changed. For instance, an Initiator Name should name the iSCSI initiator driver, and not a particular NIC or HBA card. When multiple NICs are used, they should generally all present the same iSCSI Initiator Name to the targets, since they are really to the same entity. In most operating systems, the named entity is the operating system image. Most hosts will have a single OS running; some of the really big ones could have

multiples.

A Target Name should similarly not be tied to hardware interfaces that can be changed. A Target Name should identify the logical

target,and must be the same for the target regardless of the physical porton which it is addressed. This gives iSCSI initiators an easy way to determine that two targets it has discovered are really two paths to the same target.

The iSCSI Name is designed to fulfill the functional requirements for Uniform Resource Names (URN) [RFC1737]. Among these requirements are that the name must have a global scope, independent of address or location, and that it be persistent and globally unique. It must be extensible, and scale with the use of naming authorities. The encoding of the name should be transcribable by a human, as well as be machine-readable. There are other requirements as well; please read RFC1737 (only 5 pages) for definitions of these requirements.

The iSCSI Name may be displayed by user interfaces, but is generally uninterpreted and used as an opaque, case-insensitive string for comparison with other iSCSI Name values.

An iSCSI name is text-based. This was done for the following reasons:

- A text-based identifier is transcribable, and is easier to differentiate when looking at a user interface, or while debugging problems with iSCSI login and discovery.

- iSCSI names are only used during login and discovery phases, so the overhead does not get in the way of the data path.

- The iSCSI protocol communicates these via text strings anyway, so it "fits in" easily.

An iSCSI name consists of three parts: a type designator, followed by A naming authority, with the remaining format designated by the naming authority itself, subject to the following requirements.

An iSCSI name can be any Unicode character string with the following properties:

- it is in Normalization Form C (see Unicode Standard Annex #15, "Unicode Normalization Forms" at http://www.unicode.org/unicode/reports/15)

- it contains only the ASCII dash character ('-'=U+002d) or the ASCII dot character ('.'=U+002e) or is in one of the following Unicode General Categories:

  a) Lu (Letter, Uppercase)
  b) Ll (Letter, Lowercase)
  c) Lt (Letter, Titlecase)
  d) Lm (Letter, Modifier)
  e) Lo (Letter, Other)
  f) Nd (Number, Decimal Digit)

g) Nl (Number, Letter)

h) No (Number, Other)

- when encoded in UTF-8, it is no more than 255 bytes

In particular, white space, punctuation (except as noted), marks and symbols are not allowed.

When included in Text or Login messages, an iSCSI Name SHALL be formatted in UTF-8 form.

For the purposes of comparison, computing hash values, or anything else that operates on an iSCSI Name, the name must first be converted to lower-case in a locale-independent manner (case-folding) per the rules described in Unicode Technical Report #21, "Case Mappings", section 2.3 "Caseless Matching" (see "http://www.unicode.org/unicode/reports/tr21").

The iSCSI Name does not define any new naming authorities. Instead, it supports two existing authorities: a Fully-Qualified Name, using domain names as an authority, similar to the Java class naming heirarchy, and the EUI format used in Fibre Channel world-wide names.

Since there are different types of naming authorities, there are different types of iSCSI Names to make use of them. Each name is prefixed with a short type designator string that indicates the type of naming authority being used.

Here are the type designator strings that may currently be used:

iscsi	- Not unique; indicates a "canonical" target or initiator.
fqn.	- Fully-Qualified Name
eui.	- Remainder of the string is an EUI-64 address, in ASCII hexadecimal.

As these two naming authorities will suffice in nearly every case imaginable for both software and hardware-based entities, the creation of additional type designators is discouraged. Use of type strings not listed here is not allowed, as they cannot be guaranteed to be unique.

The use of the naming authority means that iSCSI names can be assigned by virtually any uniqueness scheme that can be devised by OS vendors, driver or iSCSI NIC vendors, device vendors, gateway vendors, and even the customer.

Type "iscsi"

This type does not specify a real iSCSI Name; it is used during Login as a default or canonical name.

Example Name:

iscsi

This type does not use a naming authority, and so is not a real iSCSI Name. Every device allowing target connections will support this as a default target, so it is not globally unique. Every device supporting the "iscsi" Name should also support one or more actual iSCSI Names of one of the other two types.

Type "fqn." (Fully-Qualified Name)

This iSCSI name type can be used by either a manufacturer, end user,or service provider. This naming authority is handy especially when an end user or service provider wishes to provide the iSCSI Name for a target. These customers all own domain names; the same is not true for OUI, SCSI Vendor ID, or any of the other assigned identifiers that could be used as a naming authority.

To generate names of this type, the person or organization generating the name must own a DNS domain name. This name does not have to be active, and does not have to resolve to an address; it just needs to be reserved to prevent others from generating iSCSI names using the same domain name. For example, "ACME Storage Arrays, Inc.", might own the domain "acme.com".

The fully qualified name string consists of:

- The string "fqn.", used to distinguish these names from other types, such as "eui".

- A reversed domain name, owned by the person or organization creating the iSCSI name. For example, our storage vendor example would reverse its name to "com.acme". This is similar to the process used to generate unique Java class names [22], but without the restrictions on the domain name, since iSCSI names allow hyphens, and does not have any reserved words.

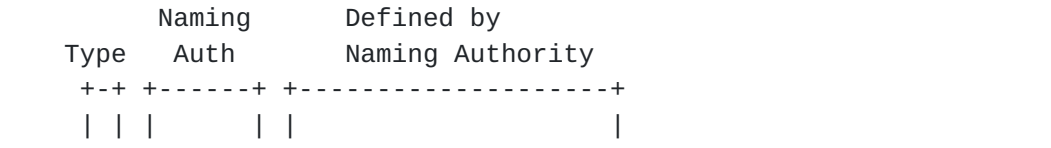
- Another ".".

- Any string, within the character set and length boundaries, that the owner of "acme.com" deems appropriate. This may contain product types, serial numbers, host identifiers, software keys, or anything else that makes sense to uniquely identify the initiator or target.

After the "fqn.", the string starts with a backwards domain name specifying the Naming Authority, using dots as separators, just as in a regular domain name. It's backwards, since it is not really used as a fully qualified host name; only the necessary top levels need by used.

Basically, everything after the backwards domain name, followed by another dot ".", can be assigned as needed by the owner of the domain name.

Here is an example of a fully-qualified iSCSI Name string from an equipment vendor:



Where:

"fqn" specifies the use of the fully-qualified name as the authority.

"com.acme" defines the naming authority. The owner of the DNS name "acme.com" has the sole right of use of this name within an iSCSI name, as well as the responsibility to keep the remainder of the iSCSI name unique. In this case, acme.com happens to manufacture disk arrays.

"diskarrays" was picked arbitrarily by acme.com to use to identify the disk arrays they manufacture. Another product that ACME makes might use a different name, and have their own namespace independent of the disk array group. Again, this is not specified here; it's just an example of what ACME could do.

"sn" was picked by the disk array group of ACME to show that what follows is a serial number. They could have just assumed that all iSCSI Names are based on serial numbers, but they thought that perhaps later products might be better identified by something else. Adding "sn" was a future-proof measure.

"a8675309" is the serial number of the disk array, uniquely identifying it from all other arrays.

A research example:

```
  Type    Auth                   Naming Authority
   +-+ +----------------------+ +-----------+
   | | |                      | |           |
   fqn.edu.pika-u.cs.users.oaks.proto.target4
```

In the above example, Professor Oaks of Pika University is building research prototypes of iSCSI targets. Pika-U's computer science department allows each user to use his or her user name as a naming authority for this type of work. Professor Oaks chose to use "proto.target4" for a particular target.

Here is an example of a fully-qualified iSCSI Name string from a service provider:

```
        Naming      Defined by
  Type   Auth       Naming Authority
   +-+ +--------+ +----------------------+

   | | |        | |                      |

   fqn.com.my-ssp.customers.4567.disks.107
```

In this case, a storage service provider (my-ssp.com) has decided to re-name the targets from the manufacturer, to provide the flexibility to move the customer's data to a different storage subsystem should the need arise.

My-ssp has configured the iSCSI Name on this particular target for one of its customers, and has determined that it made the most sense to track these targets by their Customer ID number and a disk number. This target was created for use by customer #4567, and is the 107th target configured for this customer.

Note that when reversing these domain names, that the first component(after the "fqn.") will always be a top-level domain name, which includes "com", "edu", "gov", "org", "net", "mil", or one of the two-letter country codes. The use of anything else as the first component of these names is not allowed. In particular, companies generating these names must not eliminate their "com." from the string.

Again, these iSCSI names are NOT addresses. Even though they make use of DNS domain names, they are used only to specify the naming authority. An iSCSI name contains no implications of the iSCSI target or initiator's location. The use of the domain name is only a method of re-using an already ubiquitous name space.

Note that we could have allowed the SCSI Vendor ID or IEEE OUI as a naming authority. However, some large customers and service providers may wish to use their own identification scheme, rather

than that provided by the manufacturer. These customers would not likely have a registered Vendor ID, but the domain name we used is ubiquitous, and was deemed more appropriate.

Further examples of fully-qualified iSCSI names are given at the end of this document.

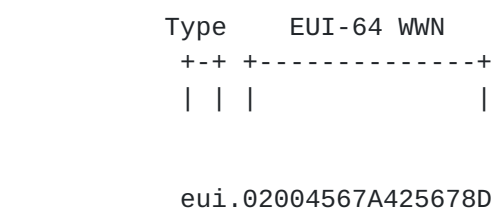
Type "eui." (IEEE EUI format)

The IEEE iSCSI name might be used when a manufacturer is already basing unique identifiers on World-Wide Names as defined in the SCSI SPC-2 specification.

It may also be used by a gateway representing a Fibre Channel or SCSI device that is already adequately identified using a world-wide name.

The format is "eui." followed by 16 hex digits.

Example iSCSI name:



Initiator and Target Requirements for iSCSI Name support:

Each initiator and target implementation must support the use of iSCSI names.

The initiator MUST send an InitiatorName and a TargetName as text fields within the login request. If the initiator does not have or support an iSCSI name, it must send an InitiatorName of "iscsi". If the initiator is logging in to the canonical (default) target, it must specify a TargetName of "iscsi". Note that if an InitiatorName of "iscsi" is used, the initiator stands the risk that it will be excluded from accessing some or all of its targets.

An initiator MAY send an InitiatorAlias as a text field within its login request. The target may use this as an informational field only; it must not be used for unique identification or authentication purposes.

The target MUST send a TargetName as a text field within its login response. Unless the initiator specified the TargetName "iscsi" in the request, this TargetName MUST match that specified by the initiator. If the initiator had specified a TargetName of "iscsi",

this TargetName should be the actual iSCSI Name of the target, or can be returned as "iscsi" if either the target is just a canonical target used for the SendTargets command, or if the target does not have an iSCSI Name.

The target MAY send a TargetAlias as a text field within its login response. The initiator may use this as an informational field only; it must not be used for unique identification or authentication purposes.

Initiators and targets shall support the receipt of iSCSI names of up to the maximum length. If configuration of the initiator or target name is allowed, the implementation shall support the maximum length.

In their user interfaces, both shall support, at a minimum, the display of the ASCII characters within the iSCSI Name's UTF-8 string.

If the other characters are unsupported, they may be displayed with escape codes as specified in [RFC 2396].

3.4 iSCSI Alias

The iSCSI alias is a UTF-8 text string that may be used as an additional descriptive name for an initiator and target. This may not be used to identify a target or initiator during login, and does not have to follow the uniqueness or other requirments of the iSCSI name. The alias strings are communicated between the

initiator and target at login, and can be displayed by a user interface on either end, helping the user tell at a glance whether the initiators and/or targets at the other end appear to be correct. The alias must NOT be used to identify, address, or authenticate initiators and targets.

The alias is a variable length string, between 0 and 255 characters, and is terminated with at least one NULL (0x00) character. No other structure is imposed upon this string.

3.2.1 Purpose of an Alias

Initiators and targets are uniquely identified by an iSCSI Name. These identifiers may be assigned by
a hardware or software manufacturer, a service provider, or even the customer. Although these identifiers are nominally human-readable, they are likely be be assigned from a point of view different from that of the other side of the connection. For instance, a target name for a disk array may be built from the array's serial number, and some sort of internal target ID.

Although this would still be human-readable and transcribable, it offers little assurance to someone at a user interface who would like to see "at-a-glance" whether this target is really the correct one.

The use of an alias helps solve that problem. An alias is simply a descriptive name that can be assigned to an initiator or target, that is independent of the name, and does not have to be unique. Since it is not unique, the alias must be used in a purely informational way. It may not be used to specify a target at login, or used during authentication. It is not used in place of the old iscsi "path" concept; the iSCSI Name is used there instead.

Both targets and initiators may have aliases.

3.2.2 Target Alias

To show the utility of an alias, here is an example using an alias for an iSCSI target.

Imagine sitting at a desktop station that is using some iSCSI devices over a network. The user requires another iSCSI disk, and calls the storage services person (internal or external), giving any authentication information that the storage device will require for the host. The services person allocates a new target for the host, and sends the Target Name for the new target, and probably an address, back to the user. The user then adds this Target Name to the configuration file on the host, and discovers the new device.

Without an alias, a user managing an iSCSI host would click on some sort of "show targets" button to show the targets to which the host is currently connected.

```
+--Connected-To-These-Targets----------------------
|

|  Target Name
|
|  fqn.com.acme.sn.5551212.target.450
|  fqn.com.acme.sn.5551212.target.489
|  fqn.com.acme.sn.8675309
|
+-------------------------------------------------
```

In the above example, the user sees a collection of iSCSI Names, but with no real description of what they are for. They will, of course, map to a system-dependent device file or drive letter, but it's not easy looking at numbers quickly to see if everything is there.

If a more intelligent target configures an alias for each target, perhaps at the time the target was allocated to the host, a more descriptive name can be given. This alias is sent back to the initiator as part of the login or sendTargets responses, for use in a display such as this. The new display might look like:

```
+--Connected-To-These-Targets----------------------
|
|  Alias          Target Name
|
|  Oracle 1       fqn.com.acme.sn.5551212.target.450
|  Local Disk     fqn.com.acme.sn.5551212.target.489
|  Exchange 2     fqn.com.acme.sn.8675309
|
+-------------------------------------------------
```

This would give the user a better idea of what's really there.

In general, flexible, configured aliases will probably be supported by larger storage subsystems and configurable gateways. Simpler devices will likely not keep configuration data around for things such as an alias. The TargetAlias string could be either left unsupported (not given to the initiator during login) or could be returned as whatever the "next best thing" that the target has that might better describe it. Since it does not have to be unique, it could even return SCSI inquiry string data.

Note that if a simple initiator does not wish to keep or display alias information, it can be simply ignored in the login or sendTargets responses.

3.2.3 Initiator Alias

An initiator alias can be used in the same manner as a target alias. An initiator would send the alias in a login request, when it sends its iSCSI Initiator Name. The alias is not used for authentication, but may be kept with the session information for display through a management GUI or command-line interface (for a more complex subsystem or gateway), or through the iSCSI MIB.

Note that a simple target can just ignore the Initiator Alias if it has no management interface on which to display it.

Usually just the hostname would be sufficient for an initiator alias, but a custom alias could be configured for the sake of the service provider if needed. Even better would be a description of what the machine was used for, such as "Exchange Server 1", or "User Web Server".

Here's an example showing a list of sessions on a target device.

For this display, the targets are using an internal target number, which is a fictional field that has purely internal significance.

```
+--Connected-To-These-Initiators-------------------
|
|  Target   Initiator Name
|
|  450      fqn.com.sw.cd.12345678-OEM-456
|  451      fqn.com.os.hostid.A598B45C
|  309      fqn.com.sw.cd.87654321-OEM-259
|
+-------------------------------------------------
```

And with the initiator alias displayed:

```
+--Connected-To-These-Initiators-------------------
|
|  Target   Alias               Initiator Name
|
|  450      Web Server 4        fqn.com.sw.cd.12345678-OEM-456
|  451      scsigate.yours.com  fqn.com.os.hostid.A598B45C
|  309      Exchange Server     fqn.com.sw.cd.87654321-OEM-259
|
+-------------------------------------------------
```

This gives the storage administrator a better idea of who is connected to their targets. Of course, one could always do a reverse DNS lookup of the incoming IP address to determine a host name, but simpler devices really don't do well with that particular feature due to blocking problems, and it won't always work if there is a firewall or iSCSI gateway involved.

Again, these are purely informational and optional.

Aliases are extremely easy to implement. Targets just send a TargetAlias whenever they send a TargetName. Initiators just send an InitiatorAlias whenever they send an InitiatorName. If an alias is received that does not fit, or seems invalid in any way, it is ignored.

4. iSCSI Discovery

The goal of iSCSI discovery is to allow an initiator to find the targets to which it has access, and at least

one address at which each target may be accessed. This should generally be done using as little configuration as possible. This section defines the discovery mechanism only; no attempt is made to specify central management of iSCSI devices within this document.

There are several methods that may be used to find targets and their addresses, ranging from configuring a list of targets and addresses on each initiator and doing no discovery at all, to configuring nothing on each initiator, and allowing the initiator to discover targets via multicast mechanisms.

An iSCSI initiator can discover iSCSI targets in these ways:

a. iSCSI targets are configured on the initiator.
b. The initiator queries iSCSI servers using the SendTargets command.
c. The initiator queries a storage name server, such as iSNS, for targets.
d. The initiator uses the Service Location Protocol (SLP) to find iSCSI targets, iSCSI servers, and storage name servers.

4.1 Configuring Target Information

The exact manner in which the target information is hard-coded at the initiator is an implementation detail. The information could be present in some persistent location (such as a file) that can be accessed by the initiator.

Target discovery can be configured on an initiator in several ways:

- Full Target URL. This includes the target's IP address or host name, TCP port, and iSCSI Name. No further discovery is required to contact this target.

- Target Name. This includes only the target's name, and contains no address information. The initiator must query SLP or a name server to locate this target.

- Canonical Target Name. This is just an iSCSI server's IP address and TCP port, the canonical name "iscsi". The initiator must connect to this address, log in to the canonical target, and issue a SendTargets command to acquire the list of targets it can use.

- Storage Name Server Address. This is an address of a storage name server, such as iSNS, that the initiator may query to find more targets. The information required to configure an initiator for a storage name server is outside the scope of this document.

4.2 SendTargets Command

An initiator may connect to an iSCSI address (IP address + TCP port)and log in to the canonical target name "iscsi". The login process for this target is identical to that of any other target. If

there are no targets available that would provide access to the initiator's Name, the target SHOULD reject the initiator's login to the canonical target with the status code set to 0x202 "forbidden target". Upon successful login to the "iscsi" target, the initiator may send the text command "SendTargets", to retrieve a list of target names to which it may attempt login. The canonical target MUST support this command, and MUST return a list of zero or more target Names. Each iSCSI Name returned may include zero or more TargetAddress fields, as well one optional TargetAlias field. If zero Names are returned, the canonical target is unaware of any targets that are accessible by the initiator. The command is sent by formatting an iSCSI Text Command, with the Final (F) bit set to 1. The first key in the command's text must be

SendTargets=

No value is sent for the send-targets key, and no other keys are sent.

The response to this command is a text response containing a list of text keys. The text keys are organized as a set of records, one for each iSCSI target. Each record includes the target name, a list of zero or more target addresses, and the optional target alias. The order of the text keys is very important. The appearance of the TargetName key signals the start of a new record. Subsequent TargetAddress and TargetAlias keys apply only to that record, until the next TargetName key is found (or the command response is done).

Each target starts with one text key of the form:

TargetName=<target-name-goes-here>

It may then include zero or more address keys:

TargetAddress=<hostname-or-ipaddress>[:<tcp-port>]

It may then include the optional target alias key:

TargetAlias=<alias-string-goes-here>

This example is the SendTargets response from a single target, that has no other interface ports, and does not support an alias:

TargetName=fqn.com.acme.diskarray.sn.8675309

Note that all it really had to return in the simple case was the iSCSI name. It is assumed by the initiator that the IP address and

TCP port for this iSCSI Name are the same as used on the current connection to the canonical iSCSI target.

The next example has two internal iSCSI targets, each support via two different ports with different IP addresses:

```
TargetName=fqn.com.acme.diskarray.sn.8675309
TargetAddress=10.1.0.45:3000

TargetAddress=10.1.1.45:3000
TargetAlias=Oracle disk four
TargetName=fqn.com.acme.diskarray.sn.1234567
TargetAddress=10.1.0.45:3000
TargetAddress=10.1.1.45:3000
TargetAlias:Oracle disk five
```

Note that both targets share both addresses; the multiple addresses are likely used to provide multi-path support. The initiator may connect to either targetName on either address.
Also note that in the above example, a DNS host name could have been returned instead of an IP address, and that an IPv6 addresses (5 to 16 dotted-decimal numbers) could have been returned as well.

After obtaining a list of targets in this manner, an iSCSI initiator may create new sessions to log in to the discovered targets. The initiator MAY keep the session to the canonical target open, and MAY send subsequent SendTargets commands to discover new targets. The target MUST send any iSCSI-level async event notifications on this session, to allow the initiator to discover new targets as they are created. [Note: The Asynch Message codes in the iSCSI document need to be updated to reflect support for this feature].

Note that since SendTargets is a text response, vendor-specific keys may be introduced in the response as specified in the iSCSI specification. A vendor-specific key appearing after a TargetName key should generally be treated as part of that target's record; however, this is, of course, defined by the manufacturer introducing the key. Initiators receiving unknown or unsupported vendor-specific keys in a SendTargets response MUST silently ignore them.

4.2.1 Redirect Responses

If a target has moved, or if the iSCSI device logged in to has knowledge of another address at which a target should be accessed, it MAY return a redirect response by setting the iSCSI login status to one of the "redirect"-class status codes, and returning at least one text key with a new target address on which to find the target. This status terminates the session.

The initiator, upon receiving a redirect response, SHOULD either abandon

attempts to log in to the intended target, or attempt to re-login to the target using one of the addresses provided.

A target might do this for load balancing or it might do this to provide multiple virtual targets through a simple initiator discovery protocol.

The target's response includes the Nameof the target, plus one or more TargetAddress fields, as specified in the SendTargets response.

Here's a simple example:

```
T->Login Response(status=redirect)
TargetName=fqn.com.acme.diskarray.sn.999999
TargetAddress=10.1.0.49:3000
```

In the above example, a new address exists for the target name at 10.1.0.49, TCP port 3000. If the TCP port was not specified, it would use the default port (to be assigned by IANA).

Another example would include multiple addresses for a target, Perhaps through multiple ports on a storage controller, or through multiple gateways:

```
T->Login Response(status=redirect)
TargetName=fqn.com.acme.diskarray.sn.999999
TargetAddress=10.2.30.100
TargetAddress=10.2.40.100:2301
TargetAddress=mystorage.mycompany.com
```

Note that the address may be either an IP address or DNS host name. The first and third addresses to not include a TCP port; these would use the default, IANA-assigned TCP port.

In any case, the TargetName returned is identical to that requested by the initiator in the initial Login Request. The redirect status is not used to change names; it is only used to move a iSCSI Name from one IP address and/or TCP port to another.

4.3 Initiator queries a Storage Name Server (SNS)

Discovery and management of iSCSI devices can be extended by the use of Storage Name Servers (SNS). The term "SNS" used in this document should not be confused with the specific implementation used in Fibre Channel; it is meant to be a generic term.

An SNS can add capabilities beyond discovery of iSCSI targets, but

for the purposes of this section it must at least provide a method of discovering:

1. The addresses at which a particular iSCSI Name may be found
2. A list of iSCSI Names and/or addresses to which the initiator has access.

To make use of an SNS, an initiator must support a protocol that provides SNS query facilities.

4.4 Initiator Uses the Service Location Protocol
A storage name server address may be either configured, or discovered through SLP.

An initiator may use the Service Location Protocol, Version 2 (SLPv2) to locate iSCSI targets, canonical targets, and storage name servers, without having to configure their addresses. SLP Version 1 is not supported by iSCSI.

The Service Location Protocol (SLP) is a standard protocol for locating the addresses of resources on a network. iSCSI targets,

canonical targets, and storage name servers may advertise themselves to iSCSI initiators using SLP.

Three types of nodes participate in SLP discovery. A User Agent (UA) is the entity that wishes to discover resources. In this case, the UA is part of the iSCSI initiator. A Service Agent (SA) is the entity that wishes to be discovered. In our case, the SA is part of the iSCSI target, canonical target, or storage name server. A third entity, the Directory Agent (DA) is an optional part of discovery. If a DA is present, it collects information about the Service Agents, and is queried by the User Agents, to reduce the network load of all Uas trying to discovery all SAs.

For true zero-configuration, SLP makes use of multicast to locate DAs or SAs. However, SLP is designed to use as little multicast traffic as possible, and by using a DA, and configuring its address on each initiator, will not require multicast at all.

The SLP Protocol is described in detail in [RFC2608].

A target can register either its canonical target address, its targets themselves, or both with SLP. A storage name server can register its address with SLP, or can also register its targets with SLP, if desired.

Initiators can send the following service requests using SLP:

1. Locate all canonical targets ("iscsi")
2. Locate specific targets to which the initiator might have access
3. Locate a specific target by its iSCSI Name

4. Locate storage name servers

In addition, a storage name server can act as an initiator and make use of SLP to discover targets and canonical targets for its own use.

If a specific target is found, the initiator may simply attempt to log in to that target. An initiator supporting a storage name service may additionally query the SNS for more information on the target before logging in. Note that the same target may exist at more than one address; it is the responsibility of the initiator to ensure that the targets' names are compared, and that either only one address is used, or that some form of multi-path software is in place.

If a canonical target is found, the initiator may log in to the canonical target, and issue a SendTargets command as described in the previous section.

If a storage name server is found, and the initiator supports the use of this type of storage name server, the initiator may query the SNS as described by its particular protocol specification.

In general, if an initiator supports an SNS, it should normally not attempt to discover targets and canonical targets via SLP; it should first attempt to discover the SNS itself, and query the SNS for this information.

The choice of static configuration, SNS discovery or target storage discovery protocols is a configuration choice of the initiator.

In summary, this discovery approach is flexible in that the initiators have the freedom to select static configuration, a multicast based discovery mechanism for small, isolated iSCSI environments, or they can choose a scalable storage name server based discovery mechanism for large iSCSI environments.

Additionally, targets and initiators may be configured to participate or not participate in an SLP Scope, which allows the SLP discovery environment to be contained within a smaller group.

The Service Location Protocol uses templates, registered with IANA, to define the addresses and attributes that are communicated via SLP. The SLP templates implementation details are provided in [21] draft-bakke-iscsi-SLP-template.00, but a brief summary is as follows:

Service:iscsi - A top-level abstract template, which is just a name under which to place our other templates.

service:iscsi:target - A concrete target template, which defines the addresses and attributes for iSCSI targets and canonical targets.

service:iscsi:name-service - A concrete target template, which defines the addresses and attributes for storage name services.

5. Storage Name Server (SNS)

The following section describes requirements for any Storage Name Server used to support iSCSI. An example of a Storage Name Server is the iSNS described in the draft document draft-ietf-ips-iSNS-00.txt [8]. There potentially could be other protocols which also satisfy SNS requirements.

5.1 Overview

A SNS shall be architected using a client-server paradigm, with a SNS server predominantly serving a passive role. SNS clients actively register and manipulate entity objects and their attributes in the SNS server. A SNS server MAY send asynchronous state change notifications to registered SNS clients in response to an action by a SNS client. Examples of SNS clients include initiators, targets, management stations, and switches. A SNS server can be hosted on a target, switch, or stand-alone server.

5.2 Login Control and Discovery Domains

Discovery Domains (DD's) are logical groupings of iSCSI devices that are allowed to "see" each other. SNS MUST support Discovery Domains and Login control. SNS must provide SNS clients with the ability to Enforce Discovery Domain configurations which may exist on a SNS server. Targets and management stations shall be able to register (i.e., upload) Login Control and Discovery Domain configurations to

the SNS if authorized by the end user. Discovery Domains and Login control supports two separate purposes:

5.2.1 Discovery Domain Partitions

A SNS SHALL support the ability to partition the storage network into Separate "Discovery Domains". A SNS shall not provide information if the SNS client performing the query is not in a common Discovery Domain (DD) as the SNS client that is the subject of the request. This capability prevents an initiator from attempting an iSCSI login to every single target in a large enterprise network, and is the iSCSI equivalent of "Soft" zoning.

5.2.2 Login Control

Login access security which is specified in the iSCSI Draft (Appendix A) [7] and may be implemented by the iSCSI target. A SNS shall support login control by storing a mapping of initiators

that are permitted to access each target. Targets shall be able to query the SNS for a list of initiators that are allowed login access. This list shall include the key attribute (e.g.,iSCSI Name) used to identify the initiator. This capability is the iSCSI equivalent of "Hard" zoning.

5.3 Object Model

A SNS MUST store the following objects and attributes:

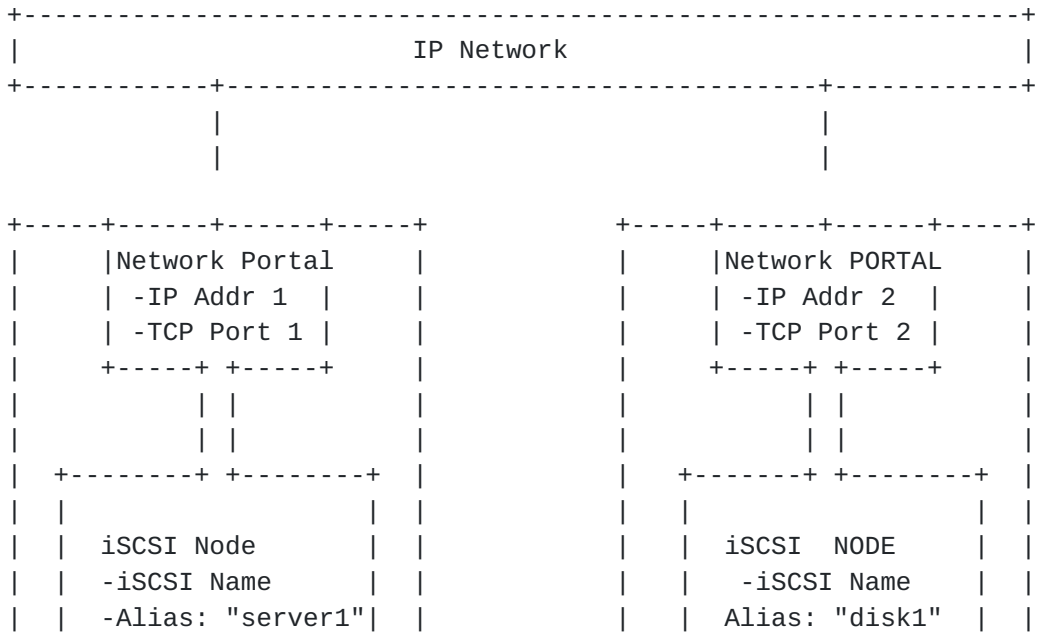
Network Entity:
- Entity Identifier
- Management IP Address
- Entity Type (iSCSI)

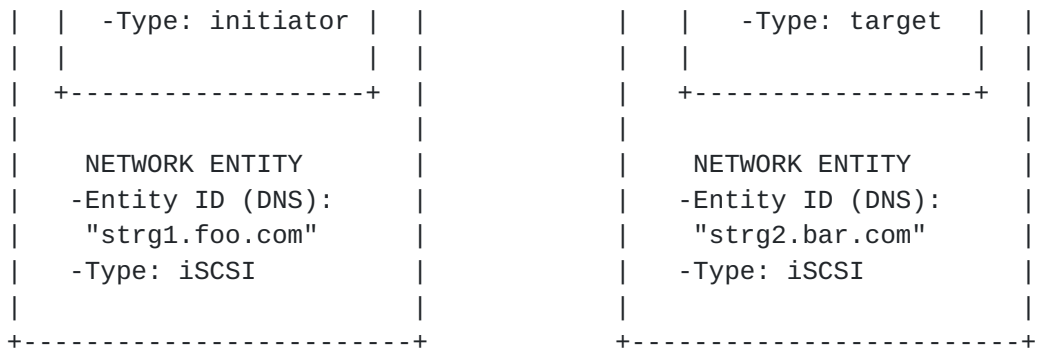
Network Portal:
- IP Address
- TCP Port Number

iSCSI Node:
- iSCSI Name
- Alias
- Node Type (target or initiator or both)

Discovery Domain:
- DD symbolic name
- DD ID
- DD Member: iSCSI Name
- DD Member: IP Address

A diagram of how the above objects are related is shown below.





A DISCOVERY DOMAIN contains one or more NETWORK ENTITY, NETWORK PORTAL, and/or iSCSI NODE, objects. Each NETWORK ENTITY object contains one or more NETWORK PORTAL objects, and one or more STORAGE NODE objects.

5.4 SNS Message Format Requirements

The SNS protocol SHALL be TLV based.
TLV (TLV is already used in many networking protocols such as DHCP).
The SNS protocol shall allow manipulation of multiple objects and attributes in a SNS server through a single message and response.

5.5 SNS Authentication Requirements

The SNS protocol SHALL include optional authentication of SNS protocol messages from SNS clients. The authentication mechanism will allow for authentication of both client and server.

5.6 SNS Query and Registration Services Requirements

The SNS protocol allows initiators and targets to register themselves at The SNS server. Initiators and targets can also query a SNS server for information. For example, targets can register themselves at a SNS server, and the initiators can query a SNS server about which targets they can access.

During registration, the initiators and the targets must provide the following information:

a) Network Portal object address (IP address and Port Number)
b) iSCSI Name information
c) Storage node type

They could optionally also provide other information such as:

a) iSCSI Node name
b) Alias string information
c) Registration for State Change Notification

If the iSCSI Node name is not provided in the initial registration, then a SNS shall create a unique Entity name for that client, and the client shall use that Entity name for all subsequent queries and updates.

When querying address information in order to establish an iSCSI connection, the query, as a minimum, should return the following information:

a) iSCSI Node IP address

The Network Portal Object IP address can be the same as the iSCSI Node IP address, and the Network Portal port number can be the (TBD) default iSCSI port number. Furthermore, the iSCSI Name of the target device can be queried by issuing the SendTarget command to the default canonical iSCSI target present at the IP address and port number.

5.7 State Change Notification Requirements

Asynchronous notification (State Change Notifications): A SNS must be able to inform SNS clients of changes to its database, including changes or modifications to Discovery Domain or login control policies and the presence or absence of initiators and targets. These changes may occur as a result of various events, including an SNS client (e.g., a management workstation) actively changing the SNS database, response or non-response to an SNS status inquiry message, or a hardware interrupt delivered by a SNS host platform (such as a switch). Asynchronous notification shall be delivered only to SNS clients that register for the notification, and only for SNS clients that are in the same Discovery Domain as the event.

5.8 The SNS protocol SHALL be a lightweight protocol that can be scaled down for implementation on switches and targets, or scaled up for implementation on servers.

5.9 The SNS protocol SHALL meet the iSCSI boot requirements (see draft-ietf-ips-iscsi-boot-00.txt).

6. iSNS - Internet Storage Name Service

iSNS is a name service protocol which can be used for discovery and management of iSCSI devices. The iSNS protocol is described in the document draft-ietf-ips-iSNS-01.txt, and meets the requirements of section 5 of this document. The following section describe how iSNS is used to support iSCSI devices.

6.1 iSCSI Requirements for iSNS

iSNS MAY be used to fulfill iSCSI Naming and Discovery Requirements. Section 5.1 of the iSNS document lists specific implementation and usage requirements for iSCSI. Sections 5.2 and 5.3 are applicable to non-iSCSI protocols, and do NOT have to be implemented to support iSCSI. The remaining sections of the iSNS document provide important background and protocol format information which are generally applicable to an iSNS implementation that supports iSCSI. One exception is the RqstDmnID and RlsDmnID commands, which are used to support Fibre Channel and iFCP fabrics.

6.2 Summary of iSNS Features & Capabilities

The following are a summary of iSNS capabilities used to support iSCSI:

6.2.1 iSNS Registration Service

iSNS allows iSCSI devices to register their identity and attributes in the iSNS database. Multiple attributes can be registered in a single message. This allows management stations to directly manage large numbers of iSCSI devices by accessing the iSNS as a single, consolidated information repository.

6.2.2 Discovery Domains (DD's)

iSNS organizes iSCSI devices into logical groups. This accomplishes two primary purposes: 1) it limits the targets visible to each initiator to the more relevant and appropriate subset of devices in the entire storage network universe; 2) it eases administration by partitioning storage devices into smaller, more manageable groups.

6.2.3 iSCSI Device Query Service

iSNS responds to queries from iSCSI devices requesting information about other iSCSI devices residing in a common Discovery Domain. Multiple attributes can be queried for in a single message.

6.2.4 State Change Notification (SCN's)

A network event, such as removal of another device from a common Discovery Domain, will cause the iSNS to send an asynchronous notification message of the event to iSCSI devices that have registered for such a notification.

6.2.5 Distribution and Retrieval of Public Key Certificates

iSNS provides a convenient mechanism to distribute X.509 Public Key certificates. These certificates can be used to set up TLS or IPSec security associations for authenticating and/or encrypting storage traffic, as well as for the Public Key authentication method in the iSCSI login process. iSCSI devices can upload their own Public Key Certificates, allowing other iSCSI devices in their Discovery Domain to retrieve them.

6.2.6 Entity Status Inquiry (ESI)

iSNS provides a polling service to detect the removal or loss of

connectivity to iSNS clients. iSCSI devices that register for ESI will receive an inquiry message from the iSNS server at regular time intervals. If the iSCSI device does not respond to three consecutive ESI messages, the iSNS server will determine that the iSCSI device is no longer available. Appropriate SCN messages will be sent to affected devices in the Discovery Domain.

6.2.7 Event Logging

iSNS provides an SCN Event Bitmap attribute for each iSCSI device allowing a management client to learn the last State Change Notification event to occur to that device. The Timestamp attribute records the precise time of the latest SCN event.

6.2.8 Name Service Heartbeat

iSNS provides a regular local subnet broadcast that allows iSCSI devices in the local network to passively listen for and learn the IP address of the iSNS server.

6.2.9 Network Time Service

iSNS provides an optional network time service allowing iSCSI devices to synchronize their time to the clock used by the iSNS.

6.3 iSCSI Attributes Supported by iSNS

The following attributes are supported by the iSNS protocol. Attributes indicated in the "REQUIRED TO IMPLEMENT" column MUST be supported by a server compliant with the iSNS protocol. Attributes indicated in the "REQUIRED TO USE" column MUST have values stored for an iSCSI device registered in the iSNS server.

Object	Attribute	REQUIRED to Implement	REQUIRED to Use
NETWORK ENTITY	Entity Identifier	*	*
	Entity Type	*	*
	Management IP Address		
	ESI Interval	*	
	Timestamp	*	
	Entity Certificate	*	
	SCN Event Bitmap	*	
	ESI TCP/UDP Port	*	*
Network POR	IP Address	*	*
	TCP/UDP Port	*	*
	Portal Symbolic Name	*	
iSCSI NODE	iSCSI Name	*	*
	Node Type	*	*
	Alias/Symbolic Node Name	*	
	Node Certificate	*	

DISCOVERY DOMAIN	DD_ID	*	*
	DD_Symbolic Name	*	
	DD Member (Entity ID)	*	
	DD_Member (iSCSI Name	*	*
	DD_Member (IP Address)	*	

6.4 iSNS Message Summary

The following messages are used by iSNS to support iSCSI devices. Messages listed in the "REQUIRED TO IMPLEMENT" column MUST be supported in the iSNS server. Messages listed in the "REQUIRED TO USE" column MUST be supported in the iSCSI device using iSNS.

			REQUIRED TO:	
Message Description	Abbreviation	Func_ID	Implement	Use
Register Dev Attr Req	RegDevAttr	0x0001	*	*
Dev Attr Query Request	DevAttrQry	0x0002	*	*
Dev Get Next Request	DevGetNext	0x0003	*	
Deregister Dev Request	DeregDev	0x0004	*	*
SCN Register Request	SCNReg	0x0005	*	
SCN Deregister Request	SCNDereg	0x0006	*	
SCN Event	SCNEvent	0x0007	*	
State Change Notification	SCN	0x0008	*	
Register DD	RegDD	0x0009	*	*
Deregister DD	DeregDD	0x000A	*	*
Register Dev in DD	RegDevDD	0x000B	*	*
Deregister Dev in DD	DeregDevDD	0x000C	*	*
Entity Status Inquiry	ESI	0x000D	*	
Name Service Heartbeat	Heartbeat	0x000E		
NOT USED		0x000F		
Request Network Time	RqstTime	0x0010		
NOT USED		0x0011-0x0012		
RESERVED		0x0013-0x8000		

The following are iSNSP response messages used in support of iSCSI:

			REQUIRED TO:	
Response Message Desc	Abbreviation	Func_ID	Implement	Use
Register Dev Attr Rsp	RegDevRsp	0x8001	*	*
Dev Attr Query Resp	DevAttrQryRsp	0x8002	*	*
Dev Get Next Resp	DevGetNextRsp	0x8003	*	
Deregister Dev Resp	DeregDevRsp	0x8004	*	*
SCN Register Resp	SCNRegRsp	0x8005	*	
SCN Deregister Resp	SCNDeregRsp	0x8006	*	
SCN Event Resp	SCNEventRsp	0x8007	*	
SCN Response	SCNRsp	0x8008	*	
Register DD Resp	RegDDRsp	0x8009	*	*

Deregister DD Resp	DeregDDRsp	0x800A	*	*
Register Dev in DD Resp	RegDevDDRsp	0x800B	*	*
Deregister Dev in DD Resp	DeregDevDDRsp	0x800C	*	*
Entity Stat Inquiry Resp	ESIRsp	0x800D	*	
NOT USED		0x800E-0x800F		
Request Net Time Resp	RqstTimeRsp	0x8010		
NOT USED		0x8011-0x8012		
RESERVED		0x8013-0xFFFF		

7) Related Work
Jini, UPnP, Salutation, and HaVi specifications also provide discovery protocol services. However, iSCSI uses SLP broadcast

discovery mechanism because SLP is being developed as an IETF standard and since it provides all of the key broadcast discovery functionality provided by the other discovery protocols [1].

8) Security
   The iSCSI initiators and targets must have a secure way of interacting with each other. Hence, once a target or name server is discovered, authentication and authorization are handled by either the iSCSI protocol, or by the name server's protocol. It is the responsibility of the providers of these services to ensure that an inappropriately advertised or discovered service does not compromise their security.

8. Appendix A: iSCSI Name Notes
 Some iSCSI Name Examples for Targets

- Assign to a target based on controller serial number

  fqn.com.acme.diskarray.sn.8675309

  See the ASCII iSCSI Name example above for discussion.

- Assign to a target based on serial number and logical target alias

  fqn.com.acme.diskarray.sn.8675309.oracle_database_1

  Where oracle_database_1 might be a target alias assigned by a user.

  This would be useful for a controller that can present
  different logical targets to different hosts.

Obviously, any naming authority may come up with its own scheme and hierarchy for these names, and be just as valid.

A target iSCSI Name should NEVER be assigned based on interface hardware, or other hardware that can be swapped and moved to other devices.

Some iSCSI Name Examples for Initiators

- Assign to the OS image by fully qualified host name

    fqn.com.osvendor.dns.com.customer1.host_four

    Note the use of two FQDNs - that of the naming authority and also that of the host that is being named. This can cause problems, due to limitations imposed on the size of the iSCSI Name.

- Assign to the OS image by OS install serial number

    fqn.com.osvendor.newos5.12345-OEM-0067890-23456

    Note that this breaks if an install CD is used more

    than once. Depending on the O/S vendor's philosophy, this might be a feature.

- Assign to the OS image by a service provider

    fqn.com.mydisk.users.mbakke05657

    Note that this could also be assigned to a particular iSCSI address if more than one service provider is used.

Using Initiator and Target iSCSI Name During Login

Some examples.

1. Login to a known target iSCSI Name; initiator supports iSCSI Names.

```
   I->Login Request
      InitiatorName= fqn.com.os.hostid.34567890
      InitiatorAlias= myhost
      TargetName= fqn.com.acme.diskarray.sn.8675309
         .
         .  text/login commands flow here during authentication phase
         .
   T->Login Response
      TargetName= fqn.com.acme.diskarray.sn.8675309
```

```
      TargetAlias= foo
```

2. Login to an unknown target Name; initiator supports iSCSI Names.

   This only works if there is a single iSCSI Name at the IP address and TCP port to which the initiator has connected.

```
   I->Login Request
      InitiatorName= fqn.com.os.hostid.34567890
      InitiatorAlias= myhost
      TargetName= iscsi
         .
         .  text/login commands flow here during authentication phase
         .
   T->Login Response
      TargetName= fqn.com.acme.diskarray.sn.8675309
      TargetAlias= 8675309
```

3. Login to a canonical target, for the SendTargets command.

```
   I->Login Request
      InitiatorName= fqn.com.os.hostid.34567890
      InitiatorAlias= myhost
      TargetName= iscsi
         .
         .  text/login commands flow here during authentication phase
         .
   T->Login Response
      TargetName= iscsi
```

   Since the target returned a iSCSI Name of "iscsi", the initiator will now use the SendTargets text command to find out which target names are actually supported at this address. It will then create new connections for each target, and do the login scenario shown in example 1.

Answers to Potentially Frequently Asked Questions

 What happens if an Initiator Name is not unique?

  - Targets will authenticate both as same entity
  - Targets will believe that one initiator is using them via different network interfaces.
  - Initiators may end up sharing a device by accident.

Appendix B: iSCSI Login Scenarios

B.1. Introduction

The Initiator Name MUST always be sent during login. As a target may use the Initiator Name as part of its access control mechanism, an initiator that does not send its Name stands the risk that it will be excluded from accessing some or all of its targets.

The target Name MUST be sent in the login phase (with the exception that the key-word iscsi can replace unknown target). This can enable the distinction between several (virtual of physical) storage entities in the device.

The iSCSI Names MUST be sent in the Login Request message, establishing the login session (together with the other login parameters). The iSCSI Names MUST be in text command format - UTF-8 coded as described in section 3.

The target MUST response to the login request with the appropriate status. The status codes are defined in the iSCSI draft [7].

B.2. Request Format

The requests and responses are in key=value format. When more than one Value is required, a comma separator is used, i.e., key=value1,value2,..valuen.

The key words are:

Key	Description
InitiatorName	Initiator's Name
TargetName	Target's Name
TargetAlias	Target's Alias
InitiatorAlias	Initiator's Alias
TargetAddress	Target IP:Port

In the Login Request command, the initiator uses the keys and the appropriate iSCSI Names as values. For example:

```
I->Login Request
     InitiatorName= fqn.com.os.hostid.34567890
     InitiatorAlias= myhost
     TargetName= fqn.com.acme.diskarray.sn.8675309
```

Here, both initiator and target Name are presented. Other parameters (security, negotiation) MAY be added.

In the following example, only the initiator's Name is presented (the

key-word iscsi is used):

```
I->Login Request
     InitiatorName= fqn.com.os.hostid.34567890
     TargetName= iscsi
```

Other parameters (security, negotiation) MAY be added.

B.3. Response Format

The response to the login request can be to accept the request, to reject it or to proceed for further processing (authentication). This status should be reflected on the response message.

B.4. Examples

B.4.1 Successful login, known target:

```
I->Login Request
   InitiatorName= fqn.com.os.hostid.34567890
   InitiatorAlias= myhost
   TargetName= fqn.com.acme.diskarray.sn.8675309
```

If no further process is needed:

```
T->Login Response ("login accept 00", F set)
   TargetName= fqn.com.acme.diskarray.sn.8675309
   TargetAlias= foo
```

Or, if more authentication and/or negotiation is required:

```
T->Login Response ("challenge 20", F clear)
    .
    . authentication/negotiation
    .
T->Login Response ("login accept", F set)
   TargetName= fqn.com.acme.diskarray.sn.8675309
   TargetAlias= foo
```

In this case, target name is specified in the request. The response Reflects the iSCSI Names, indicating successful login.
Target Alias MAY be presented.

B.4.2 Successful login, unknown target:

```
I->Login Request
   InitiatorName= fqn.com.os.hostid.34567890
   InitiatorAlias= myhost
   TargetName= iscsi
    .
    . authentication/negotiation
```

```
    .
T->Login Response ("login accept", F set)
   TargetName= fqn.com.acme.diskarray.sn.8675309
   TargetAlias= foo

If there is a single iSCSI Name at the IP address and TCP port
to which the initiator has connected, this will work.
he target returns its  Name so the initiator can keep it
for future use.

Note that in the case of partial response, the target Name is
reflected Only after the authentication process.

B.4.3 Login to a canonical target, for the SendTargets command.

The initiator MUST use the key word iscsi as target's Name:

I->Login Request
   InitiatorName= fqn.com.os.hostid.34567890
   InitiatorAlias= myhost
   TargetName= iscsi
    .
    . authentication/negotiation
    .
T->Login Response ("login accept", F set)
   TargetName= iscsi

Since the target returned a iSCSI Name of "iscsi", the initiator
 MAY now use the SendTargets text command to find out which
target Names are actually supported at this address.
It will then create new connections for each target, and do
the login scenario shown in
A.4.1.

B.4.4 Redirection

If a target has moved, or is accessible only via a proxy, the target
may respond with one of several redirection status codes, along with
one or more TargetAddress fields specifying the new location(s) of
the target.

Note that a "moving target" is not changing its identity, or Name. It
is only changing its address.  A target returning a redirect status
SHOULD also include one or more TargetAddress fields specifying the
new locations of the target.

For example, if the target moved temporarily:

I->Login Request

   InitiatorName= fqn.com.os.hostid.34567890
   InitiatorAlias= myhost
```

```
   TargetName= fqn.com.acme.diskarray.sn.8675309
    .
    . authentication/negotiation
    .
  T->Login Response ("Target moved temporarily 31", F set)
   TargetName= fqn.com.acme.diskarray.sn.8675309
   TargetAddress= 10.1.40.50:384
   TargetAddress= storage1.mydata.com
```

(The same goes for the permanent move - code 32). Note that if TCP port is not specified, the canonical port is assumed.

The login response terminates the session and the initiator SHOULD start a new login session with the forwarded target. Further parameters MAY be reflected on other key=value pairs.

Or, if a proxy is required for this target:

```
I->Login Request
   InitiatorName= fqn.com.os.hostid.34567890
   InitiatorAlias= myhost
   TargetName= fqn.com.acme.diskarray.sn.8675309
    .
    . authentication/negotiation
    .
T->Login Response ("Proxy required 33", F set)
   TargetName= fqn.com.acme.diskarray.sn.8675309
   TargetAddress= 10.1.40.50:384
```

If more than one proxy exist, their addresses can be reflected in a list format.

B.4.5 Login fail

In case of login failure - forbidden target, unauthorized initiator and so on, the target terminates the session.

```
I->Login Request
   InitiatorName= fqn.com.os.hostid.34567890
   TargetName= fqn.com.acme.diskarray.sn.8675309
T->Login Response ("forbidden target 42", F set)
```

In this example, the initiator is not allowed on the required target. The initiator SHOULD terminate the login session and MAY try connecting to another target.

```
I->Login Request
   InitiatorName= fqn.com.os.hostid.34567890
   TargetName= fqn.com.acme.diskarray.sn.8675309
T->Login Response ("Target removed 44", F set)
```

In this case the target has been removed. In contrast with codes 31

and 32 (in B.4.4), no redirection information is supplied.

I->Login Request

```
   InitiatorName= fqn.com.os.hostid.34567890
   TargetName= fqn.com.acme.diskarray.sn.8675309
T->Login Response ("Target Conflict 45", F set)
```

Here, the target is busy with another initiator and cannot handle another one. The initiator MAY try again later. This can be the case of simple devices that can handle one device or the target has reached the limit of its initiators' capacity. In contrast to the previous examples, this rejection is temporary.

```
I->Login Request
   InitiatorName= fqn.com.os.hostid.34567890
   TargetName= fqn.com.acme.diskarray.sn.8675309
T->Login Response ("Target removed 44", F set)
```

Here, the target has been removed. The initiator SHOULD terminate the login session. It MAY query the SNS for the new location of the target. (This should apply for the case when the target was not found - code 44).

In any case of the 4x and 5x class, there is no name reflection on the Login response. However, detailed messages can be carried on other key=value pairs.

B.4.6 Proxy Login

When the initiator logs to a target via an (iSCSI) proxy, the following procedure is applied:

The initiator connects to the proxy's port and sends a login request of the destination target's Name and address:

```
I->Login Request
   InitiatorName= fqn.com.os.hostid.34567890
   TargetName= fqn.com.acme.diskarray.sn.8675309
   TargetAddress= 10.1.30.75:240
```

Using the TargetAddress key saves the discovery process of the target. The proxy logs into the required target with the initiator's Name. The results of the login are reflected back to the initiator.

Note that a transparent (iSCSI) proxy does not have an iSCSI Name of its own.

Appendix C: iSCSI Proxies and Firewalls Taxonomy

iSCSI has been designed to allow SCSI initiators and targets

to communicate over an arbitrary network. This, making some assumptions about authentication and security, means that in theory, the whole internet could be used as one giant storage network.

However, there are many access and scaling problems that would come up when this is attempted.

1. Most iSCSI targets are only meant to be accessed by one or

   a few initiators. Discovering everything would be silly.

2. The initiator and target may be owned by separate entities, each with their own directory services, authentication, and other schemes. An iSCSI-aware proxy may be required to map between these things.

3. Many environments use non-routable IP addresses, such as the 10. network.

For these and other reasons, various types of firewalls and proxies will be deployed for iSCSI, similar in nature to those already handling protocols such as HTTP and FTP.

1. Port Redirector

A port redirector is a stateless device that is not aware of iSCSI. It is used to do Network Address Translation (NAT), which can map IP addresses between routable and non-routable domains, as well as map TCP ports. While devices providing these capabilities can often filter based on IP addresses and TCP ports, they generally do not provide meaningful security, and are used instead to resolve internal network routing issues.

Since it is entirely possible that these devices are used as routers and/or aggregators between a firewall and an iSCSI initiator or target, iSCSI connections must be operable through them.

Effects on iSCSI:

- iSCSI-level data integrity checks must not include information from the TCP or IP headers, as these may be changed in between the initiator and target.

- iSCSI messages that specify a particular initiator or target, such as login requests and third party requests, should specify the initiator or target in a location-independent manner. This is accomplished using the iSCSI Name.

2. SOCKS server

A SOCKS server can be used to map TCP connections from one network domain to another. It is aware of the state of each TCP connection.

The SOCKS server provides authenticated firewall traversal for applications that are not firewall-aware. Conceptually, SOCKS is a "shim-layer" that exists between the application (i.e., iSCSI) and TCP.

To use SOCKS, the iSCSI initiator must be modified to use the encapsulation routines in the SOCKS library. The initiator the opens up a TCP connection to the SOCKS server, typically on the canonical SOCKS port 1080. A subnegotiation then occurs, during which the initiator is either authenticated or denied the connection request. If authenticated, the SOCKS server then

opens a TCP connection to the iSCSI target using addressing information sent to it by the initiator in the SOCKS shim. The SOCKS server then forwards iSCSI commands, data, and responses between the iSCSI initiator and target.

Use of the SOCKS server requires special modifications to the iSCSI initiator. No modifications are required to the iSCSI target.

As a SOCKS server can map most of the addresses and information contained within the IP and TCP headers, including sequence numbers, its effects on iSCSI are identical to those in the port redirector.

3. iSCSI Proxy

An iSCSI proxy is similar to proxies available in HTTP. The initiator is aware of the actual addresses of the targets, but instead of connecting to the addresses, connects instead to a proxy's address. The proxy, in turn, connects to the actual targets. This is similar to the HTTP/1.1 proxy, where the client passes the entire URL (including IP and TCP address) to the proxy, rather than just the path name.

An iSCSI proxy can provide some good iSCSI-level access control and other functionality, while adding fairly light configuration responsibilities.

Effects on iSCSI:

- When logging in to a target at a proxy address instead of the actual address, the target should include the TargetAddress (IP address and TCP port) of the target, in addition to its iSCSI Name. Note, however, that this directly conflicts with the statement made regarding NAT firewalls. Since the iSCSI Name can uniquely identify an iSCSI device, the TargetAddress must then be used by

the proxy as a hint on where to find the iSCSI Name, and not as the final authority.

- This is beginning to be covered in the iSCSI specification.

Having the address passed with the iSCSI Name would allow an iSCSI proxy to exist without extra configuration or name services. Using this type of proxy can eliminate the need to implement SOCKS.

4. SCSI gateway

This gateway presents logical targets (iSCSI Names) to the initiators, and maps them to real iSCSI targets as it chooses. The initiator sees this gateway as a real iSCSI target, and is unaware of any proxy or gateway behavior. The gateway may manufacture its own iSCSI Names,or use those provided by the real devices. This type of gateway is used to represent parallel SCSI, Fibre Channel, SSA, or other devices as iSCSI devices.

Nearly any capability that could be imagined is possible with this

type of gateway, but it may require more configuration than an iSCSI proxy.

Effects on iSCSI:

- Since the initiator is unaware of any addresses beyond the gateway, the gateway's own address is for all practial purposes the real address of a target. Only the iSCSI Name needs to be passed. This is already done in iSCSI, so there are no further requirements to support SCSI gateways.

5. Stateful Inspection Firewall (stealth iSCSI firewall)

The Stealth model would exist as an iSCSI-aware firewall, that is invisible to the initiator, but provides capabilities found in the iSCSI proxy.

Effects on iSCSI:

- Since this is invisible, I don't think there are any additional requirements on the iSCSI protocol for this one.

This one is more difficult in some ways to implement, simply because it has to be part of a standard firewall product, rather than part of an iSCSI-type product. For this reason, I would not expect to see these implemented for a while.

Also note that this type of firewall is only effective in the outbound direction (allowing an initiator behind the

firewall to connect to an outside target), unless the iSCSI target is located in a DMZ. It does not provide adequate security otherwise.

8. References

[1] Pascoe, R., "Building Networks on the Fly", in IEEE Spectrum, March, 2001.

[2] John, R., "UPnP, Jini and Salutation- A look at some popular coordination frameworks for future networked devices", http://www.cswl.com/whiteppr/tech/upnp.html", June 17, 1999.

[3] http://www.srvloc.org

[4] Freed, N., "Behavior of and Requirements for Internet Firewalls", RFC 2979, October 2000.

[5] ANSI/IEEE Std 802-1990, Name: IEEE Standards for Local and Metropolitan Area Networks: Overview and Architecture

[6] Kessler, G. and Shepard, S., "A Primer On Internet and TCP/IP Tools
and Utilities", RFC 2151, June 1997.

[7] Satran, J., Sapuntzakis, C., Wakeley, M., Von Stamwitz, P., Haagens, R., Zeidner, E., Dalle Ore, L., Klein, Y., "iSCSI", draft-ietf-ips-iscsi-00.txt, February, 2000.

[8] Gibbons, K., Tseng, J. and Monia, C., "iSNS Internet Storage Name Service", draft-tseng-ips-isns-00.txt, October 2000.

[9] RFC 1737, "Functional Requirements for Uniform Resource Names".

[10] RFC 1035, "Domain Names - Implementation and Specification".
OUI - "IEEE OUI and Company_Id Assignments", http://standards.ieee.org/regauth/oui/index.shtml
[11]EUI - "Guidelines for 64-bit Global Identifier (EUI-64) Registration Authority
http://standards.ieee.org/regauth/oui/tutorials/EUI64.html

[12] RFC 2396, "Uniform Resource Identifiers".
[13] RFC 2276, "Architectural Principles of URN Resolution".

[14] RFC 2483, "URI Resolution Services".

[15] RFC 2141, "URN Syntax".

[16] RFC 2611, "URN Namespace Definition Mechanisms".

[17] RFC 2608, SLP Version 2.
[18] RFC 2610, DHCP Options for the Service Location Protocol.

[19] P. Sarkar et al, "A Standard for Bootstrapping Clients using the iSCSI Protocol", draft-ietf-ips-iscsi-boot-01.
[21] M. Bakke et al,"Finding iSCSI Targets and Name Servers using SLP", draft-bakke-iscsi-SLP-template.00.
[More references to add to the end of NDT:]

[22]] Sun Microsystems, "Java Language Specification", section 7.7 "Unique Package Names", 2000,

http://java.sun.com/docs/books/jls/second_edition/html/jTOC.doc.html.

[23]] Flanagan, et. al, "Java in a Nutshell", O'Reilly, 1997.

6. Contact Author

Kaladhar Voruganti
650 Harry Road
IBM Almaden Research
San Jose, CA
USA
Email: kaladhar@us.ibm.com