VMS Software

VSI OpenVMS

VSI DECnet-Plus Planning Guide

Document Number: DO-DNTPPL-01A

Publication Date: May 2024

Operating System and Version: VSI OpenVMS IA-64 Version 8.4-1H1 or higher
VSI OpenVMS Alpha Version 8.4-2L1 or higher

VMS Software, Inc. (VSI)
Boston, Massachusetts, USA

VSI DECnet-Plus Planning Guide

Legal Notice

Preface

This manual provides an overview of the transition and planning tasks necessary to move from DECnet for OpenVMS (Phase IV) to DECnet-Plus for OpenVMS (Phase V) and DECnet-Plus/OSI for UNIX. It describes how to:

- Use global network addresses
- Obtain a unique initial domain part (IDP) for your DECnet Phase V network
- Set up your namespace and enter node names and addresses into it
- Use name service access features
- Set up DECdts clerk nodes using DECdts software

See your *Software Product Description (SPD)* for detailed information about new features and product requirements.

1. About VSI

VMS Software, Inc. (VSI) is an independent software company licensed by Hewlett Packard Enterprise to develop and support the OpenVMS operating system.

2. Intended Audience

This book is written for:

- Network planners and managers, both existing DECnet Phase IV users and new DECnet-Plus customers
- OpenVMS system managers
- Installers of OpenVMS and UNIX
- Namespace planners and managers
- DECdts planners and managers
- Managers of the following Open Systems Interconnection (OSI) features:
    - OSI Transport connections
    - X.25 communications
    - Wide area network device drivers (WANDD)
    - Remote OSI file operations
    - DECnet Phase V virtual terminal
    - Writing and running additional OSI applications

3. Document Structure

The manual consists of the following chapters:

Chapter 1	Describes the transition concepts.
Chapter 2	Helps you plan for an orderly, efficient transition from a Phase IV network to a DECnet Phase V network.
Chapter 3	Discusses your immediate transition tasks and the tools you need to perform them.
Chapter 4	Discusses the format of NSAP addresses for DECnet-Plus systems and describes how to get unique identification for your DECnet Phase V network.
Chapter 5	Describes DECdns, Local Namespace, and DECdts concepts.
Chapter 6	Provides general guidelines for all DECnet-Plus names and specific guidelines for naming clearinghouses and namespaces.
Chapter 7	Contains basic DECdns planning guidelines if you choose to plan and implement a DECdns distributed namespace.
Chapter 8	Provides additional guidelines to plan your distributed namespace.
Chapter 9	Shows an example of a DECdns Namespace design.
Chapter 10	Describes how to plan your DECdts implementation, including personnel selection for the planning process, and planning for DECdts on a LAN, an extended LAN, or a WAN.

4. VSI Encourages Your Comments

You may send comments or suggestions regarding this manual or any VSI document by sending electronic mail to the following Internet address: <docinfo@vmssoftware.com>. Users who have VSI OpenVMS support contracts through VSI can contact <support@vmssoftware.com> for help with this product.

5. OpenVMS Documentation

The full VSI OpenVMS documentation set can be found on the VMS Software Documentation webpage at https://docs.vmssoftware.com.

6. Terminology

These terms are used interchangeably:

- Transition and migration
- Phase IV and DECnet Phase IV
- Phase V and DECnet Phase V
- System and node
- End system and end node

- Intermediate system and router
- Multivendor
- Link state and link state routing algorithm, link state protocol, DECnet Phase V routing algorithm, or DECnet Phase V routing
- Name service and directory service

7. Typographical Conventions

VMScluster systems are now referred to as OpenVMS Cluster systems. Unless otherwise specified, references to OpenVMS Cluster systems or clusters in this document are synonymous with VMScluster systems.

The contents of the display examples for some utility commands described in this manual may differ slightly from the actual output provided by these commands on your system. However, when the behavior of a command differs significantly between OpenVMS Alpha and Integrity servers, that behavior is described in text and rendered, as appropriate, in separate examples.

In this manual, every use of DECwindows and DECwindows Motif refers to VSI DECwindows Motif for OpenVMS software.

The following conventions are also used in this manual:

Convention	**Meaning**
Ctrl/ *x*	A sequence such as **Ctrl/** *x* indicates that you must hold down the key labeled Ctrl while you press another key or a pointing device button.
PF1 *x*	A sequence such as PF1 *x* indicates that you must first press and release the key labeled PF1 and then press and release another key or a pointing device button.
Return	In examples, a key name enclosed in a box indicates that you press a key on the keyboard. (In text, a key name is not enclosed in a box.)
…	A horizontal ellipsis in examples indicates one of the following possibilities: • Additional optional arguments in a statement have been omitted. • The preceding item or items can be repeated one or more times. • Additional parameters, values, or other information can be entered.
. . .	A vertical ellipsis indicates the omission of items from a code example or command format; the items are omitted because they are not important to the topic being discussed.
( )	In command format descriptions, parentheses indicate that you must enclose the options in parentheses if you choose more than one.
[ ]	In command format descriptions, brackets indicate optional choices. You can choose one or more items or no items. Do not type the brackets on the command line. However, you must include the brackets in the syntax for OpenVMS directory specifications and for a substring specification in an assignment statement.
[|]	In command format descriptions, vertical bars separate choices within brackets or braces. Within brackets, the choices are options; within braces, at least one choice is required. Do not type the vertical bars on the command line.

Convention	Meaning
{ }	In command format descriptions, braces indicate required choices; you must choose at least one of the items listed. Do not type the braces on the command line.
bold text	This typeface represents the introduction of a new term. It also represents the name of an argument, an attribute, or a reason.
italic text	Italic text indicates important information, complete titles of manuals, or variables. Variables include information that varies in system output (Internal error *number*), in command lines (/PRODUCER= *name*), and in command parameters in text (where *dd* represents the predefined code for the device type).
UPPERCASE TEXT	Uppercase text indicates a command, the name of a routine, the name of a file, or the abbreviation for a system privilege.
`Monospace type`	Monospace type indicates code examples and interactive screen displays. In the C programming language, monospace type in text identifies the following elements: keywords, the names of independently compiled external functions and files, syntax summaries, and references to variables or identifiers introduced in an example.
-	A hyphen at the end of a command format description, command line, or code line indicates that the command or statement continues on the following line.
numbers	All numbers in text are assumed to be decimal unless otherwise noted. Nondecimal radixes—binary, octal, or hexadecimal—are explicitly indicated.

Chapter 1. Preplanning: Understanding the Transition Process

Chapters 1 through 4 of this book describe the considerations that will help you determine how best to transition your network. The following sections define what the transition is, explain why you would want to transition your systems from the DECnet Phase IV to DECnet Phase V architecture, and help you plan how long this transition might take.

1.1. Transition Defined

Transition is the process of migrating a network from DECnet Phase IV to DECnet-Plus (Phase V) by changing:

- Node-name-to-address mapping from the use of a Phase IV local permanent node databases to a DECdns distributed namespace, DNS/BIND, or to a Local namespace
- Network management from the use of NCP to NCL

Transition is a multistep process. Some tasks you perform immediately as part of the DECnet-Plus installation/configuration procedure. Others you do immediately afterwards, still others perhaps later, as your network's needs change. In addition, some steps target the transition of one system, while other steps are part of the network's overall transition to DECnet-Plus.

Transition begins with the installation of the DECnet-Plus software onto the first node in the network. With this installation/configuration, the system becomes a DECnet-Plus system. Your network operates in a **transition environment** from the transition of the first node until all DECnet nodes migrate to DECnet-Plus.

The steps of transition are:

1. Developing a transition plan for your network.
2. Installing the DECnet-Plus software onto the first node.
3. Configuring node-name-to-address mapping in at least one of the following ways:
    - Using the Local namespace.
    - Configuring the node as a DECdns clerk in an existing distributed namespace.
    - For an OpenVMS VAX or UNIX system, configuring the node as a DECdns server and creating a new distributed namespace. (DECdns server software is not available for OpenVMS Alpha systems.)
    - Using DNS/BIND.
4. Performing the remaining transition-related network management tasks.
5. Migrating to DECnet-Plus all the nodes you planned to migrate.

1.1.1. Why Make the Transition to DECnet-Plus?

DECnet Phase IV supports DECnet applications only. It does not support OSI (Open Systems Interconnection) and TCP/IP applications, which must use separate protocol stacks that DECnet Phase IV does not provide.

With DECnet Phase V (formerly known as DECnet/OSI and now, with Version 7.1, DECnet-Plus), your DECnet applications can communicate (without change) with peer OSI and DECnet applications on any system running DECnet Phase IV, Phase V, or OSI software. DECnet-Plus integrates the OSI protocol stack with the DECnet protocol stack and includes the ability to run DECnet and/or OSI applications over TCP/IP.

A growing number of emerging network applications are designed to be transported over the Open Systems Interconnection. DECnet-Plus provides a collection of standard OSI applications such as File Transfer, Access, and Management (FTAM) and Virtual Terminal (VT), plus a complete library of OSI transport protocols.

With DECnet-Plus, you can use one or more of the following name services to translate network names to node numbers and back, depending on the transport used (DECnet, OSI, or TCP/IP):

- Local namespace (similar to DECnet Phase IV's NETNODE_REMOTE.DAT node database)
- DECdns (Distributed Naming Service)
- DNS/BIND name resolution protocol (for TCP/IP)

When a user provides a node name to a DECnet or OSI application, the application translates it using the appropriate name service.

TCP/IP support is crucial for the many customers who need access to the TCP/IP-based Internet and to the growing number of low-cost, IP-compatible routers and carrier services. DECnet-Plus, in combination with a TCP/IP protocol stack for OpenVMS and UNIX, allows a DECnet customer to convert to TCP/IP network protocols without having to give up existing DECnet or OSI applications.

The DECnet over TCP/IP (DOTI) feature (RFC 1859), included with DECnet-Plus, allows DECnet applications to communicate over TCP/IP. The OSI over TCP/IP support (RFC 1006), included with DECnet-Plus, allows OSI applications to communicate over TCP/IP.

For new OpenVMS and UNIX systems, VSI now bundles run-time licenses for DECnet-Plus and TCP/IP services. If you are an existing DECnet support customer, you receive DECnet-Plus automatically but need to purchase a license to use TCP/IP Services for OpenVMS.

If you want to migrate your system to TCP/IP, you must install and configure the TCP/IP stack as part of your transition. Both DECnet-Plus and TCP/IP transports can be up and running concurrently and you can make the transition to TCP/IP gradually. You do not need to choose between OSI or TCP/IP. You can even choose to run your DECnet-Plus node with DECnet Phase IV functionality only and begin to use the OSI and/or TCP/IP features as you become more familiar with them.

1.1.2. How Long Does Transition Take?

The time it takes to make a complete network transition depends on your strategy. DECnet-Plus software gives you the flexibility to decide how and when to make the transition from Phase IV to DECnet-Plus. No definite time period is prescribed for all networks.

You can choose to make the transition gradually, perhaps installing DECnet-Plus software in a single area first to see how it operates and how to take advantage of its features. Or, you can install DECnet-Plus software on end systems in several areas. For example, you may chose to operate with a mixture of Phase IV and DECnet-Plus software within each area. At some point, you might decide to use the link state routing algorithm and take advantage of OSI addressing that is beyond Phase IV limitations, that is, extended addressing.

Note

The time you spend planning the transition is an essential part of your total transition time frame.

The time it takes to migrate one system from DECnet Phase IV to DECnet-Plus is the time it takes to perform the installation and configuration procedures on this system. However, if this is the first DECnet-Plus node on the network, you must spend some additional time on other transition tasks, such as populating the namespace, if you are creating a DECdns namespace, and converting some Phase IV databases.

1.1.3. What Is the Transition Environment?

During transition, some or all of your network operates in a mixed environment of Phase IV and DECnet-Plus nodes, either spread throughout the network or grouped into all-Phase-IV or all-DECnet-Phase-V areas.

In this transition environment, you have a mixture of Phase IV and DECnet Phase V features, including addressing, routing, and network management.

1.1.4. When Do DECnet-Plus Features Become Available?

Most features offered by the VSI DECnet-Plus for OpenVMS and DECnet/OSI for UNIX products become available upon installation and configuration of the product software. However, RFC 1006, RFC 1859, and DNS/BIND require the installation of additional TCP/IP software. DECnet-Plus features include:

- OSI applications over TCP/IP (the RFC 1006 feature)
- DECnet applications over TCP/IP (the RFC 1859 feature)
- DECdns for node-name-to-address mapping using a distributed namespace
- Local namespace for node-name-to-address mapping using a local database
- DNS/BIND as a naming service with DECnet-Plus RFC 1006 or RFC 1859 features
- New network management structure and interfaces
- Multicircuit end-system capability
- OSI addressing (addressing beyond Phase IV limitations)
- Multihoming
- End-system autoconfiguration

In an all-end-system LAN, these systems autoconfigure their addresses using the default local area address of 49::00-40.

However, you can override this feature with network management commands and with the `decnet_register` management support tool.

- For VSI DECnet-Plus for OpenVMS, additional features such as OSI remote file operations (FTAM).

1.1.5. When Does Transition End?

Transition from Phase IV to DECnet-Plus ends when all:

- Systems in the network run DECnet-Plus software
- Routers operate with the DECnet Phase V link state routing algorithm

Because DECnet-Plus features allow for flexible transition, the network can operate with a mixture of Phase IV and DECnet-Plus systems for as long as your network requires. Therefore, as network manager, you decide how long the network continues to operate in a transition environment.

1.2. Transition Features

DECnet-Plus software provides the following features to help you during transition:

- Coexistence of Phase IV and DECnet-Plus: backward compatibility

  DECnet-Plus software is compatible with DECnet Phase IV software; DECnet-Plus systems can coexist with and fully interoperate with Phase IV nodes. In addition to supporting OSI protocols, DECnet-Plus systems continue to support all Phase IV protocols (with the exception of the Phase IV NICE network management protocol). With this interoperability, you do not have to migrate your network — or even an entire area — all at once.

  For compatibility, DECnet-Plus systems can run two transports, both the Network Architecture (NA) Network Services Protocol (NSP) and OSI transport protocol. In addition, DECnet-Plus end systems send routing packets and data link packets in either DECnet Phase V format or Phase IV format, depending on the phase of the receiving system.

- Addressing options

  DECnet-Plus systems require a Phase IV-compatible address to communicate with Phase IV nodes. On the same DECnet-Plus system, you can also assign an OSI address that is not Phase IV-compatible.

  For details, see Section 1.3.

- Node synonyms

  A **node synonym** is a Phase IV-style node name, between 1 and 6 characters long, that is unique within your namespace. (See Section 5.4 for a complete explanation of node synonyms.)

  The node synonym is required for Phase IV applications that can handle only a maximum of six-character node names.

Use your Phase IV node name as your synonym. The default node synonym is the first six characters of your simple name, the string that follows the last period of your full name. For example, if you specify `XYZ_CORP:.sales.east_coast.ElanaCole` as a DECnet-Plus full name during the configuration procedure, your simple name is `ElanaCole` and your default node synonym is `ElanaC`.

The configuration procedure gives you the option of specifying a different node synonym.

You do not need a node synonym if:

- Your network has made a full transition to all DECnet-Plus and other OSI-compliant systems.
- These systems do not run any applications that are limited to six-character node names.

OpenVMS: Node synonyms allow for backward compatibility with older applications that cannot use long domain names. DECnet-Plus allows for node synonyms to provide backward compatibility DECnet Phase IV node names.

DECnet programming interfaces $QIO and $IPC also provide long name support. User-written applications using these interfaces can use long names.

- Coexistence of Phase V routing algorithm and Phase IV routing algorithm

  In DECnet-Plus, the Phase V routing algorithm, called the **link state protocol**, can coexist with the Phase IV routing algorithm, called the **routing vector protocol**. See Section 2.3.5 for details.

- The Local namespace

  DECnet-Plus includes a Local namespace, independent of DECdns. DECnet-Plus also gives you an option of maintaining DECnet Phase IV functionality until your network makes the transition to DECnet Phase V. The Local namespace replaces functionality previously provided by the DECdns Local Naming Option (LNO). Depending on the number of address towers stored, the Local namespace is designed to scale to at least 100,000 nodes. For more information, refer to Section 1.5.1 and Section 5.1.

1.3. Addressing

The DECnet-Plus software supports a new address format, the OSI addressing format. OSI addresses (NSAPs) can be bigger and longer than Phase IV addresses, or they can fall within the limits of Phase IV addressing. OSI addresses that falls within the limits of Phase IV addressing are referred to as Phase IV-compatible addresses. DECnet Phase V addresses that fall outside Phase IV address space are referred to as extended addresses. Refer to Section 4.5 for more information.

Making communication possible between a Phase IV node and a DECnet-Plus system depends on the address of the DECnet-Plus system and on your routing configurations. You have the following options for assigning addresses to DECnet-Plus systems:

- You can assign OSI addresses that are also Phase IV-compatible to all DECnet-Plus systems.
- You can use OSI addresses that are not Phase IV-compatible on all DECnet-Plus systems, if you meet the requirements listed in Section 2.3.4.
- You can have a mixture of Phase IV-compatible and extended addresses:

- By having Phase IV areas and DECnet Phase V areas.
- Within a single area, if the area has multiple area addresses, one of which is a Phase IV-compatible address and another which is an extended address.

  You can have an area with DECnet Phase V routers running link state in which some nodes have Phase IV addresses and some have OSI addresses that are beyond Phase IV limitations. As long as the routers all have Phase IV addresses, the nodes with Phase IV addresses can communicate with other Phase IV nodes outside the area. Nodes with extended OSI addresses cannot communicate with Phase IV nodes in other areas.

- You can multihome each system, giving it from one to three different addresses (see Section 1.3.4).

1.3.1. Phase IV Addressing and OSI Addressing: A Comparison

The DECnet Phase IV network-address format consists of 16 bits (2 bytes) of information:

- 6 bits for the area address
- 10 bits for the node identifier

This format limits the network to 63 areas and a maximum of 1023 nodes per area. In contrast, the OSI address format can be up to 20 bytes long, thus extending network addresses beyond Phase IV limits.

Table 1.1 lists the differences between Phase IV and OSI addresses. Figure 1.1 shows the address parts listed in Table 1.1. For a complete description of OSI addresses and their individual parts, see Chapter 4.

Table 1.1. Comparison of Phase IV and OSI Addresses

Phase IV Address	**OSI Address**
2 bytes long	Up to 20 bytes long

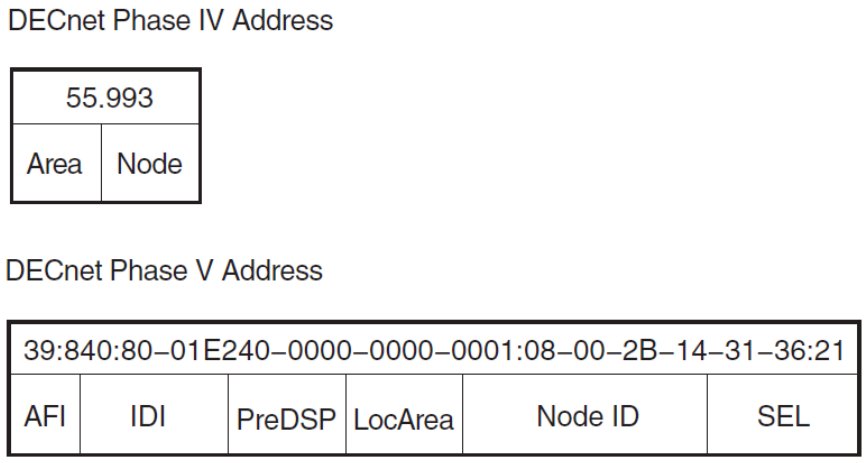
Consists of two parts:
• Area number (1 to 63)
• Node number (1 to 1023)
	Consists of two parts:
	• Initial domain part (IDP), which has two fields:
	  • Authority and format identifier (AFI)
	  • Initial domain identifier (IDI)
	• Domain-specific part (DSP), which has four fields:
	  • PreDSP[1] (the size of this field can be 0)
	  • Local area (LocArea)
	  • Node ID
	  • Selector (SEL)

Area = area number (1 to 63)	Area = IDP + preDSP [1] + LocArea fields in DSP
Node address = area number + node number	System address = IDP + preDSP + LocArea + node ID is called the **network entity title** (NET).

Phase IV Address	OSI Address
	The NET is the address by which the Network layer is identified. It identifies a particular system in a particular network.
No transport selection (NSP assumed)	Provides SEL field for transport selection (NSP or OSI transport). AN OSI address (NET) that also includes a SEL field other than 00 is a **network service access point** (NSAP). The NSAP is a global network address. It is the addressable point at which a network entity provides the network service to a network user. It identifies both the particular network system and the transport on that system that is to receive the data.

[1]Which is also called the "high-order part of the DSP" (HO-DSP).

Figure 1.1. Examples of Phase IV and DECnet Phase V Addresses



Phase IV nodes can communicate with DECnet-Plus systems only if the DECnet-Plus systems have Phase IV-compatible addresses. A **Phase IV-compatible address** is an OSI address that has a Phase IV address encoded within it.

You can use link state routing with Phase IV-compatible addressing.

Phase IV-compatible addresses continue to be suitable for many DECnet Phase V networks; using them can ease migration to link state routing. You can choose to use Phase IV-compatible addressing until all, or most, network servers and services are running DECnet-Plus.

1.3.2. Advantages of Using OSI Addresses That Are Also Phase IV Compatible

Use Phase IV-compatible addresses for DECnet-Plus systems that:

- Need to communicate with Phase IV nodes, either for the short term (until the Phase IV nodes migrate to DECnet-Plus), or for the long term (because those nodes cannot migrate).
- Exist in the same area as Phase IV routers.

1.3.3. Advantages of Using Extended Addressing

Use Phase V addresses that are larger than Phase IV limitations if one or more of the following apply to your network:

- The network size requires more than 63 areas per network and/or 1023 nodes per area.
- You plan to connect your network to other OSI networks.

  Phase V addresses include an initial domain part (IDP), a unique identifier for a network that permits connection to other networks.
- The Phase V systems do not need to communicate with Phase IV nodes.
- You want to allow end systems to autoconfigure their network addresses during the installation/configuration process.

1.3.4. Advantages of Using Multihoming

You can assign more than one address to a system. This practice is called **multihoming**. For example, you can give any DECnet-Plus system both a Phase IV-compatible OSI address and an extended address. The benefit is that you can communicate with Phase IV, Phase V, and other OSI systems.

DECnet-Plus allows you to assign to a node as many as three addresses.

Multihoming also makes it easy for you to belong to more than one OSI network. This feature is particularly useful when you want to combine networks. Rather than have all the systems in both networks get new addresses that reflect the new combined network, the systems that need to participate in both networks can have an address in each one.

1.3.5. Autoconfiguration of Addresses

Two ways exist to configure network addresses: autoconfiguring them or manually configuring them. **Autoconfiguration** means that the adjacent router configures an end node's network address. This is the easier way to configure NETs. If you have a DECnet Phase V router adjacent to your system (on the same LAN or connected to your system by a point-to-point link), you can let the router configure your network addresses for you.

Note

If you have a multivendor OSI-compliant router adjacent to your end system, do not use autoconfiguration unless you know that the router uses NETs with a selector of `00`. This restriction applies even if you have a DECnet Phase V router as well as the multivendor router on the same LAN. Multivendor routers that specify NETs differently can cause incorrect autoconfiguration.

1.4. Routing

DECnet-Plus end systems can communicate with both DECnet Phase V routers and Phase IV routers. For Phase-IV compatibility, DECnet Phase V routers can communicate in Phase IV routing packet format. While running the Phase IV algorithm, DECnet Phase V routers provide full routing capability for:

- Phase IV nodes

- DECnet-Plus systems with Phase IV-compatible addresses
- Multivendor systems that conform to OSI-packet and OSI-addressing standards, and that have addresses within Phase IV limits.

Like all DECnet-Plus systems, DECnet Phase V routers support OSI addressing. You can set these systems to use either DECnet Phase V link state or Phase IV routing vector for routing on either level 1 or level 2, depending on your network's needs. Note that you can also choose between using a dedicated router in your network to perform routing or to configure your system as a host-based router.

You can configure end systems with DECnet Phase V addresses beyond the limits of Phase IV addressing if your routing infrastructure supports it. For complete information about routing topology choices, see the documentation set of your routing product.

1.4.1. Interdomain Routing

DECnet-Plus supports connections to other networks through **interdomain routing**, the ability for a network (one **routing domain**) to connect to a different routing domain.

The IDP of an OSI address provides a unique identifier for the network. The ISO protocols and the OSI addressing scheme enable global multivendor interoperability.

When planning your transition from Phase IV to Phase V, allow for connections to expand to a global network, if appropriate for your enterprise. Also consider your network's accessibility and security needs if it were to become part of a global network.

With interdomain routing, your network can connect either to another network that is based on the Network Architecture (NA) or to a network with a multivendor routing scheme. The two connected networks remain distinct. Communication takes place:

- Between systems with addresses that you defined in reachable-address tables that reside on level 2 routers.
- Over circuits for which you have set the routing circuit entity characteristic `DNA neighbor` to false.

Setting `DNA neighbor false` ensures that your network passes no routing information to the other. However, this configuration may not provide sufficient security for your network, because it does not stop any packets that the other network sends to systems in your network.

For information about setting up connections between networks, see your network management guide.

1.4.2. Level 2 Routing Between Phase IV and Phase V Areas

All level 1 routers within an area must use the same routing protocol. At level 2, however, you can have a mix of routing vector and link state. Level 2 routers running different protocols communicate through interphase links. An **interphase link** directly connects a level 2 router using routing vector with a level 2 router using link state. DECnet Phase V routers running link state use reachable-address tables to manually configure routing information across the link.

For information about how to use interphase links and reachable-address tables, see your network management guide. For further details, see the router management documentation.

1.4.3. Multivendor Routers

If your network includes multivendor OSI-compliant routers, they might not support one or more of the following DECnet-Plus features:

- Phase IV interoperability
- Autoconfiguration of end-system addresses
- Congestion avoidance
- Management by the Network Control Language (NCL), the network management utility that comes with

  DECnet-Plus

Therefore, using multivendor routers requires additional management considerations. For example, if the router does not support Phase IV interoperability, your DECnet-Plus systems may not be able to connect to any Phase IV nodes on the network. For details, see the section on coexistence with multivendor routers that do not support backward compatibility in your network management guide.

1.5. Name Services and Time Service Considerations

An important part of planning for migration to DECnet-Plus is planning for your name services and for the Distributed Time Service (DECdts).

The DECdns distributed namespace is no longer a requirement for DECnet-Plus and the Local namespace is not dependent on DECdns. However, the DECdns clerk software is still required on each node. DECnet-Plus provides access to the node name and addressing information stored in one or more name services. DECnet-Plus supports the following name services:

- Local namespace – A discrete, nondistributed namespace that stores name and address information locally in database files.
- DECdns – Distributed Name Service, a distributed, global name service.
- Domain Name System – The Domain Name System (DNS/BIND) supported for storage of IP addresses.

While configuring DECnet-Plus, the system administrator specifies one or more of the following name services to use on the node: the Local namespace, DECdns, or Domain.

1.5.1. The Local Namespace

The Local namespace is a discrete, nondistributed namespace that exists on a single node and provides that node with a local database of name and addressing information. Depending on the number of address towers stored, the Local namespace is designed to scale to at least 100,000 nodes.

The prefix `LOCAL:` (or `local:`) is reserved to indicate that the information for the node is stored in the Local namespace. DECnet-Plus recognizes that when a node full name begins with `LOCAL:`, information for that node is stored in a Local namespace. The following are typical node full names properly formatted for the Local namespace: `LOCAL:.xyz.abc` and `local:.maximum`.

Unlike DECdns, the Local namespace does not employ backtranslation directories for address-to-node-name translation.

You cannot use the DECdns Control Program (DNSCP) to manage information stored in the Local namespace. Instead, use `decnet_register` to manage the node name and address information stored in your namespace. The new `decnet_register` tool is described in your network management guide.

1.5.2. The Name Service Search Path

At configuration time, you will be asked to specify a naming service search path. This search path applies systemwide and allows DECnet-Plus to search a list of name services in a predetermined order when looking up names or addressing information. The *primary* name service (the name service to be searched first)is listed before the *secondary* name services. The secondary name services are listed in the order in which they are to be searched after the primary name service.

The search path contains a list of name service keywords. Each keyword is followed by a *naming template* that specifies a "defaulting rule" so users can enter shorter node names. In each template, the user-supplied portion of the name (usually the node's terminating name or rightmost simple name) is indicated with an asterisk (*). For example, if the DECdns template is: `"ABCDE:.xyz.*"` and a user supplies the name `foo`, then the following full name: `ABCDE:.xyz.foo` will be looked up in namespace ABCDE in the DECdns name service.

1.5.3. Name Service and Time Service Interdependencies

DECdts synchronizes the system clocks in computers connected by a network and, therefore, enables distributed applications to execute in the proper sequence even though they run on different systems.

The DECnet-Plus software, your name service, and DECdts are interdependent:

- DECnet-Plus software requires one or more of the following name services to perform node-name-to-address mapping – the Local namespace, DECdns, and/or DNS/BIND.
- DECdns servers need DECdts to synchronize system clocks so DECdns servers generate consistent timestamps.
- DECdts uses the namespace as a registry for DECdts global servers that synchronize the time on all systems in the network.

DECnet/OSI for UNIX contains DECdns and DECdts clerk software. If you use the Local namespace, DECdts uses a local configuration script. For DECdns and DECdts planning information, see Chapters 6, 7, 8, 9, and 10.

1.5.4. Mapping Node Names to Addresses

During transition, the network maps node names to addresses in the following ways:

- Phase IV nodes continue to use their permanent node databases.
- All DECnet-Plus systems use either the Local namespace or the DECdns distributed namespace, whether they have:
    - Phase IV-compatible addresses

  - Extended addresses
  - Both
- Domain Name System (DNS/BIND) should be used as a naming service for DECnet-Plus systems that are configured to take advantage of the RFC 1859 and/or RFC 1006 features.

To ensure that Phase IV nodes and DECnet-Plus systems can communicate:

1. Enter the Phase IV-compatible node name and address of each DECnet-Plus system (if the name or address is changing from what it was when the node was a Phase IV node, or if the node is new) into the permanent database of each Phase IV node.
2. If you are using the DECdns namespace and/or local namespace, register each Phase IV node in the namespace.

   Use the `decnet_register` tool to register all existing Phase IV nodes in the namespace. For complete information about using `decnet_register`, see the *VSI DECnet-Plus for OpenVMS Network Management Guide*.

To ensure that DECnet-Plus systems can communicate with other DECnet-Plus systems using RFC1006 or RFC1859, enter the Internet Addresses of the nodes in the Domain Name System (DNS/BIND).

1.6. OpenVMS Cluster Systems (OpenVMS Only)

You can combine a single OpenVMS Cluster consisting of both Phase IV and DECnet-Plus nodes with the following restrictions:

- DECnet-Plus nodes must be booted from a different system disk than the Phase IV nodes.
- DECnet-Plus nodes cannot have the same cluster alias as the Phase IV nodes. A single cluster consisting of both Phase IV and DECnet-Plus nodes can have two cluster aliases, one for the Phase IV nodes and one for the DECnet-Plus nodes.
- An OSI router is required on the same LAN as the cluster, adjacent to each DECnet-Plus node joining the alias, for the DECnet-Plus alias to work.
- Currently, the fast configuration option is not supported on a node that is running in an OpenVMS Cluster or on anode that is a DECdns server.

Given these restrictions, you can migrate an existing Phase IV OpenVMS Cluster to a DECnet-Plus OpenVMS Cluster one node at a time. DECnet-Plus no longer requires that at least one node in the cluster be a routing node, but there must be a Phase V router on the network for the alias to work. The DECnet-Plus cluster alias allows a cluster to consist of all end systems. Therefore, using multivendor routers requires additional management considerations. For example, if the router does not support Phase IV interoperability, your DECnet-Plus systems may not be able to connect to any Phase IV nodes on the network. For details, see the section on coexistence with multivendor routers that do not support backward compatibility in your network management guide.

1.7. DECnet Phase IV Applications

DECnet-Plus supports DECnet Phase IV applications as described in the following sections.

1.7.1. DECnet for OpenVMS Phase IV Applications (OpenVMS Only)

DECnet for OpenVMS Phase IV applications that use Queue I/O Request ($QIO) system service calls continue to work in DECnet-Plus without changes. You do not have to change node names to DECnet Phase V format.

VSI DECnet-Plus for OpenVMS offers both the $IPC and $QIO interfaces. OpenVMS Interprocess Communication ($IPC) system service is an operating system interface that is new with VSI DECnet-Plus for OpenVMS. This interface to the Session Control layer lets you use DECnet software to perform interprocess communications.

With $IPC, you can connect to the target application by specifying its full name or its NSAP address, as well as the Phase IV way of specifying node name and application object number and name.

For details about these programming interfaces, refer to *VSI DECnet-Plus for OpenVMS Programming*.

1.7.2. DECnet-ULTRIX Phase IV Applications (UNIX Only)

DECnet/OSI for UNIX supports ported user-written DECnet-ULTRIX Phase IV applications. However, these programs cannot take advantage of certain new features, such as location-independent services (network objects).

Although not required, recoding your custom applications has the following advantages:

- Your application can exchange tower information with services defined as network objects. Using protocol towers offers you a greater level of control. Using network objects allows your program to take advantage of the location independence of DECdns objects.
- If your Phase IV application uses system calls, VSI recommends that you recode them using the new library subroutines.

1.8. Network Management

DECnet-Plus network management implements a new architectural model, has a new structure, and provides the new NCL command interface. Once you install DECnet-Plus software on a node, you must use DECnet-Plus network management to manage that node. You should consider the following network management issues before you start your transition:

- Managing both Phase V and Phase IV nodes and applications running on these systems (see Section 1.8.1 and Section 1.8.2).
- Managing multivendor systems, if your network has them (see Section 1.8.3).
- Using NCL (see Section 3.3.4).

1.8.1. Managing a Mixture of Phase V and Phase IV Nodes

To manage the local DECnet-Plus node and remote DECnet-Plus nodes, use NCL. To manage remote Phase IV nodes, use the Network Control Program (NCP). Refer to *VSI DECnet-Plus for OpenVMS*

Network Control Language Reference Guide for information about how to issue NCL and NCP commands on a DECnet-Plus node.

Note

You cannot manage a DECnet-Plus system remotely from a Phase IV node.

1.8.2. Network Management Protocols and Constraints

DECnet-Plus network management is based on the draft ISO Common Management Information Protocol (**CMIP**) standard for network management operations. DECnet-Plus, using DNA CMIP, provides local and remote management support with the CMIP requester and listener, CML.

1.8.2.1. Phase IV Protocols in User Applications (OpenVMS Only)

DECnet Phase IV user applications that call in NML or NICE must be modified to use CML and CMIP in order to run on Phase V systems. Refer to *VSI DECnet-Plus for OpenVMS Introduction and User's Guide* for more information.

1.8.3. Managing Multivendor OSI-Compliant Systems

DECnet-Plus systems do not support remote management of other vendors' OSI-compliant systems because DNA CMIP is not compliant with OSI CMIP.

1.9. Noncompatibility with Pre-Phase IV Systems

DECnet-Plus does not interoperate with Phase III or Phase II implementations of DECnet. However, you can configure Phase III systems, such as a DECnet-RT system and a DECnet-RSX-11M-PLUS system, as end nodes in a Phase IV area if the Phase III system is adjacent to a Phase IV router (not a DECnet Phase V router running routing vector).

If your network has nodes running Phase III software, your migration plan must take this into account. You have the following options:

- Replace Phase III software with DECnet-Plus software or Phase IV software.
- Move nodes with Phase III software to a Phase IV area.
- Keep the area in which Phase III nodes reside as a Phase IV area.

1.10. Products That Do Not Migrate to DECnet Phase V

The following products and functions will not migrate to DECnet-Plus:

- End systems
  - PDP-11 systems
  - DECsystem-10 and DECSYSTEM-20 systems

- Routing and routing systems
  - DECSA-based dedicated routers
  - DECrouter 200 product
  - PDP-based routing
  - DECsystem-10- and DECSYSTEM-20-based routing
  - Multivendor non-OSI-compliant routers
- Hardware
  - DEQNA hardware for Phase IV DECnet for OpenVMS VAX systems
  - DMR11 and DMV11 hardware for VSI DECnet-Plus for OpenVMS systems
  - K series devices

  For information about supported communications controllers, see the *DECnet-Plus for OpenVMS Software Product Description (SPD)* or the DECnet/OSI for UNIX SPD.

For additional product support information for DECnet-Plus, consult the *DECnet-Plus for OpenVMS Release Notes*, as well as *DECnet-Plus for OpenVMS Installation and Basic Configuration Manual* and *DECnet/OSI for UNIX Installation and Configuration*.

You have the following transition options:

- For the Phase IV products:

  These systems can continue full communication with all DECnet-Plus nodes that have Phase IV-compatible addresses. You can isolate them in a Phase IV area, where you can continue to use your Phase IV routers.

- For DECnet Phase IV systems running host-based routing, you have the following options:
  - Keep them as Phase IV nodes in Phase IV areas
  - Migrate them to VSI DECnet-Plus for OpenVMS and configure them as end nodes in either Phase IV or Phase V areas
  - Migrate them to VSI DECnet-Plus for OpenVMS Version 7.1 and configure themas Phase V host-based routers running Routing Vector protocol
- For the non-Phase IV products and functions:

  You may need to replace them with OSI-compatible products.

You need to decide what to do with these systems and functions during migration and what place, if any, they will eventually have in the configuration of your DECnet Phase V network. You may decide to replace them, either immediately or in the future, with DECnet-Plus products.

Chapter 2. Planning the Transition

This chapter helps you plan for an orderly, efficient transition from a Phase IV network to a DECnet Phase V network.

Whatever the size or complexity of the network, a successful transition requires planning. Planning helps to ensure continued communication throughout the network during migration, with a minimum of disruption to users.

Use the checklist of transition-planning steps in Table 2.1 to draw up your network's transition plan and, later, to coordinate and execute the transition. Before using this checklist, note that:

- Planning activities are interdependent; the outcome of the planning activities for one step may influence decisions of a subsequent or previous step.

  For example, in Step 3, you might plan to configure your routers a certain way, only to find that the decisions you make later about the placement of name servers causes you to reassess the router configuration.

- Although the checklist shows planning activities in rough sequential order, you are not required to follow this order.

- Table 2.1 assumes you are familiar with the information in Chapter 1.

Table 2.1. Checklist of Transition Planning Activities

Step 1: Document the current network configuration:	
	Identify the name, address, operating system, and DECnet version of each node.
	Identify the nodes that are routers, network management stations, load hosts, and name servers.
	Identify any existing DNS Version 1 servers and DECdns Version 2 servers.
	Identify the hardware and software that cannot move to DECnet-Plus.
	Identify the router types and locations.
	Identify the network topology: show all end nodes, level 1 routers, and level 2 routers.
	Identify any multivendor routers.
	Identify the Phase IV applications that use the NICE Protocol.
	Identify any applications that depend on a particular operating system version.
Step 2: Determine your transition strategy:	
	Decide which DECnet Phase V features the network needs: OSI interoperability and/or OSI addressing.
	Determine if the network needs DECnet Phase IV areas.
	Decide which DECnet-Plus software components provide each feature you need, and which optional components to install.
Step 3: Develop a new network configuration:	
	Consider the new network configurations available with DECnet-Plus.
	Decide if you can use the default IDP, or if you need a unique one.
	Determine the address scheme: Phase IV-compatible and/or addresses beyond Phase IV limitations.

	Determine if you will continue to use Phase IV routers.
	For Phase V routers, determine the type of routing to run: Phase IV and/or Phase V routing algorithm.
	Determine which nodes go to DECnet-Plus, which stay at Phase IV.
	Select which nodes will be end systems, routers, and area routers.
Step 4: Plan for your name services and DECdts Time Service:	
	Choose your name services — Local namespace and/or DECdns distributed namespace, or Domain (DNS/BIND) — for the network and for individual nodes.
	If you will use the DECdns distributed namespace, choose the systems that will be name servers and time servers.
	Plan for converting DNS Version 1 namespaces to DECdns Version 2 namespaces.
Step 5: Choose the first end system to migrate:	
	Examine recommendations and evaluate the available nodes.
	Select the first node to migrate.

2.1. Step 1: Document the Current Network Configuration

Planning the migration of a network requires an accurate, up-to-date picture of the network's current state. This picture should be a detailed topological map of the network. Use the information you gather in developing this map as input for decisions you make in the following steps.

For example, information on products that cannot make the transition to DECnet-Plus can help determine the configuration of end systems and routers. Identifying all the routers in your network, and determining each one's type, helps you in Step 2 to decide which routing algorithm to use for each Phase V router. Therefore, the more complete your topological map, the more informed your decisions can be.

The `decnet_migrate` tool provides two commands that, when used together, create a report on the network's configuration:

COLLECT	Collects specified information from specified nodes
REPORT	Creates a report from the data gathered by COLLECT

The tool offers another command, SHOW PATH, that displays the possible paths that node-to-node communication might take through the network, helping to determine what effect the transition has had on the network's communication paths. For complete information about using `decnet_migrate`, see your network management manual.

To complete Step 1:

- Identify all current:
  - Routers
  - Network management stations
  - Load hosts

    - Name servers
- Compile a list of node names and addresses
- Identify the operating system and DECnet version of each end node
- Identify any hardware or software that cannot make the transition to

  DECnet-Plus
- Specify the type and area of each router
- Identify any multivendor routers
- For large or complicated networks, separately document the topology for the level 2 network and for each level 1 area

2.2. Step 2: Determine Your Transition Strategy

DECnet-Plus offers various transition strategies to suit individual customer networks. Most strategies can be divided into two categories, based on your answer to the following question: Does the network need DECnet Phase V areas?

Your network needs DECnet Phase V areas if:

- You want to take advantage of DECnet-Plus features, especially OSI addressing.
- You need connectivity with multivendor OSI networks.
- You want to take advantage of new DECnet-Plus name service access features.

Your network needs to maintain Phase IV areas if:

- You want to maintain systems that cannot migrate to DECnet-Plus.
- You want to migrate slowly over a period of time.

If your network needs Phase IV addressing, your general strategy will be to move the network partially to a DECnet-Plus environment, at least for the near future. You will have a configuration with both Phase IV areas and DECnet Phase V areas.

If your network is ready to use OSI addresses that are beyond the limits of Phase IV and, therefore, not compatible with Phase IV addresses, your strategy will be to move the network entirely to the DECnet-Plus environment with no Phase IV areas and no Phase IV nodes.

2.2.1. If the Network Is Not Moving Entirely to the DECnet-Plus Environment

If the network needs to remain in a DECnet Phase IV environment for at least one area, an appropriate transition strategy might be that the network has DECnet-Plus systems with Phase IV-compatible addresses. These systems will be able to communicate with the remaining Phase IV nodes.

Another option is that you decide to gain experience using DECnet-Plus in one part of your network before migrating the entire network to DECnet-Plus. You can migrate one area to the DECnet-Plus environment, while the network as a whole operates in the transition environment.

In addition, as part of your "partial" transition strategy, you can use the Local namespace to maintain DECnet-Plus node-name and OSI addressing information until it is more convenient to design and move to a distributed namespace.

Install DECnet-Plus software onto each system that is ready to migrate and assign it a Phase IV-compatible address during the configuration procedure. The network transition plan, then, would highlight the order of migration for individual nodes.

2.2.2. If the Network Is Moving Entirely to the DECnet-Plus Environment

If you need at least one DECnet Phase V area, determine whether or not the entire network can move to DECnet-Plus. For example, consider a small, single-area LAN of approximately 20 nodes. If it has no Phase III nodes or other products that cannot migrate, you can migrate it entirely to a DECnet-Plus environment by installing DECnet-Plus software on every end system.

2.3. Step 3: Develop a New Network Configuration

To develop a network configuration, make the following decisions:

- Which routing algorithm will each Phase V router system run?
- Which systems will be end systems and which will be routers?

  For example, existing DECnet Phase IV routing nodes can:

  - Remain at Phase IV and continue to perform routing in a Phase IV area
  - Transition to DECnet-Plus and function as end nodes (assuming you intend to use dedicated routers)
  - Transition to DECnet-Plus for OpenVMS Version 7.1 and continue to function as host-based routers running the Routing Vector protocol
- Will the network use Phase IV-compatible addressing, OSI addressing that is beyond Phase IV limitations, or both?

Your planning activities include:

- Evaluating the possible new network configurations and choosing which to implement
- Considering the addition of multivendor systems to the network
- Planning the network addressing scheme
- Planning routing

  For complete routing configuration information, see the documentation set of your Phase V router.

2.3.1. Planning the New Network Configuration

As you create your new network configuration, review the information in Chapter 1 for:

- Products that will not migrate to DECnet-Plus
- DECnet-Plus incompatibility with Phase III
- Routing products

In addition, to determine the configuration of end systems and routers, use the following information to guide your decisions:

- If you plan to migrate level 1 routers to use link state, you can convert Phase IV PDP-11 and DECnet-Plus for OpenVMS VAX routing nodes to end systems. However, you must remove DECrouter 200 and DECSA routers from the DECnet Phase V area.
- DECnet-Plus supports multivendor OSI-compliant end systems in the network.
- DECnet-Plus supports the following configurations, each with distinct advantages that should be considered for your transition:
  - One LAN with a single, extended-area
  - LANs without routers
  - Multicircuit end systems

  For detailed information about planning routing configurations, seethe documentation set of your Phase V routing product.

2.3.1.1. Single Extended-Area LANs

In the Phase IV environment, LANs can have multiple areas. In the DECnet-Plus environment, however, a LAN can have only one DECnet Phase V area address: an area over 63. This area can have a virtually limitless number of systems, but they all must reside in that single DECnet Phase V area address space. In the transition environment, you can give a LAN multiple areas if the area numbers are within Phase IV limits: 63 or less.

If you plan to migrate a multiple Phase IV-area LAN to the DECnet-Plus environment, during the migration, the DECnet-Plus end systems will be multihomed to a Phase IV-compatible area as well as to an area in the larger DECnet Phase V address space.

2.3.1.2. End-System-Only LAN Configurations

A LAN can consist of end systems only. A DECnet-Plus LAN without routers has virtually no limit on the number of systems it can contain.

The end system to intermediate system (ES-IS) routing exchange protocol (*ISO 9542*) provides the process by which end nodes communicate with routers, or with each other, to exchange configuration information. In a DECnet-Plus end-system-only network, end systems on a LAN use the ES-IS protocol to communicate directly with each other without depending on a router.

DECnet-Plus end systems communicate directly by using a multicast address called "`All End Systems`". All end systems normally listen to this address for `Router Hello` messages, identifying

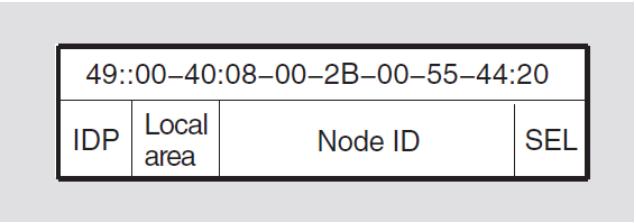
routers to end systems. An end system also uses this multicast address when sending a packet to an end system for which it does not have a cache entry.

When an end system receives a multicast data packet that belongs to itself, it makes a cache entry for the sending end system and sends back an `End System Hello`, which causes the other end system to make an entry in its cache. When it has another packet to send, the end system checks its cache and uses the cached address instead of the multicast address.

In a LAN with only end systems, these systems can have Phase IV-compatible addresses, extended DECnet Phase V addresses, or a mixture of both. You can allow end systems to autoconfigure their OSI addresses, or you can manually assign them Phase IV-compatible addresses.

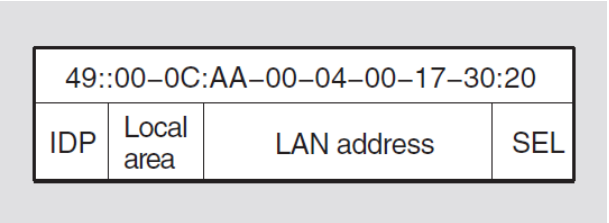
When end systems do not have an assigned Phase IV address, they construct an NSAP for each transport operating over routing. Each NSAP is constructed by concatenating the default local area address (49::00-40), the node ID, and the selector representing the transport (NSP or OSI transport). The Figure 2.1 illustrates a network address, or NSAP, of an autoconfigured end system in an end-system-only network:

Figure 2.1. Autoconfigured End System NSAP



When an end system does have an assigned Phase IV address, it constructs an NSAP for each transport operating over routing by concatenating the local AFI (49), 2 octets containing the area portion of the Phase IV address, a Phase IV-style LAN address (aa-00-04-00-*xx-xx*), and the selector representing the transport. Figure 2.2 illustrates this type of NSAP.

Figure 2.2. Assigned Phase IV Address NSAP



The corresponding Phase IV address is 12.23.

Instead of allowing an end system to autoconfigure its address, you can assign it an OSI address. To manually configure, use the DECnet-Plus configuration utility, `net$configure.com` (for OpenVMS) or `decnetsetup` (for UNIX), or use NCL. For details, see your network management guide.

2.3.1.3. Multicircuit End-System Configurations

A Phase IV end node can have multiple circuits to one or several adjacent nodes, but only one of these circuits can be active at a time. Among Phase IV nodes, only routers have multiple active circuits. DECnet-Plus end systems, on the other hand, can be multicircuited. DECnet-Plus multicircuit end systems can have up to four multiple active circuits to either the same or different subnetworks.

A **subnetwork** is a communications network, within a group of interconnected networks, of OSI systems that use a common addressing format and that forms an autonomous whole. Examples are

X.25 packet switched networks, HDLC data links and *ISO 8802.3* LANs.

These restrictions apply to multicircuit configurations:

- All circuits must be in the same area; the entire set of area addresses for the multicircuit end nodes that correspond to a single network area.
- The circuits must all have the same amount of usable bandwidth.
- All the circuits should provide approximately the same connectivity to other nodes over approximately equivalent paths. Do not use multiple circuits to connect disjointed networks.

Multicircuit end systems do not forward packets to other systems. However, the Routing layer of a DECnet-Plus end system receives information from routers about paths to destinations. The end system gets this information by using the ES-IS protocol to communicate with routers in its area. The routers give the end system information about direct paths to destinations, that is, paths that do not require forwarding the packet. The end system stores the information in a cache and uses it to select an appropriate circuit for sending data to a particular destination. If an end system has no information about a certain destination, it selects a router at random and forwards the data to that router.

Use the following guidelines when you configure multicircuit end systems:

- Configure the circuits to routers with paths that are fairly similar in cost.

  The preferable configuration of end-system circuits is such that information in the cache is on the most efficient paths to a destination. Multicircuit end systems, unlike routers, store no information on path costs. The only decision about paths an end system makes is to choose a direct path, as opposed to an indirect path, to a destination. Multicircuit end systems choose circuits in the following order:

  1. Circuits by which the packet can reach the destination directly, that is, without being forwarded through other systems. The multicircuit end system gets information from routers about direct paths and stores the information in its end-system cache.

     When two or more circuits provide direct paths, the end system selects a circuit on a round-robin (alternating) basis.

  2. Circuits by which the packet can reach the destination indirectly. The end system uses these circuits if it knows of no direct path to the destination.

     Again, when two or more circuits provide indirect paths, the end system selects a circuit on a round-robin (alternating) basis.

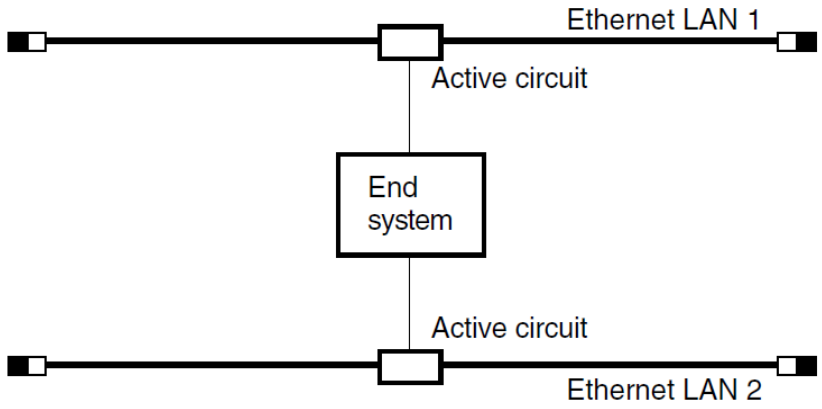
- If you plan to change the configuration of the area, first assess the effect it could have on the operation of the circuits of any multicircuit end systems.

  An end-system cache has a timer on its circuit information. When the timer expires, the end system gets new information about paths. If the configuration has changed, the path information may not be similar in cost any more.

DECnet-Plus for OpenVMS and DECnet/OSI for UNIX end systems with multiple active circuits provide redundancy and increased data throughput. An end system with multiple circuits to an Ethernet LAN, for example, provides a guarantee that if one circuit fails, this system still has access to the LAN. There is no gain in data throughput, however, since both circuits go to the same LAN.

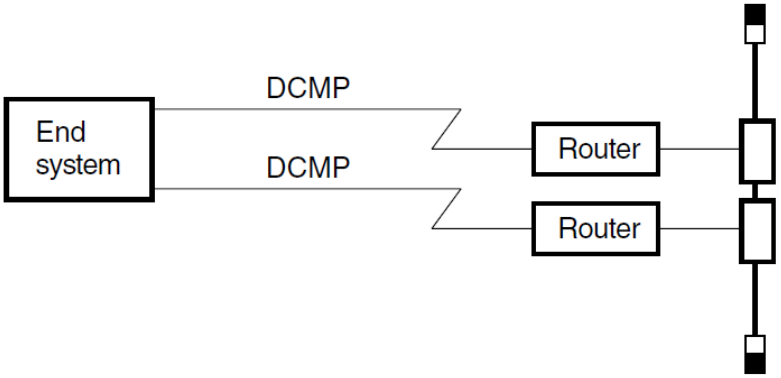
You can also configure a multiple-active-circuit end system with its circuits to different LANs. Figure 2.3 shows an end system with circuits to two different Ethernet LANs. This configuration is useful where reliability is critical such as in a network that supports banking applications, for example. This configuration also provides for increased data throughput, since circuits to two different LANs can carry double the number of packets.

Figure 2.3. Multicircuit End System with Circuits to Two Ethernet LANs



OpenVMS: Figure 2.4 shows a configuration in which an end system has two active Data Communications Message Protocol (DCMP) circuits to different routers on a LAN. This configuration could provide redundancy in a situation where, for example, there is concern about disturbances on one of the telephone lines.

Figure 2.4. Multicircuit End System with Two DCMP Circuits



Though you can configure circuits to either the same or a different subnetwork, you can enable the Phase IV address of only one circuit per LAN. To configure a multicircuit end system with a Phase IV address and one or more circuits to the same LAN, take the following steps:

1. If you have not already done so, install the Ethernet hardware necessary for each of the system's connections to the LAN.

2. During the DECnet-Plus configuration process, assign the system a Phase IV address. Also, choose the option that allows the system to autoconfigure its network address.

3. After configuration, on all but one of the Ethernet circuits, set the Routing Circuit entity characteristic `enable phase IV address` to false. The one circuit with this attribute set to true has its Ethernet address set to a Phase IV Ethernet address. All other circuits retain the address of their Ethernet controller.

This example sets the `enable phase IV address` routing characteristic to FALSE on the routing circuit `csmacd-1` of the local node:

```
ncl> set routing circuit csmacd-1 enable phase iv address=false
```

You can set the Routing characteristic `enable phase IV address` in two ways:

- Answer the installation prompts appropriately.
- Edit the Routing layer initialization script and add the commands to set the characteristic. When you next reboot the system, the initialization process runs the script and enables the Phase IV address on only one circuit.

  For for OpenVMS, the initialization script is:

  ```
  sys$manager:net$routing_startup.ncl
  ```

  For UNIX, the initialization script is:

  ```
  /var/dna/scripts/start_routing.ncl
  ```

2.3.2. Including Multivendor Systems in the Network

DECnet-Plus software allows OSI-compliant systems from other vendors to participate as end systems in the DECnet Phase V network. Use the following guidelines when you add other vendors' OSI-compliant systems to your network:

- You can include only OSI-compliant systems.
- You can include OSI end systems in a DECnet Phase V area (or network). You can also include them in a Phase IV area (or network operating in a transition environment) if you can configure their addresses as Phase IV-compatible addresses.
- Phase V routers treat the multivendor OSI end system as they do any DECnet-Plus end system.
- You must have DECnet-Plus paths to the multivendor OSI end systems. This means that the network is configured so every possible connection path to the OSI end system has only Phase V routers, not Phase IV routers.
- Do not register multivendor OSI end systems in the DECdns namespace. Only NA applications and NA network management use node-name objects in the namespace; OSI applications use their own private naming databases. Register the multivendor OSI end systems in those OSI databases as directed by your OSI application documentation.
- To include a multivendor OSI end system in the network, check that the system meets one of the following conditions:
  - The end system uses the ES-IS protocol and the end system's address can be configured. See the other vendor's documentation for this information.
  - The end system does not use the ES-IS protocol but resides on a LAN for which there is an OSI router and the NSAP address can be configured.
  - The end system does not use the ES-IS Protocol and you cannot configure its NSAP address; but you can attach the OSI-compliant end system to a level 2 Phase V router system. This router must be running link state if the OSI-compliant end system's address is not Phase IV-compatible.

- You attach the OSI-compliant end system through an X.25 dynamically assigned circuit.

  On either the router or the end system, you create a `routing circuit reachable address` entity for the OSI-compliant end system.

2.3.3. Using the Inactive Network Layer Protocol

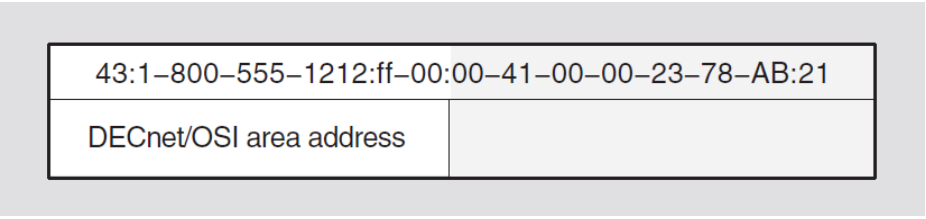
DECnet-Plus provides support for the inactive Network layer protocol specified in *ISO 8473*. The inactive subset bypasses the Network layer, running transport directly over a data link connected to a LAN. You can use this configuration when the source and destination end systems are on the same LAN.

Consider these guidelines before using the inactive Network layer protocol:

- Only one transport module can make use of the inactive subset. By default, this is OSI transport, as specified by the preset routing characteristic `inactive selector`.
- All end systems using the inactive subset should specify the same `inactive area address`.

To configure the inactive subset on a circuit, use NCL to set the circuit attribute `inactive area address`. The area address portion of the NSAP address is placed in this set, as shown in Figure 2.5.

Figure 2.5. Area Address Portion of the NSAP Address



This command configures the inactive subset on `circuit-1`:

```
ncl> set routing circuit circuit-1 inactive area address {49::ff-00}
```

You can assign only one inactive area address to a circuit.

During transmission, when a service data unit (SDU) is passed to an end system's Network layer, transport provides a destination NSAP. If the area address portion of the destination NSAP matches the inactive area address assigned to the circuit, then routing uses the inactive Network layer protocol and passes the SDU directly to the Data Link layer, specifying the ID portion of the destination NSAP as the destination data link address.

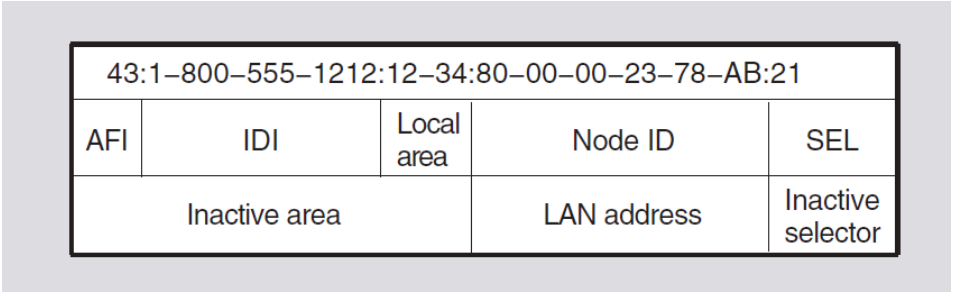
If an end system's Network layer receives a protocol data unit (PDU) with the inactive Network layer protocol header on a circuit that has the inactive area address set, then routing passes the SDU up to transport.

In addition, routing calculates the source and destination NSAPs as follows:

- **Source NSAP**

  Figure 2.6 illustrates a source NSAP.

Figure 2.6. Source NSAP



- **Destination NSAP**

  The numerically lowest NSAP in the set of NSAPs assigned to the port for which the SDU is bound.

If an end system's Network layer receives a PDU with an inactive Network layer protocol header on a circuit that does not have the `inactive area address` set, then the PDU is dropped.

2.3.4. Planning Addressing

When planning your network's addressing scheme, determine the need for unique IDP and DECnet Phase V addresses, which, in turn, depends on the general transition strategy you chose in Step 2.

2.3.4.1. Do You Need a Unique IDP for Your Network?

DECnet-Plus software includes a default IDP, which has the local AFI of 49 and a null IDI. The local AFI and null IDI means that the IDP is intended for use within a private network. Therefore, the local AFI is useful only in a network not connected to any external networks.

As network manager, consider whether the default IDP is sufficient for your network. If you plan to connect your network to other networks, you must obtain a unique IDP. Consider obtaining a unique IDP if there is a possibility that, in the future, you might connect your network to other networks. Otherwise, the default IDP is sufficient.

For information about obtaining a unique IDP for your network, see Section 4.7. If you change your network's IDP, use the `decnet_register` tool (see your network management guide).

2.3.4.2. Do You Need Extended OSI Addresses?

The general transition strategy you planned in Step 2 determines your decision about the need for OSI addressing that goes beyond Phase IV address limitations:

- If **all** of your network is moving to the DECnet-Plus environment, you have already decided to use OSI addressing. This strategy requires that routers run the link state protocol. See Section 2.3.4.3 for addressing guidelines.

- If **none** of your network is moving to the DECnet-Plus environment, then you have already decided to use PhaseIV-compatible addressing for all systems. With this strategy, you can use the default IDP that comes with the DECnet-Plus software. In this case, you can ignore the considerations in Section 2.3.4.3 until you are ready to move the network to OSI addressing.

- If **part** of your network is moving to the DECnet-Plus environment, then you will make address decisions on an area-by-area basis. Whether an area will be a Phase IV or DECnet Phase V area

affects the decisions you make as to the area's configuration, addressing, and routing algorithm. See Section 2.3.4.3 for addressing considerations.

2.3.4.3. Considerations for a Network with DECnet Phase V Areas

A network with one or more DECnet Phase V areas requires well-planned addresses because:

- There are differences between Phase IV and OSI addresses, and communication between Phase IV nodes and DECnet-Plus systems depends on coordinating those addresses (see Section 1.3.1).
- Using extended OSI addresses requires that routers run linkstate (see Section 2.3.5).

Also use the following guidelines when planning addresses:

- If an area has Phase IV nodes, you must assign Phase IV addresses to all the Phase V routers in the area.
- A network area can use OSI addressing only when all systems are DECnet-Plus systems and all routers run link state *or* when Phase IV nodes communicate with a small number of DECnet-Plus systems that have Phase IV-compatible addresses.
- For a DECnet-Plus system to communicate with another DECnet-Plus system with an extended address, the path between the systems must be a DECnet-Plus path. A DECnet-Plus path is a routing path on which all routers are running link state.
- In a OSI address, the area address equals the IDP +PreDSP + LOC-AREA fields. Phase IV nodes do not recognize the concept of an IDP. Therefore, they cannot communicate outside of their own network. A Phase IV node cannot connect to any node in another network that has a different IDP, not even another Phase IV node.
- When the network area is ready to use extended OSI addressing, assign these addresses to the Phase V routers, which must be running the link state protocol. Assigning OSI addresses to the Phase V routers makes it possible for end systems in the area to autoconfigure their node addresses.
- If your network has a unique IDP, you must specify that IDP as part of the node address at either of the following times:
  - During the configuration of every router, if you plan that the end nodes will autoconfigure their addresses
  - During the configuration of every system, if you plan that the end nodes will not autoconfigure their addresses

2.3.5. Planning Routing

When you plan routing configurations, use the following guidelines:

- Phase V router products can support either the Phase IV routing vector protocol or the DECnet Phase V link state protocol on level 1 and level 2.
- All level 1 routers in an area must run the same routing protocol.

  For example, if one level 1 router runs routing vector which is a DECSA router that cannot migrate to DECnet-Plus, then all routers in the area must run routing vector.
- To use OSI addresses, all level 1 routers must be running link state.

- For end systems to autoconfigure their addresses, all level 1 routers must run link state. Also, during configuration, you must have assigned DECnet Phase V area addresses to the level 1 routers.
- You must have DECnet-Plus routing paths:
  - Among all DECnet-Plus name servers
  - Between each DECnet-Plus system and OSI end system with which it communicates, if they have non-Phase-IV addresses
  - Between two OSI end systems that need to communicate, if they have non-Phase-IV addresses.
- Routers in the level 2 network can run different routing algorithms. To allow for a level 2 router running routing vector to communicate with a level 2 router running link state, the two routers must meet the following requirements:
  - A router running routing vector must be connected directly to a router running link state.
  - A router running link state must have an interphase link entry in its reachable-address table for the router running routing vector.

    See your network management guide for information about interphase routing and how to use the `decnet_migrate` tool's `create ipl_initialization_file` command that helps you set up interphase links.
- You can use NCL to set the routing algorithm. For example, the following command sets the routing algorithm to link state on adjacent routing node `.ultra.router:`

```
ncl> set node .ultra.router routing manual L1 algorithm link state
 algorithm
```

- When you switch the routing algorithm, for example, in an area where all systems will have OSI addresses, all routers must change algorithms at the same time or you lose connectivity in the area.

For detailed information about DECnet-Plus routing, see the *Routing Overview* guide in the Phase V router product's documentation set.

2.4. Step 4: Plan for Your Name Services and the DECdts Time Service

The basic planning tasks related to the name services and DECdts are as follows:

- Decide to use the Local namespace, the DECdns distributed namespace, DNS/BIND, or several namespaces to maintain DECnet node name and addressing information. A distributed namespace uses both DECdns clerk and server software.

  Namespace planning for a distributed namespace includes designing the directory structure, planning an access control policy, and deciding on the contents of directories. For more information about planning a distributed namespace, read Chapters 6, 7, 8, 9, and 10.

  Also, if you are considering having multiple namespaces, read Section 2.4.2 and Section 7.1.
- If you choose to use a distributed namespace, plan which nodes become name servers and time servers.

If your network uses a distributed namespace, every DECnet-Plus system in the network is a DECdns clerk and a DECdts clerk, but not every system should be a name server or a time server. Select your servers carefully based on the guidelines in Section 2.4.1.

2.4.1. Choosing DECdns and DECdts Servers

You do not need to install a DECdns server if you use a Local namespace on all DECnet-Plus systems. All references to DECdns servers apply to those servers running on UNIX or OpenVMS VAX systems. You need to configure one or more DECdns server systems if you intend to use a distributed namespace on one node or selected nodes. Use the following guidelines when choosing which DECnet-Plus systems will be DECdns and DECdts servers:

- Overall, VSI recommends you choose more time servers than name servers; therefore, the name server nodes can be a subset of the time server nodes. Two name servers and two time servers for every 1,000 nodes are usually sufficient. See your installation and configuration guides for detailed server configuration guidelines in both LAN and WAN environments.
- A DECdns server and a DECdts server must maintain at least one Phase IV-compatible address until no Phase IV nodes exist on the network that need to communicate with these servers. Lack of Phase IV-compatible addresses prevent the servers from communicating with Phase IV nodes.
- DECdns servers must have the NSP transport protocol enabled and should have the OSI transport protocol enabled. For DECnet Phase V networks using a distributed namespace, every system is a DECdns clerk and systems running either protocol must be able to communicate with any name server. If DECdns servers already exist on Phase IV systems, this requirement also ensures that they can talk to your DECnet-Plus DECdns servers in the network.

2.4.2. Creating Multiple Namespaces

VSI recommends that you avoid using multiple namespaces. However, if your organization or network has special requirements that justify the use of multiple namespaces, you can create them and they can coexist without harm. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for details about the implications of using multiple namespaces.

When multiple namespaces do exist in a network, they are separate and do not share information. You cannot replicate data across namespaces. Users can refer to a name in a namespace other than their own by including the namespace nickname in the full name reference.

If you want namespaces to interoperate, one trade-off is increased administrative overhead to keep track of and maintain entries in each namespace.

- Using multiple namespaces greatly increases the complexity of managing node object entries and soft links.
- Setting up access control to allow users to read entries in both namespaces is also complicated.

The easiest way to avoid the complexity is to limit node object entries and soft links to a single namespace, and use other namespaces for other purposes, such as testing, or building and using client applications.

If you need multiple namespaces in your network, and you want more than one namespace to contain node object entries and soft links, take the following steps after you install and configure new DECnet-Plus systems to ensure that they can communicate across namespaces:

- Decide which nodes each namespace will contain. Ensure that each node is registered in one, and only one, namespace.
- Create the namespaces you have planned. DECdns namespace creation involves configuring at least one server in each namespace.
- Run `decnet_register` once in each DECdns namespace to create the required DECnet-Plus directories. For instructions on running `decnet_register`, see your network management guide.
- Use the `decnet_register export` command to extract the node information from the Phase IV database. Edit the Export/Import file to correctly format the node information for the namespace you intend to use on the node. See your network management guide for more information.
- Use the `decnet_register import` command on each node to import the edited node information contained in the Export/Import file into the namespace on the node.
- Inform users that they must include a namespace nickname in anode full name when they refer to a node in a namespace other than their own.
- You can also create a soft link in your namespace for each node in another namespace. You can create this soft link in any directory that you consider appropriate.

  For example, a user in the IAF namespace could create a synonym soft link named `IAF:.sample.Node01` that points to a node whose full name is `ABC:.sales.Node01` in the ABC namespace. Then, users in the IAF namespace could refer to the node by the name Node01 and DECnet-Plus would translate the soft link to the node's full name. Make sure you establish adequate cross-namespace access control entries. Keep in mind that cross-namespace node synonym soft links must be manually updated if there is a change in the name of the node.

2.4.3. Preparing a DNS Version 1 Namespace for Use By DECnet-Plus

If you are already using a namespace created with Version 1 of the VAX Distributed Name Service (DNS) running on DECnet-VAX Phase IV, you can continue to use this namespace when you upgrade your networking software to DECnet-Plus. However, because of differences in the way DNS Version 1 and DECdns Version 2 handle access control, you must follow several steps to prepare your DNS Version 1 namespace for use by DECnet-Plus. These differences affect the way in which DNS Version 1 and DECdns Version 2 interpret principal specifications in access control entries (ACEs).

In DNS Version 1, servers recognize principals only of the form *`nodename::username`*. In DECdns Version 2, servers recognize principals primarily of the form *`nodename.username`*. To make it possible for the DECdns Version 2 clerks and servers that you will create later to interpret and process the DNS Version 1-style ACEs already in use in the namespace, create a backtranslation directory (`.DNA_BackTranslation`) and node synonym directory (`.DNA_Node_Synonym`) in the namespace's root directory. Then, populate these directories by registering all the nodes participating in the Version 1 namespace. See your installation guides for complete information on how to prepare a DNS Version 1 namespace for use by DECnet-Plus.

You need to perform these operations only once to prepare a DNS Version1 namespace for use with DECnet-Plus. Once the node synonym and backtranslation directories are populated, you can configure new DECdns clerks and servers, or convert existing DNS Version 1 servers to DECdns Version 2 format in the normal manner. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for complete information on how to configure a DECdns server into an existing namespace and how to convert a DNS Version 1 clearinghouse to DECdns Version 2 format.

Note

If you do intend to convert any of your existing DNS Version 1 clearinghouses to DECdns Version 2 format, VSI strongly recommends that you do *not* configure DECnet-Plus on any of your existing DNS Version 1 server nodes until after you have prepared your DNS Version 1 namespace for use by DECnet-Plus.

If you are using a namespace created with DNS Version 1, you are already familiar with many of the distributed naming and namespace planning concepts described in Chapter 6 through Chapter 10 of this guide. However, be sure to read all the DECdns-related sections in this guide to better understand the differences between DNS Version 1 and DECdns Version 2.

2.5. Step 5: Choose the First End System to Migrate

It is important to carefully identify the first node to migrate because, in a sense, it is from this point that you will move most of the network to DECnet Phase V. You will use this first system to manage other systems during migration and, probably, to set up the DECdns name service and initialize the namespace, if you are not using a Local namespace. The following are recommendations for choosing the first node to migrate:

- You will use this node to manage other nodes during transition, so ensure that it has access to other systems you want to manage and, possibly, migrate.
- If you want to use DECdns and it does not exist on the network, make the first node you migrate a DECdns server and create a distributed namespace. See Section 7.6 for guidelines on choosing a name server node.
- You can configure this node as a DECdns clerk and use an existing Version 1 name server with the following conditions:
  - The server node must be adjacent to this first DECnet-Plus system.

    On a LAN, **adjacent** means on the same LAN as the existing server; in a WAN environment, it means no other systems, except the router, are configured between the Phase IV name server and the DECnet-Plus system.
  - All DECnet-Plus systems must have DECnet-Plus paths to the Version 1 name server.

For details, see the appendix on DECdns version interoperability in the *VSI DECnet-Plus for OpenVMS DECdns Management Guide*.

Chapter 3. Performing Transition Tasks

This chapter discusses your immediate transition tasks and the tools you need to complete these tasks.

You will perform some of these tasks during DECnet-Plus configuration, some while running the transition tools immediately after configuration, and some at other times. For information about how to perform these tasks, see your installation and network management guides.

3.1. Tools: Network Management, Node-Name Management, and Transition

DECnet-Plus software includes the following tools to help you during transition. After transition, some of these tools are used for ongoing network management support and node-name management.

- `sys$system:net$mgmt.exe` (for OpenVMS) or

  `/usr/sbin/dna_mgmt` (for UNIX)

  A graphical user interface provided to manage Phase V nodes. You can enable NCL logging if you wish to see any NCL commands performed on your behalf by this Motif-based application.

- `decnet_register`

  Use the `decnet_register` tool to manage the node names and addressing information contained in both Local namespaces and DECdns distributed namespaces. Use `decnet_register` to:

  - Register node names, node synonyms, and addresses in your namespaces.
  - Add addresses to a node registration.
  - Remove addresses from a node registration.
  - Modify a node registration's synonym or addresses.
  - Display a node registration and verify its internal consistency.
  - Export node registration information from a namespace to a text file.
  - Import node registration information from a text file into a namespace.
  - Update a remote node's registered address information with information obtained from the node itself.
  - Rename a registered node in a namespace.
  - Deregister a node from a namespace.

  You can also use `decnet_register manage` command to create and manage the directories associated with the DECdns namespace.

- `decnet_migrate`

  Use this tool to:

  - Check information on the network's current configuration
  - Set up routing between Phase IV and DECnet Phase V areas
  - Convert NCP command files to NCL command files and upgrade either DCL command files or UNIX scripts that contain NCP commands
  - Use the Language Sensitive Editor (LSE) to create and edit NCL command files (for OpenVMS)
  - Learn NCL syntax and NCP/NCL equivalents (for OpenVMS)

  For complete information about using `decnet_migrate`, see your network management guide.

- `sys$update:net$convert_database` (for OpenVMS)

  Upgrades the object database for VSI DECnet-Plus for OpenVMS and converts the Phase IV MOP database to the DECnet-Plus MOP client database. During `net$configure` you are asked if you want to convert Phase IV databases. Answering `yes` runs `sys$update:net$convert_database`.

  Note

  VSI DECnet-Plus for OpenVMS uses the proxy file created with DECnet for OpenVMS Phase IV; therefore, no update is needed.

- `/usr/field/objtoncl` (for UNIX)

  Converts the Phase IV object database to an NCL script (see Section 3.3.1).

- `/usr/field/proxytoncl` (for UNIX)

  Converts the Phase IV proxy file to an NCL script (see Section 3.3.2).

- `/usr/field/update_mopdb` (for UNIX)

  Converts the Phase IV MOP database to the DECnet-Plus MOP client database (see Section 3.3.3).

3.2. Immediate Tasks

Migrating a network ultimately means migrating individual systems. Follow your transition plan to decide which procedures you will use, and in what order. No one way to migrate is correct for all networks. However, the following steps always apply:

1. Determine if the default IDP is appropriate for your network, and, if not, obtain a unique IDP. See Section 3.2.1.
2. Decide if you will use a Local namespace, a DECdns distributed namespace, or both.
   - If you use a DECdns namespace, see Section 3.2.3 for information about migrating the first end node. See Section 3.2.4 for information about migrating subsequent end systems.

- If you use a Local namespace, see Section 3.2.2 for instructions on how to create a Local namespace for each node you configure.
- If you use both, refer to the *DECnet-Plus for OpenVMS Applications Installation and Advanced Configuration Guide* for detailed information.

3. For VSI DECnet-Plus for OpenVMS, migrate OpenVMS cluster nodes. See Section 3.2.5.
4. Migrate local area networks. See Section 3.2.6.
5. Configure DECnet-Plus end systems in a multivendor OSI network, if your network is multivendor. See Section 3.2.7.

Note

The DECnet-Plus software installation procedure differs considerably from Phase IV installations. For complete instructions on installing and configuring, see your installation guides.

3.2.1. IDP Planning

To configure the first DECnet-Plus system, you must know its new OSI address, including your network's IDP and, in some cases, the preDSP as well.

Determine if you can use the default IDP. If you cannot, before you start the transition to DECnet-Plus, apply for a unique IDP. For information about the format of DECnet Phase V addresses, see Chapter 4; for details about IDPs, see Section 4.1, and for instructions on how to get a unique IDP for your network, see Section 4.7.

3.2.2. Using a Local Namespace: Migrating the Network

With a Local namespace, migrating consists of installing and configuring DECnet-Plus. To create a Local namespace, take the following steps:

1. Install and configure your DECnet-Plus software.
   a. When the configuration procedure prompts you for your node full name, type `local:`.*nodename*. The reserved nickname for the Local namespace is `local`.
   b. The configuration procedure automatically creates the local name file.

      If neither data file exists or is readable, the procedure creates the local name file with information for the node in it.

      For **OpenVMS**, if the DECnet Phase IV node data file `sys $system:netnode_remote.dat` exists and is readable, then answer YES to the question, `Do you want to convert a Phase IV database?` and the procedure converts it to the local name file.

      The procedure will also use the `decnet_register export` and `import` commands to extract the node information from the Phase IV database and to import it into the Local namespace.
2. You are ready to use Go to Section 3.2.6.

3.2.3. Using a Distributed Namespace: Migrating the First End Node

Migrating the first end node includes several tasks that affect the management and migration of the entire network. These tasks include creating the namespace, configuring the first name server, and setting up access control. Therefore, it is important that the network manager carry out or oversee the migration of the first end node.

To help you plan, this section outlines the steps required to install and configure the first DECnet-Plus system in the network. Section 3.2 is meant to serve as a "road map" to familiarize you with the transition steps you need to take. The goal is to prepare you for installation, configuration, and transition as documented in the installation guides.

DECdns server software is not available for OpenVMS Alpha systems, therefore, all references to DECdns servers apply to those servers running on UNIX or OpenVMS VAX systems.

Choose the scenario that applies to your transition:

- If the network does not have a DECdns namespace, create a new DECdns namespace when you migrate the first system (see Section 3.2.3.1).
- If the network has a DNS Version 1 namespace (OpenVMS only), you join this existing namespace when you migrate the first system (see Section 3.2.3.2).
- If your network already has a DECdns Version 2 namespace, follow the instructions in Section 3.2.4. Each node you install and configure is a subsequent node. If you are considering multiple namespaces, see Section 2.4.2 and Section 7.1 for restrictions and guidelines.

3.2.3.1. If the Network Does Not Have a Namespace

If you choose to install and configure the first DECnet-Plus node in a network with no existing namespace, follow the steps in this section. All references to DECdns servers apply to those servers running on UNIX or OpenVMS VAX systems.

1. On the system to undergo the transition, ensure that the permanent node database is up to date. Back it up. This database file is:

   `sys$system:netnode_remote.dat` (OpenVMS only)

   `/usr/lib/dnet/nodes_p` (UNIX only)

2. Replace any unsupported hardware.

3. Install your DECnet-Plus software. As part of the installation and configuration, install and configure the DECdns server software.

   On an OpenVMS VAX system, perform the advanced configuration `@sys$manager:net$configure advanced`.

   On a UNIX system, perform the advanced configuration, `decnetsetup advanced`.

   During the configuration procedure:

   a. It automatically configures the system as a DECdts server if a time synchronization service is not already running in your network.

b. You create a namespace in one of the following ways:

- If you specified `local` during network configuration, a local namespace is created automatically.
- If you specified `decdns` during network configuration, DNS$CONFIGURE is called to create a new DECdns namespace.
- If you choose DECdns, you also need to do the following after the configuration procedure exits:
  - Use the `decnet_register manage` command to create the namespace directories, create and enable a clearinghouse, and create the root directory of the namespace in that clearinghouse.
  - Use the `decnet_register manage` command to initialize the namespace for use by DECnet-Plus. This initialization creates the node synonym directory (`.DNA_NodeSynonym` by default), the backtranslation directory (`.DNA_BackTranslation` by default), and the global time servers directory (`.DTSS_GlobalTimeServers` by default).
  - Use the `decnet_register manage` command to create the `.DNA_Registrar` access control group. The group is initially empty; use `decnet_register` later to add members.
  - Rerun the configuration procedure for the system to configure it as a DECdns server and a DECdns clerk.

4. Implement the namespace wide access control policy you have decided to use. VSI highly recommends this practice, especially in large networks. Section 7.4 describes how to plan an access control policy.

   To implement this policy, use the `decnet_register manage` command to add members to `.DNS_Admin`, the namespace administrator group, and add access to the root directory.

5. Use the `decnet_register manage` command to modify default access control for the node directories.

6. While you are using `decnet_register`, make sure that the account under which you will run your startup procedure has at least write access to the newly created clearinghouse and root directory.

7. Your DECnet-Plus node cannot communicate with a Phase IV node until the namespace has an object and soft links for that Phase IV node. Therefore, if your network is to include Phase IV nodes, register those nodes in the namespace now. Follow these steps:

   a. Make sure that on the newly installed DECnet-Plus node, you have the up-to-date copy of:

      `sys$system:netnode_remote.dat` (OpenVMS only)

      `/usr/lib/dnet/nodes_p` (UNIX only)

   b. Use `decnet_register` to populate the namespace with Phase IV node objects and soft links. See your network management guide.

      Do this before populating the namespace with any other node names.

Use the `decnet_register manage` command to create the `.DNA_Node` directory at this time. The node information files are initially set up to register the Phase IV nodes in this directory. If you want, you can edit those files to change the name of the directory into which the nodes are to be registered.

c. Use `decnet_register` to:

- Fill `.DNA_Node` with objects for Phase IV node names.
- Fill the `.DNA_NodeSynonym` directory with soft links pointing from each node's Phase IV-style synonym to its full object name.
- Fill the `.DNA_BackTranslation` area subdirectories with soft links pointing from each node address to its full object name. Each area subdirectory is populated with the node IDs of systems in that area.

After you install and configure the first DECnet-Plus system, install subsequent systems according to the transition schedule you have planned. See Section 3.3 for additional transition tasks.

3.2.3.2. If the Network Has a DNS Version 1 Namespace

If you are already using a namespace created with Version 1 of the VAX Distributed Name Service (DNS) running on DECnet-VAX Phase IV, you can continue to use this namespace when you upgrade your networking software to DECnet-Plus. However, because of differences in the way that DNS Version 1 and DECdns Version 2 handle access control, you must perform several steps to prepare your DNS Version 1 namespace for use by DECnet-Plus. These differences affect the way in which DNS Version 1 and DECdns Version 2 interpret principal specifications in access control entries (ACEs). For more information, refer to Section 2.4.3.

3.2.4. Using a Distributed Namespace: Migrating Subsequent End Nodes

The transition of any end node other than the first end node consists of installing and configuring the DECnet-Plus software. System managers can therefore migrate subsequent end nodes, as long as the network manager is available to supply the information required to answer the prompts during installation and configuration.

Installers might need help registering nodes in the namespace. Near the end of the configuration, the procedure attempts to automatically register the node in the namespace. Registration could fail, depending on the type of installation being performed and the protection level of the namespace. If it does fail, you get a message stating that you will have to manually register this node in the namespace.

To simplify node registration, you can implement one of the following strategies before installing subsequent nodes:

- Manually register each subsequent system in the namespace before it is installed and configured. Use `decnet_register`. See your network management guide.
- Enable node autoregistration for all directories into which users will register systems so that, during node configuration, each system can be registered automatically. Table 3.1 and Table 3.2 summarize how automatic registration and manual registration apply to:
  - New systems

- Systems making the transition from Phase IV to DECnet-Plus
- Systems downgraded from DECnet-Plus to Phase IV

3.2.4.1. Registering a New System

Table 3.1 describes how you use `decnet_register` to register a new system in the namespace after installing DECnet-Plus or Phase IV software on the system, assuming that the system is not currently registered in the namespace.

Table 3.1. Registering a New System

Phase of the New System	**Autoregistration Allowed**[1]		**Autoregistration Disallowed**[2]	
	Automatic Registration	**Manual Registration**	**Automatic Registration**	**Manual Registration**
DECnet-Plus System	Yes	Not required	No	Register the system as a DECnet-Plus system.
Phase IV System	No	Register the system as a Phase IV system.	No	Register the system as a Phase IV system.

[1]If a directory allows *autoregistration*, all systems and users have read/write access to that directory, allowing anyone to register nodes in it.
[2]If a directory disallows *autoregistration*, only users who are explicitly granted read/write access to that directory can register nodes in it.

3.2.4.2. Registering a System You Migrated from Phase IV to DECnet-Plus

Table 3.2 describes how you use `decnet_register` to reregister a Phase IV system currently registered in the namespace after migrating the system to DECnet-Plus.

Table 3.2. Registering a System You Migrate from Phase IV to DECnet-Plus

DECnet Phase V Name and Address	**Autoregistration Allowed**[1]		**Autoregistration Disallowed**[2]	
	Automatic Registration	**Manual Registration**	**Automatic Registration**	**Manual Registration**
Same Name Same Address	Yes	Not required	Yes	Not required
Same Name Different Address	Yes	Not required	No	Before booting the system for the first time, reregister the system as a DECnet-Plus system.
Different Name Same Address	Yes[3]	Before booting the system for the first time, change the Phase IV name in	No	Before booting the system for the first time, change the Phase IV name in

DECnet Phase V Name and Address	Autoregistration Allowed[1]		Autoregistration Disallowed[2]	
	Automatic Registration	**Manual Registration**	**Automatic Registration**	**Manual Registration**
		the namespace to the new DECnet-Plus name.		the namespace to the new DECnet-Plus name.
Different Name Different Address	Yes[3]	Before booting the system for the first time, change the Phase IV name in the namespace to the new DECnet-Plus name.	No	Before booting the system for the first time, change the Phase IV name in the namespace to the new DECnet-Plus name and then reregister the system as a DECnet-Plus system.

[1]If a directory allows *autoregistration*, all systems and users have write access to that directory, allowing anyone to register nodes in it.

[2]If a directory disallows *autoregistration*, only users who are explicitly granted write access to that directory can register nodes in it.

[3]Autoregistration is not recommended in this case because the system creates a new node object that does not include any special information included in the old Phase IV node object, and you must delete the old Phase IV node object.

3.2.4.3. Registering a System You Changed from DECnet-Plus to Phase IV

Use `decnet_register` to reregister a system in the namespace after changing the system from DECnet-Plus to Phase IV. Phase IV systems cannot autoregister in the namespace. See your network management book for further information.

3.2.5. Migrating OpenVMS Cluster Nodes (OpenVMS Only)

Before you start to transition any system in a OpenVMS cluster, you need to gather the following information:

- The DECnet Phase V node name and address.

  When a node is configured as a cluster member, the DECnet Phase V name and address are used to set the `sysgen` parameters `scsnode` and `scssystemid`. With Phase IV, the DECnet name and DECnet Phase V address you enter during cluster configuration has to be the same as the name and address you enter during the DECnet for OpenVMS configuration procedure. With DECnet-Plus, `scsnode` can be different from the DECnet node name and the `scssystemid` can be different from the DECnet Phase V address.

- If the OpenVMS cluster system you are migrating participates in the cluster alias, you need the following information:

  - Is the cluster making the transition gradually so that it is operating with a mixture of Phase IV and DECnet-Plus systems, or all at once?

  - If gradually, should the cluster alias be enabled on this system?

Note

Do not enable the cluster alias on any DECnet-Plus OpenVMS cluster system until you have migrated all the systems participating in the alias.

To migrate an OpenVMS cluster node, when `net$configure.com` prompts you for the system's node name and address, enter the system's Phase IV node name and Phase IV address. For more information about migrating OpenVMS cluster nodes to DECnet-Plus, refer to *DECnet-Plus for OpenVMS Installation and Basic Configuration Manual* or *DECnet-Plus for OpenVMS Applications Installation and Advanced Configuration Guide*.

3.2.6. Migrating Local Area Networks

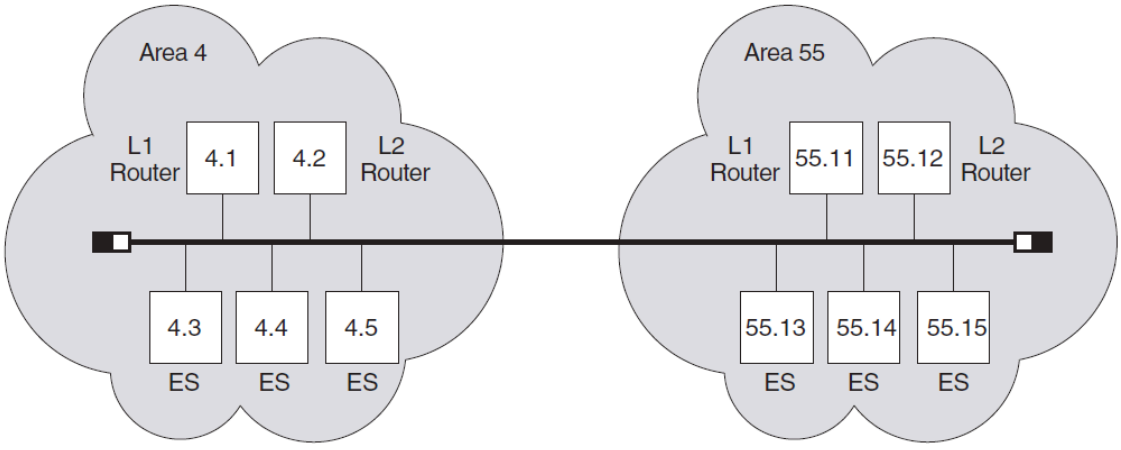
Migrating your LAN requires you to pay close attention to your routers. During Step 1 of your planning activities, you identified the routers in your network and specified their type, location, and function. Now you need that information.

This section shows the steps of a typical transition of a two-area LAN, highlighting:

- How DECnet-Plus software is introduced to the LAN
- How routing works with a mixture of Phase IV and DECnet-Plus systems
- How all the routers are gradually configured to use the link state protocol
- How the network eventually makes a complete transition to DECnet-Plus

This section, without specific NCL commands, describes the LAN migration in a general way. See your router documentation for detailed instructions. Figure 3.1 shows an example of a LAN to be migrated to DECnet-Plus.

Figure 3.1. Two-Area Phase IV LAN to Be Migrated to DECnet-Plus



Begin the transition with the following steps:

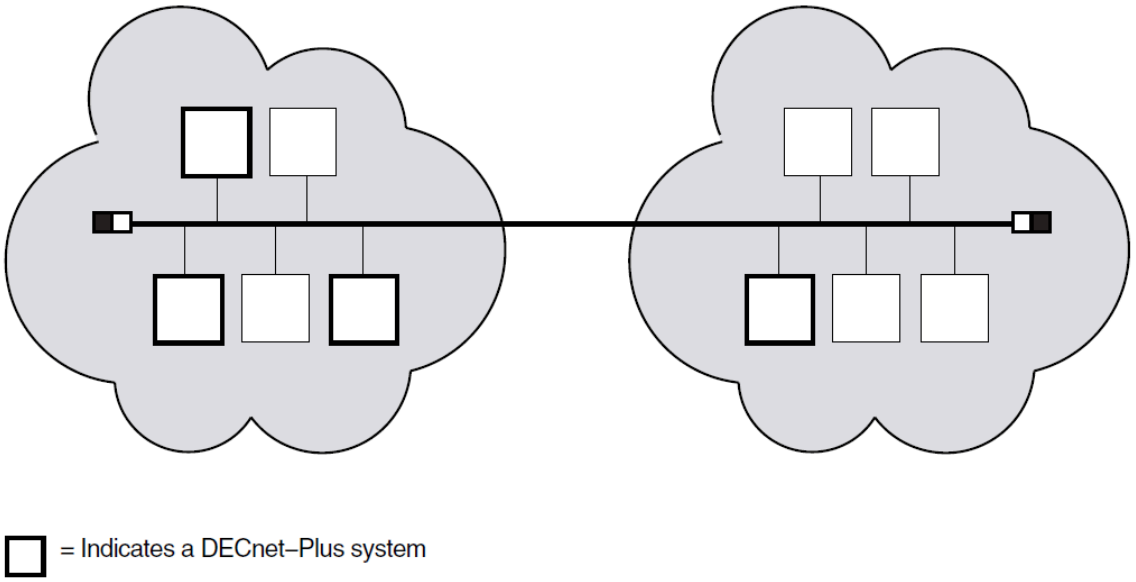
1. Install DECnet-Plus on one router and three end systems on the LAN.
2. Configure each DECnet-Plus system's address to the same node address it had as a Phase IV end node.

3. Configure the router to use the routing vector protocol at level 1 and level 2.

Areas 4 and 55 are still separate and distinct. Phase V routers allow multiple areas to exist on a LAN if the areas have Phase IV-compatible addresses.

Figure 3.2 shows the LAN once transition has started.

Figure 3.2. Phase IV LAN with Some DECnet-Plus Systems



Note

Although each system has the same address it had as a Phase IV node, the addresses are never displayed (by NCL, for example) in the familiar Phase IV-style. In this LAN, the end system with the address `4.3` actually has the address `49::00-04:aa-00-04-00-03-10:20`. DECnet-Plus automatically converts and stores the Phase IV address in this form.

The next step is to configure all level 1 routing in area 4 to link state as follows:

1. Upgrade all the routers in area 4 to DECnet-Plus.

2. Configure all the area 4 routers to run the link state protocol at level 1.

At this point, at level 1, area 4 is link state, and area 55 is routing vector. Level 2 in both areas is still routing vector.

Note

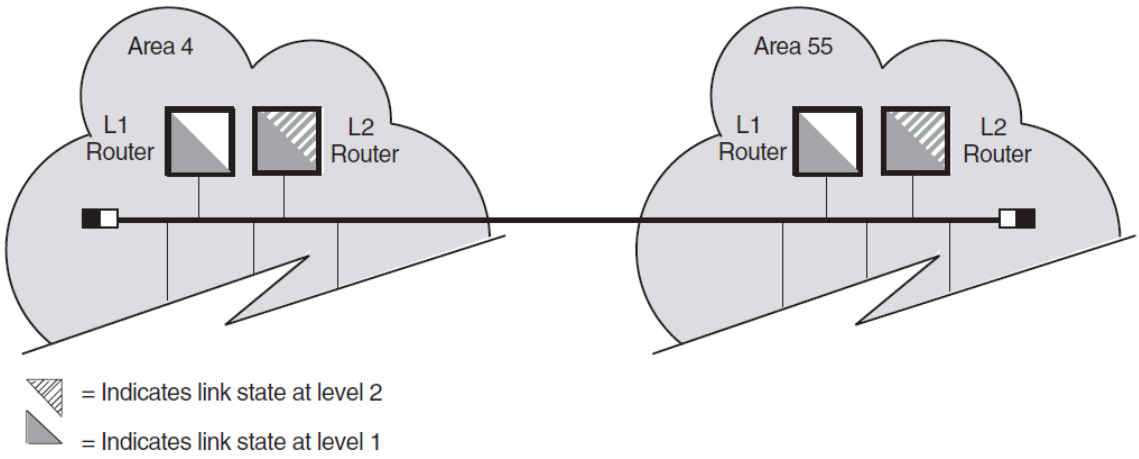
Changing the routing algorithm that each router runs does not affect the end systems. DECnet-Plus end nodes use the ES-IS protocol to communicate with Phase V routers, regardless of the routing algorithm the router is running.

Continue upgrading routers until both areas are running link state at level 1. This requires all routers to be upgraded to DECnet-Plus. Both areas are still using Phase IV-compatible addresses.

When all the level 2 routers in the LAN are running DECnet-Plus, you can convert the level 2 network to use link state simply by issuing a command to the router.

Figure 3.3 shows the network configuration.

Figure 3.3. LAN with Link State at Levels 1 and 2



The next step is to move area 4 to DECnet Phase V addressing, which allows you to enable autoconfiguration and eventually remove area 4 from the network. The LAN can contain only one DECnet Phase V address area, as well as multiple Phase IV areas.

This example shows the creation of DECnet Phase V address area `100`. In this example, the IDP is unique, with an AFI of `39` and an IDI of `345`. The new DECnet Phase V area is `39:345:00-64`. For instructions on how to construct your own unique IDP, see Chapter 4.

Follow these steps:

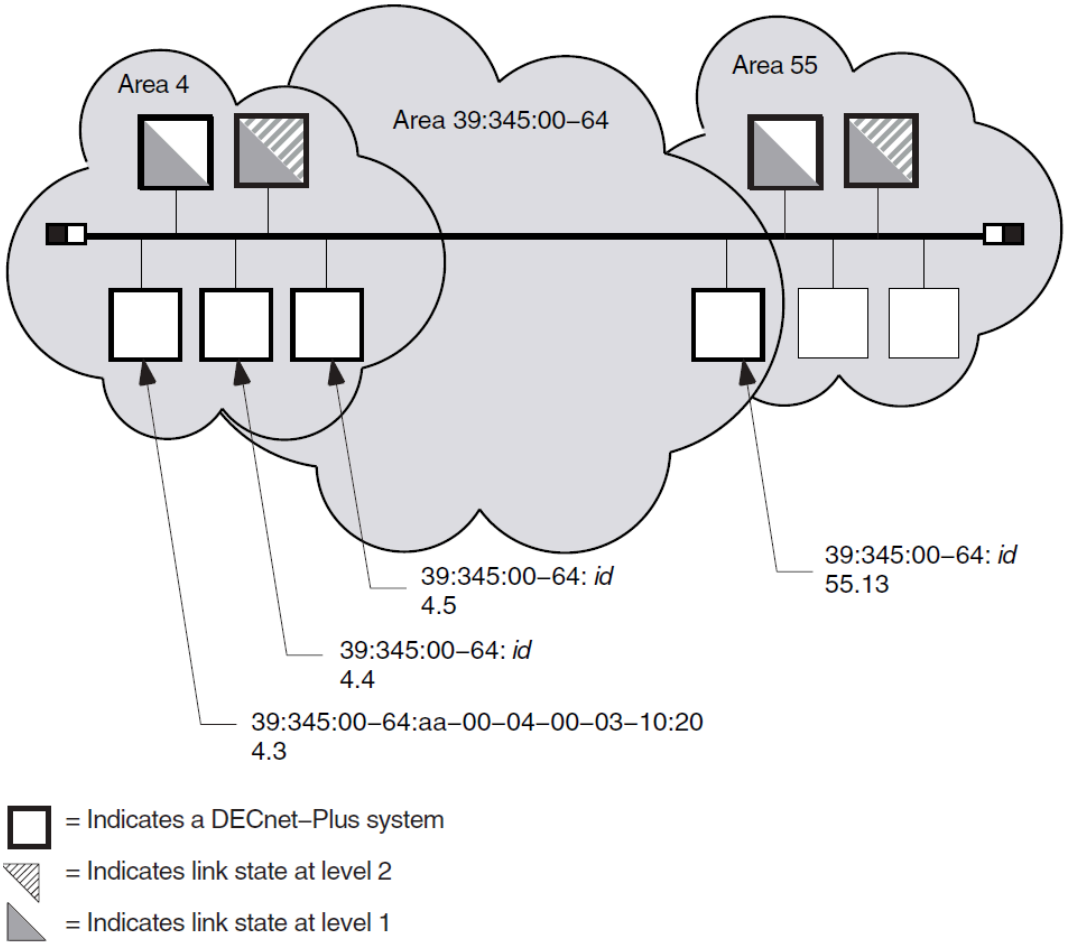
1. Upgrade all end systems in area 4 to DECnet-Plus.

2. During each configuration, answer YES when the script asks if you want to enable autoconfiguration.

3. Add the DECnet Phase V area to all the area 4 routers using NCL.

Immediately, each DECnet-Plus end system on the LAN (in both areas) autoconfigures to the new DECnet Phase V address. Each system also maintains its original manually-configured area 4 or 55 address. These systems are now multihomed (they have two distinct addresses). Thus, communication can reach the DECnet-Plus end systems in area 4 by the addresses `4.id` or `39:345:00-64.id`. The same is true for any DECnet-Plus end system in area 55. Figure 3.4 shows an example.

Figure 3.4. LAN with Multihomed Systems



Finally, when all Phase IV end systems in area 4 are either migrated to DECnet-Plus or moved to another area, you can remove area 4 from the network. Follow these steps:

1. Clear the Phase IV-compatible area 4 address from each end system using NCL.

2. Remove area 4 from each area 4 router using NCL.

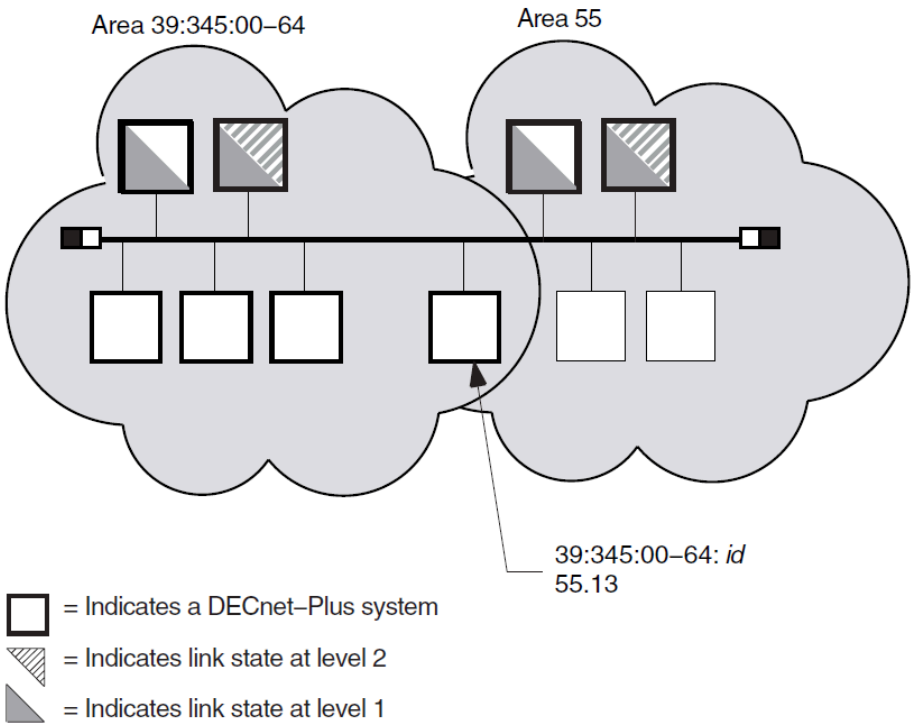
The results of this last set of changes are that:

- All end systems that were in area 4 are now autoconfigured into the DECnet Phase V address area `39:345:00-64` and are no longer multihomed.

- Unless an end system has a Phase IV-compatible address, it cannot communicate with Phase IV nodes or to nodes in areas that have only Phase IV routers. In this example, because the DECnet-Plus node in area 55 is still multihomed to a Phase IV-compatible address as well as the DECnet Phase V address space, it can still communicate with Phase IV nodes and to nodes in areas with Phase IV routers.

Figure 3.5 shows the next stage of the LAN configuration.

Figure 3.5. LAN with One DECnet Phase V Area



Eventually, the LAN manager finishes migrating the systems in area 55 to DECnet-Plus, so that area 55 can be removed from the LAN, thus completing the transition to a DECnet-Plus LAN. For now, though, the LAN can remain in the transition environment.

3.2.7. Configuring a DECnet-Plus End System in a Multivendor OSI Network

The DECnet-Plus software supports OSI addressing and OSI data-packet formats, allowing you to configure a DECnet-Plus end system as an OSI end system in a multivendor OSI environment. You can also configure other vendors' OSI end systems in a DECnet Phase V network (see your network management guide).

To configure a DECnet-Plus end system to operate in a multivendor OSI network, install the DECnet-Plus software and configure the system as follows:

- Answer `no` when the configuration script asks if you want the system to autoconfigure its address. Manually enter the NET or NETs specified by your network administrator.
- If you choose to use the DECdns name server software, configure the system as a DECdns clerk or server.
- If you choose to use the DECdts time server software, configure the system as a DECdts server or clerk.

3.3. Additional Tasks

You have several additional transition tasks. Some are performed automatically, and others you perform, perhaps on an on-going basis, as your network operation requires:

- If your network uses a distributed namespace, manage `.DNA_Registrar` and the other DECdns access control groups.

  See Section 7.4 for a discussion of possible access control policies. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for details on modifying access.

- If your network uses a distributed namespace, replicate the node directories and any other name server directories you have created.

  See Section 8.2 for guidelines on replicating node directories. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for general replication guidelines and for details on how to replicate directories.

- Become familiar with NCL. The following tools and documentation are available:

  - Use the CONVERT command of `decnet_migrate`. On the CONVERT command line, you specify an NCP command and the tool then displays the nearest NCL equivalent.

    For complete information about running `decnet_migrate`, see your network management guide.

  - Read Section 3.3.4 in this book as well as the introduction to using NCL and its command syntax in the *VSI DECnet-Plus for OpenVMS Network Control Language Reference Guide*.

  - Use the network management GUI and enable the display of NCL output. See *VSI DECnet-Plus for OpenVMS Network Management Guide* for more information.

- Modify all command procedures or shell scripts that issue NCP commands; convert the NCP commands to NCL equivalents.

  Use the commands of the `decnet_migrate convert` tool. For information about how to use these commands, see your network management guide.

- Periodically obtain a report of the network's configuration to check the progress of the transition.

  Use the `decnet_migrate` tool's COLLECT and REPORT commands. For information about how to use these commands, see your network management guide.

- If communication problems develop, use the SHOW PATH command of `decnet_migrate` to determine the communication paths between the nodes. For information about how to use this command, see your network management guide.

- Set up interphase links between level 2 routers using different routing algorithms.

  Use the `decnet_migrate` tool's `ipl_initialization_file` command. For information about how to use this command, see your network management guide.

- Convert the Phase IV object database to a DECnet-Plus application database.

  For OpenVMS, answer YES during `net$configure.com` to the question that asks if you want to convert Phase IV databases.

  For UNIX, use the `objtoncl` tool (see Section 3.3.1).

- Convert the DECnet Phase IV proxy file to a DECnet-Plus proxy database (for UNIX).

  Use the `proxytoncl` tool to convert the `/etc/dnet_proxy` file (see Section 3.3.2).

- Convert the MOP database to a DECnet-Plus MOP client database.

  For OpenVMS, answer YES during `net$configure.com` to the question that asks if you want to convert Phase IV databases.

  For UNIX, run `update_mopdb` (see Section 3.3.3).

3.3.1. Converting the Phase IV Object Database with the OBJTONCL Utility (UNIX Only)

Session Control maintains a database of applications that is similar to the Phase IV object database. This application database includes information about DECnet utilities, such as FAL (file access listener) or `dlogin`, and user-written application entities that are registered using NCL commands. The applications list is used to associate incoming connect requests with the relevant processes to handle them. If a Phase IV application uses the object spawner to start up a target application, you must convert the object database to an application database for the program to function properly.

The `objtoncl` utility allows you to convert your Phase IV object databases to DECnet-Plus application databases. This utility generates the appropriate NCL commands needed to make the conversion. You can use this utility to convert all the objects in a volatile database, all the objects in a permanent database, or selected objects in either type of database.

The `objtoncl` utility uses the following syntax:

```
/usr/field/objtoncl [-v] [-p] [-f] [filename] [object name]
```

The switches allow you to specify which object database to convert. The object name allows you to specify a particular object in a given database. Table 3.3 explains each switch.

Table 3.3. OBJTONCL Switches

Switch	Meaning
`-v`	Specifies conversion of the objects in the volatile database in `/usr/lib/dnet/objects_v`.
`-p`	Specifies conversion of the objects in the permanent databases in `/usr/lib/dnet/objects_p`. This switch does not have to be specified explicitly because it is the default.
`-f`	Allows you to specify a particular object database file name.

If you only want to convert specific objects, you can specify them as part of the command line argument. For example, the following arguments specify the conversion of specific objects in a specific database file:

```
objtoncl -f objects.saved myobj yourobj hisobj herobj
```

You can redirect the output of the command line to a file. You can then modify the NCL commands before inputting the file into NCL.

3.3.2. Converting the Phase IV Proxy File (UNIX Only)

Phase IV nodes use a proxy file, `/etc/dnet_proxy`, to specify proxy access. DECnet-Plus systems use a proxy database managed by NCL instead. When you upgrade your Phase IV node to DECnet-Plus, if you want to maintain the same proxy access, you can also upgrade the Phase IV proxy file to a DECnet-Plus proxy database.

DECnet-Plus provides a transition tool, `/usr/field/proxytoncl`, that generates the NCL commands necessary to create proxy entries in the DECnet-Plus proxy database. These proxy entries are equivalent to the entries in the original Phase IV proxy file.

Note

Check the entries in the Phase IV proxy file for wildcards:

- You can use wildcards to specify user names (*node*`::*`). The resulting NCL commands create proxy access for all users on the specified node.
- You cannot use wildcards to specify node names (`*::* or *::`*user_name*). The resulting NCL commands create proxy access on the node with the name "*".

To generate the NCL commands, issue the following command as superuser:

```
# /usr/field/proxytoncl
```

The command displays the NCL commands to standard output. You can redirect the NCL commands to a file and modify them before inputting the file to NCL. Two reasons to modify the resulting NCL commands before using them are:

- If the Phase IV proxy file contains more than one remote account granting access to the same target user, the resulting NCL commands create the same session control proxy subentity name for those proxy entries. Edit the commands so that the session control proxy subentity name is unique.
- The node names listed in the Phase IV proxy file are Phase IV node synonyms. The DECnet-Plus proxy database must have all DECnet-Plus full names.

  The `proxytoncl` command attempts to convert the Phase IV node synonyms from `/etc/dnet_proxy` to their full names. It prints out a warning message for the names it fails to convert. Edit the commands to replace each Phase IV synonym with a DECnet-Plus full name.

3.3.3. Converting the MOP Database (UNIX Only)

If your DECnet-Plus system will be a remote installation service (RIS) server or UNIX load host, you need to create the new MOP client database required by the MOP Version 4 protocol. You create this database using `/usr/field/update_mopdb`, which converts the Phase IV MOP database, `/usr/lib/dnet/nodes_p`, to the new DECnet-Plus MOP client database, `/var/dna/mop_client_db`.

During installation, if you have a `nodes_p` and no existing MOP client database on your system, `decnetsetup` automatically runs the MOP database conversion tool and creates a new MOP client database for you.

If `decnetsetup` did not create a database for your system, you can create one manually using the `/usr/field/update_mopdb` tool, as follows:

1. Make sure you have an updated copy of `/usr/lib/dnet/nodes_p` on your system.
2. Issue the following command:

   ```
   % /usr/field/update_mopdb
   ```

This creates an NCL script, named `/var/dna/scripts/create_mop_client_db.ncl`, which contains the commands that create the database.

3. Run the `create_mop_client_db.ncl` command file:

```
ncl> do create_mop_client_db.ncl
```

This command file creates the DECnet-Plus MOP client database, `/var/dna/mop_client_db`.

After completing these steps, your system is ready to function as an RIS server or an UNIX load host.

3.3.4. Becoming Familiar with NCL Commands, Network Management Modules, and Network Management Tasks

Use NCL to manage DECnet-Plus software by specifying entity values. You can use NCL commands to:

- Create and delete entities
- Enable an entity, which starts the corresponding software module's service
- Disable an entity, which halts the corresponding software module's service
- Modify attributes, or characteristics, of an entity
- Display attributes of an entity

NCL also provides module-specific functions, for example, loopback testing within the Modem Connect module.

In addition to its entity-related functions, NCL provides control functions such as:

- Aborting NCL operations
- Establishing defaults for NCL operations

You can execute NCL commands:

- Interactively at the NCL prompt
- In command files
- In initialization scripts

  Initialization scripts are part of the configuration procedure of DECnet/OSI for OpenVMS, DECnet-Plus for UNIX, and other DECnet-Plus products, for example, Phase V router systems.

- Using the graphical user interface for network management of Phase V nodes.

  `sys$system:net$mgmt.exe` (for OpenVMS) or `/usr/sbin/dna_mgmt` (for UNIX)

  Choose Default Actions from the Options pull-down menu to display the NCL commands performed on your behalf by the application.

Table 3.4 lists some network management tasks and the related network management entities that you set or modify using NCL commands to perform these tasks. You can use this table during transition to help you learn the modules and entities associated with specific network management tasks. A complete version of this table also appears in your network management guide.

Note

All required modules are enabled and all required entities are set during initial configurations, either automatically or by you. Use interactive NCL commands to change initial settings.

Table 3.4. Management Tasks and Related Network Management Modules

Task	Module	Entities
Enable or disable the system for networking	Node	`node`
Manage modem connections	Modem Connect	`modem connect`, `call control port`, `data port`, `line`, `template`
Manage CSMA-CD connections	CSMA-CD	`csma-cd`, `port`, `station`
Manage synchronous and asynchronous Data Communications Message Protocol (DCMP) connections	DCMP (UNIX only)	`ddcmp`, `link`, `link logical station`, `port`
Manage synchronous High-level Data Link Control (HDLC) connections	HDLC (UNIX only)	`hdlc`, `link`, `link logical station`, `port`
Manage X.25 level 2 protocol to exchange frames between a DTE and a DCE	LAPB	`lapb`, `link`, `port`
Manage X.25 level 2 for communications over a LAN	LLC2	`llc2`, `port`, `sap`, `sap link`
Downline load and upline dump communications servers such as DEC WAN router systems	MOP	`mop`, `circuit`, `circuit operation`, `circuit station`, `client`
Manage transport between DECnet-Plus nodes and between DECnet-Plus nodes and multivendor OSI systems	OSI transport	`osi transport`, `application`, `local nsap`, `local nsap remote nsap`, `port`, `template`
Manage transport between DECnet-Plus nodes and between DECnet-Plus nodes and Phase IV nodes	NSP	`nsp`, `local nsap`, `local nsap remote nsap`, `port`
Manage proxies, applications, and processes using the network	Session Control	`session control`, `application`, `local node synonym`, `port`, `tower maintenance`, `transport service`
Invoke loopback tests between applications on two nodes	Loopback Application	`loopback application`
Log events about network operations	Event Dispatcher	`event dispatcher`, `outbound stream`, `relay`, `relay logging`, `sink`, `sink inbound stream`

Some management tasks related to the transition are required only if your network operations need them:

- If your initial use of the Local namespace is part of the transition strategy, move to a distributed namespace when you have completed planning for a distributed namespace. For complete information about this procedure, see your network management guide.
- Set a consistent network buffer size, if you want to avoid the possible dropping of packets by Phase IV routers. For background information and instructions, see Section 3.3.5.
- If necessary, change your IDP and preDSP. For efficiency, you might choose to make the transition, perhaps with only some of your nodes, using the default IDP. This IDP is not usable for public networks; therefore, a later transition task in this situation is changing your IDP and preDSP to a real one obtained at the appropriate authorized registration agent. For information about registration agents, see Section 4.7. For instructions on making this change, see Section 3.3.6.
- Choose your main network management tool. An alternative to NCL is POLYCENTER Framework software, a separately orderable product.

3.3.5. Adjusting Buffer Sizes During Transition

In DECnet-Plus, the Transport layer segments and reassembles user messages to a size acceptable to the Routing layer on that system. This method differs from the Phase IV method, in which the Transport layer segments and reassembles user messages to a size appropriate to the complete path over which packets must pass.

DECnet-Plus routing has an additional segmentation capability: if a packet is in the *ISO 8473* packet format (the format used between DECnet-Plus systems), a DECnet-Plus router can segment it to fit the data-link size. The Routing layer of the destination system reassembles the segments before delivering the packet to the Transport layer. In contrast, a Phase IV router cannot segment packets to fit the data-link size. When forwarding a packet, a Phase IV router drops the packet if it is too large for the specific data link being used.

For a network in the transition, decide whether to:

- Establish a consistent buffer size throughout the network or,
- Accept that some packets will be discarded because they are too large for the data link being used.

To set a consistent network buffer size, use one of the following:

- On all DECnet-Plus systems, accept the default value of 570 for the routing characteristic `segment buffer size`. Accepting the default for this routing characteristic on DECnet-Plus systems ensures that they will not send packets larger than any Phase IV data link can handle.
- On all Phase IV routers, set the parameters `executor buffer size` and `line buffer size` to a value greater than or equal to the DECnet-Plus `segment buffer size`. Setting these parameters ensures that Phase IV routers will be able to forward packets that DECnet-Plus systems send.
- On all Phase IV systems, set the parameter `segment buffer size` to less than or equal to the smallest data-link blocksize. As a result, these systems will not create packets that are too big for the data-link size.

3.3.6. Changing the Network's IDP and preDSP

To change the IDP and preDSP of a network, follow these steps (waiting at least 24 hours between each step):

1. Create the new backtranslation directories, as needed. Use `decnet_register`.
2. Add the new IDP to the routers, using NCL or the appropriate router configuration program. DECnet-Plus nodes automatically learn the new IDP from the routers and update their own entries in the namespace.
3. Add the new IDP to the information stored in the namespace, using `decnet_register`. Do not change the local area values.

   This updates nodename entries that do not already have the new IDP.
4. Remove the old IDP from the routers, using NCL or the appropriate router configuration program. DECnet-Plus nodes automatically learn that the old IDP is no longer in use and update their own entries in the namespace.
5. Remove the old IDP from the information stored in the namespace, using `decnet_register`. This updates any node-name entries that have not already had the old IDP removed.

Chapter 4. Creating NSAP Addresses

This chapter discusses the format of NSAP addresses for DECnet-Plus systems and describes how to get unique identification for your DECnet Phase V network. You need this information to:

- Plan for and complete the transition to DECnet-Plus
- Design or create a namespace
- Register DECnet-Plus systems in the namespace
- Configure DECnet-Plus systems for communications

The DECnet Phase V addressing scheme for DECnet-Plus nodes complies with the ISO 8348 Addendum 2 addressing standard. This scheme uses the concepts of global addressing, addressing authorities, and addressing domains. Global network addressing is an ISO scheme designed to provide unique network addresses throughout the world.

A global network address is called a **network service access point (NSAP)**. Because it is used to determine the destination node for all packets, the NSAP must be unique for each node in a network.

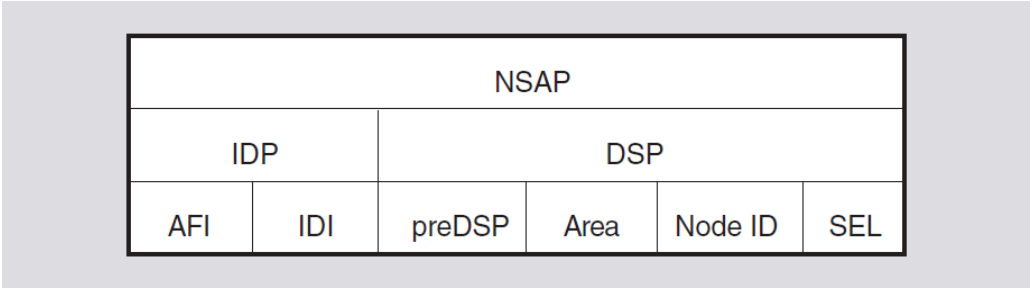
OpenVMS: For more information about addressing and service access points (SAPs), refer to the *VSI DECnet-Plus for OpenVMS Introduction and User's Guide*.

Some NSAP field values are assigned by an allocation authority, and some you assign yourself for your organization. Every NSAP has two primary fields:

- **Initial domain part (IDP)**, which consists of two subfields
- **Domain-specific part (DSP)**, which consists of four subfields

Figure 4.1 shows the parts of an NSAP and network entity title (NET). To compare these parts with Phase IV addresses, see Table 1.1.

Figure 4.1. Parts of an NSAP



4.1. IDP Values

The IDP helps ensure that NSAP values are globally unique. For this reason, IDP values are assigned by recognized authorities or are based on another value (such as a Telex number) that has already been assigned by some authority.

The IDP has two fields:

- **Authority and format identifier (AFI)**

  This field identifies the authority that allocated the globally unique IDP.

  The AFI value always consists of two decimal digits (from 0 to 9). Table 4.1 lists the recognized values and the corresponding allocation authorities.

  An AFI value of 49 indicates a private network, one that is not interconnected with other OSI networks and, therefore, does not need a globally unique IDP.

  Table 4.1 shows the NSAP field lengths for each AFI.

Table 4.1. Information for Building Unique NSAPs

Allocation Authority	AFI Value	Maximum Digits in IDI	Use AFI if the Leading Digit of the IDI is:	Maximum Digits in preDSP
Private	49	0 (none)	N/A	20
ISO DCC (single-country organizations)	39	3 (exact)	N/A	16
ISO 6523-ICD (international organizations)	47	4 (exact)	N/A	16
X.121 (X.25 address)	37 53	14 (maximum)	nonzero zero	6
F.69 (Telex number)	41 55	8 (maximum)	nonzero zero	12
E.163 (telephone number)	43 57	12 (maximum)	nonzero zero	8
E.164 (ISDN number)	45 59	15 (maximum)	nonzero zero	4

- **Initial domain identifier (IDI)**

  This field, combined with the AFI, makes the IDP globally unique for the allocating authority. Depending on the allocation authority identified in the AFI, the IDI value can be explicitly assigned by the authority, or it can be based on some other value that has already been assigned by that authority.

  The IDI value consists of zero or more decimal digits (from 0 to 9). The actual number of digits depends on the AFI. Some AFIs specify fixed-length IDIs, where you must enter all the digits in the IDI. Other AFIs specify variable-length IDIs, where you can enter up to the specified number of digits. Table 4.1 gives the IDI lengths for each AFI.

  If you are using the private AFI (49), do not specify an IDI. AFI 49 indicates a private network not interconnected with other OSI networks.

4.2. DSP Values

The DSP provides unique addresses within a specific IDP value. Different routing architectures might format and use the DSP indifferent ways. The format used by DECnet-Plus IS-IS, based on OSI IS-IS (ISO 10589), divides the DSP into four fields:

- **Prefix to the DSP (preDSP)**

  Also called the **high-order part of the DSP (HO-DSP)**, the format of this field is not defined by DECnet-Plus. If you assign a value to this field, it becomes part of the area, in conjunction with the IDP and local area values, and identifies the node's location for level 2 routing. If nodes have the same IDP and local area values, but different preDSP values, the nodes are indifferent routing areas.

  The preDSP value is zero or more hexadecimal digits (from 0 to F). The actual number of digits you can enter depends on the AFI. Table 4.1 gives the preDSP sizes appropriate for each AFI.

  The American National Standards Institute (ANSI) is the ISO DCC allocation authority in the United States. For NSAPs with AFI `39`, allocated by the ISO DCC, and AFI `47`, allocated by ISO 6523-ICD, the allocating authorities may assign an additional value for you to enter into the preDSP field. The reason for this additional value is that they have available only a limited number of IDIs, and they may give the same IDI value to different organizations. When this occurs, the preDSP value ensures global uniqueness.

  Allocation authorities other than ANSI may format the preDSP in other ways. VSI recommends that you do not assign a value to the preDSP field when you use AFIs other than `39` and `47`.

  ANSI formats the preDSP into the following subfields:

  - **DSP Format Identifier (DFI)**

    The value of this field is allocated by ANSI and consists of two hexadecimal digits. It indicates the format used for the rest of the high-order DSP, which is DFI-ORG-RES-RD.

  - **Organization ID (ORG)**

    The value of this field is allocated by ANSI and consists of six hexadecimal digits. This value is different for every organization and ensures global uniqueness.

  - **Reserved (RES)**

    The value of this field is allocated by ANSI and consists of four hexadecimal digits. This is a reserved field, whose value must always be zero.

  - **Routing Domain (RD)**

    The value of this field is allocated by each individual organization and consists of four hexadecimal digits. This field is used to separate an organization's network into multiple routing domains. It is defined so that routing domains can be identified uniquely by intermediate systems using a single short address prefix, without the necessity of listing multiple address prefixes, one for each local area within that routing domain. If your network needs only one routing domain, VSI recommends that you use the value `0000`.

- **Local area (LocArea)**

This value represents the local area within the routing domain (the node's local network area). Use the LocArea value, with the IDP and preDSP values, to identify the node's location for level 2 routing.

This value is used as the next four digits (two octets) in the DSP. The local area value is four hexadecimal digits (from 0 to F). Values `00-01 to 00-3F` are reserved for use as Phase IV-compatible area numbers for areas 1 to 63.

- **Node ID**

  This part of the DSP identifies a node within an area. It represents the node ID within the local area. Use it to identify the node for level 1 routing purposes.

  Use the node ID value as the next 12 digits (6 octets) in the DSP. This value is 12 hexadecimal digits (from 0 to F). Node IDs that range from AA-00-04-00-00-00 to AA-00-04-00-FF-FF are reserved for use as Phase IV-compatible node IDs (1.1 to 63.1023).

- **Selector (SEL)**

  This part of the DSP indicates the transport protocol you want to use. The selector value consists of two hexadecimal digits (from 0 to F). This value is used as the next two digits (one octet) in the DSP. DECnet-Plus uses these SEL values:

  - 20 (for NSAP specifying NSP transport)
  - 21 (for NSAP specifying OSI transport)
  - 00 (for an NET)

4.3. NSAP Entry and Display Formats

Enter an NSAP using either of two standard formats, DNA format or OSI format. DNA format consists of:

```
aa:iii...ii:pp-p...p-pp-ll-ll:nn-nn-nn-nn-nn-nn:ss
```

OSI format consists of:

```
aaiii...ii+ppp...pppllllnnnnnnnnnnnnss
```

where:

`aa`	is the AFI value in decimal
`ii...`	is the IDI value in decimal
`pp...`	is the preDSP value in hexadecimal
`ll...`	is the local area value in hexadecimal
`nn...`	is the node ID value in hexadecimal
`ss`	is the selector value in hexadecimal

DECnet-Plus always displays NSAPs in DNA format because this format separates the various fields, making the NSAP easier to read. The preDSP, local area, and node ID values are punctuated with hyphens after every two digits.

When you enter an NSAP using DNA format:

- You can type hyphens anywhere.
- You can omit hyphens.
- If the local area or node ID fields start with zeros, you can omit them.
- You cannot omit leading zeros for the IDI and preDSP values. Leading zeros in these fields affect the value of the NSAP when it is used in routing messages.

When you enter an NSAP using OSI format, you must always enter the proper number of digits because there is no other way to indicate where one field ends and the next starts, except between the IDP and DSP. The following examples show NSAPs in both formats.

NSAP with an IDP with the private AFI:

DNA format:

```
49::00-01:12-34-56-78-9A-BC:21
```

OSI format:

```
49+0001123456789ABC21
```

NSAP with an IDP with an allocated AFI and IDI:

DNA format:

```
41:23456789:A5:08-00-2B-19-0E-6C:20
```

OSI format:

```
4123456789+00A508002B190E6C20
```

If an NSAP has an unrecognized AFI or does not have the correct number of digits in the IDP or DSP, it is displayed in binary format. This format represents the value of the NSAP as a string of hexadecimal digits, as follows:

```
/3100A508002B190E6C20
```

4.4. Converting Phase IV Addresses to NSAPs

To convert a Phase IV address to an NSAP, you encode the Phase IV address into the local area and node ID fields of the NSAP. Fill in all the other NSAP fields in the normal manner.

To encode a Phase IV address into these fields, take the following steps:

1. Make the NSAP local area by using the Phase IV area in hexadecimal.

   For example, Phase IV area 1 becomes NSAP local area `00-01`, and Phase IV area 63 becomes NSAP local area `00-3F`.

2. The NSAP node ID represents the Phase IV area and ID. Construct it as follows:

   a. Convert the Phase IV area and ID to a single decimal value, using:

```
(area * 1024) + ID
```

b. Convert the resulting value to a four-digit hexadecimal number, and swap the first and last pairs of hexadecimal digits.

Use these as the last four digits of the NSAP's node ID field, and prefix them with `AA-00-04-00`.

For example, with:

- Network IDP `41:45436192:`
- Phase IV address 43.258
- The node using NSP transport

then create the NSAP as follows:

```
IDP and selector    = 41:45436192:local-area:node-id:20
43 decimal          = 2B    hexadecimal (local area)
(43 * 1024) + 258   = 44290 decimal
44290 decimal       = AD02  hexadecimal
AD02 swapped        = 02AD  hexadecimal (node ID)
```

The resulting NSAP is `41:45436192:00-2B:AA-00-04-00-02-AD:20`.

4.5. Converting NSAPs to Phase IV Addresses

To convert an NSAP to a Phase IV address, reverse the process described in the previous section.

You get the Phase IV area by converting the local area field from hexadecimal to decimal. If the resulting area value does not fall in the range of 1 to 63 inclusive, the NSAP is an extended address.

Calculate the Phase IV node ID from the node ID field as follows:

1. If the node ID field does not begin with `AA-00-04-00`, the NSAP does not contain a Phase IV address. If it does begin with this value, extract the last four digits of the node ID field and use them in the following steps.
2. Swap the first and the last pairs of digits, and convert the value to decimal.
3. Calculate the Phase IV area and ID values, using:

```
area  = value / 1024
id = value - (area * 1024)
```

If the calculated area value is not equal to the area value obtained from the NSAP's local area field, then the NSAP is an extended address.

For example, with NSAP `37:81076541234:00-19:AA-00-04-00-62-64:21`, calculate the Phase IV address by:

```
19 hexadecimal (from local area)  = 25 decimal
62-64 (from node ID)              = 6462 hexadecimal
6462 hexadecimal                  = 25698 decimal
25698/1024                        = area of 25
```

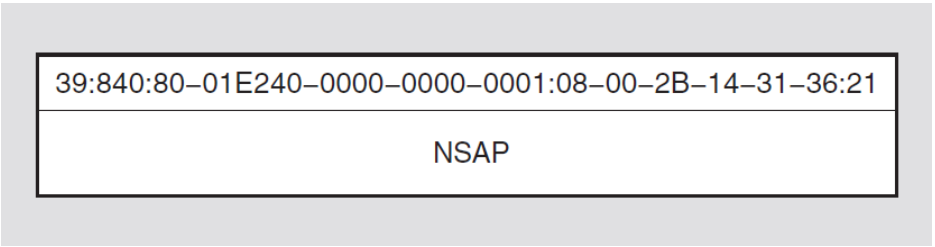
```
25698 - (25 * 1024)                = id of 98
```

The resulting Phase IV address is `25.98`.

4.6. Forms of the NSAP Displayed by NCL

Some NCL commands accept and display NETs, area addresses, node IDs, or address prefixes. These values are subsets of a node's full NSAP. This section defines these values and shows how the example NSAP in Figure 4.2 is divided into these values.

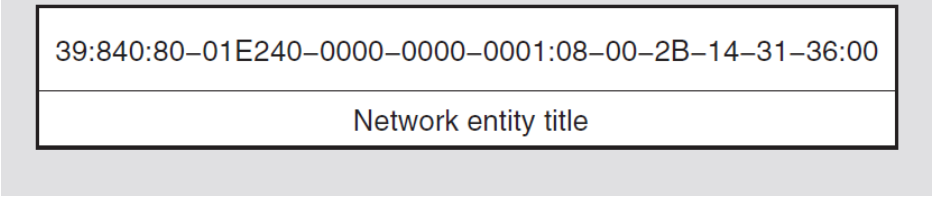
Figure 4.2. Example NSAP



Network Entity Title (NET)

The network entity title (NET) is the same as the NSAP, with a selector field of `00`. It identifies a node when the specific transport protocol to be used is either unknown or irrelevant. Figure 4.3illustrates an NET.

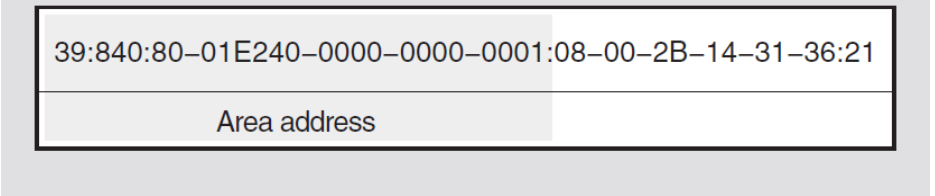
Figure 4.3. Example NET



Area Address

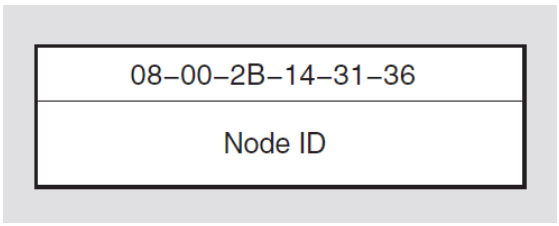
An area address consists of the IDP, preDSP, and the local area fields of the NSAP. It identifies a level 2 routing area. Figure 4.4 illustrates an area address.

Figure 4.4. Example Area Address



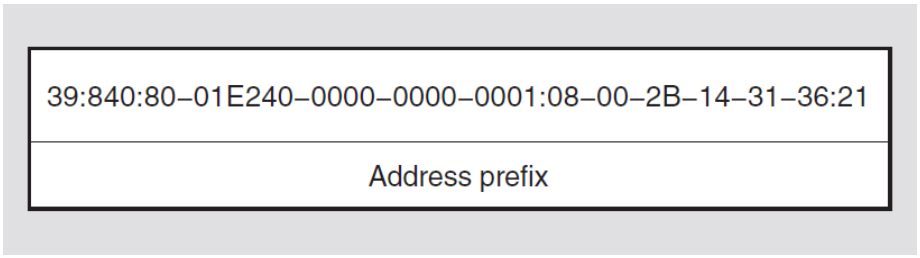
Node ID

Consists of the node ID field of the NSAP. Identifies a particular node within a level 2 routing area.



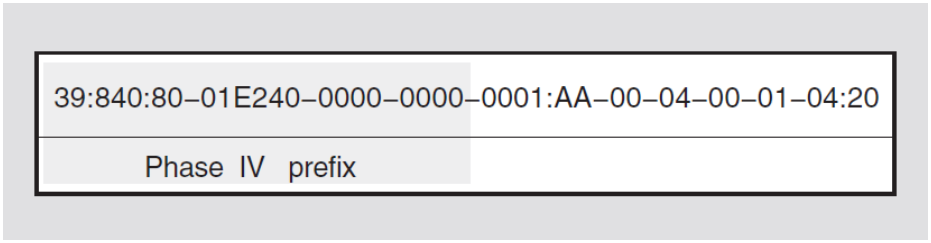
Address Prefix

Consists of some leading portion of the NSAP, and can be of any length from zero digits up to the full length of the NSAP. Can be used in a reachable-address table to indicate that packets with a destination NSAP beginning with a specified address prefix are to be routed through a specified circuit.



Phase IV Prefix

Consists of all values up to (but not including) the local area field. Includes the AFI, IDI, and, if used, the preDSP. This value is used by DECnet-Plus when converting between Phase IV addresses and NSAPs.



4.7. How to Obtain a Unique IDP and PreDSP

If your network is to be interconnected with other networks, its NSAP must have a unique IDP and, possibly preDSP, to differentiate it from all other networks. For a unique IDP (and preDSP), you can contact an allocation authority, or you can create your own according to the guidelines in the following sections.

The following sections describe where to obtain an IDI (and preDSP) for a specific AFI.

4.7.1. For a Private Network (AFI 49)

If your network will not be interconnected with other OSI networks, you do not need a unique IDP. Use the AFI 49 with no IDI.

4.7.2. Allocation Authority for Single-Country Organizations (AFI 39)

The IDI value is assigned directly for each country by an authorized registration agent, according to ISO Standard 3166. The registration authority agent in the United States is the American National Standards Institute (ANSI), with offices in New York City.

In this case, the resulting IDP is entered as given by the allocation authority. This IDP might also require the allocation of a value to be placed in the preDSP.

4.7.3. Allocation Authority for International Organizations (AFI 47)

The IDI value is assigned directly for each international organization by an authorized registration agent, according to ISO Standard 6523-ICD.

In this case, the resulting IDP is entered as given by the allocation authority. This IDP might also require the allocation of a value to be placed in the preDSP.

4.7.4. Using an X.25 Data Network Address for the IDI (AFIs 37 and 53)

The IDI value is based on CCITT Recommendation X.121 and is created from a public or private data network address. The parts of this IDI are:

- **Data network identification code (DNIC)**

  Assigned by the CCITT registration agent in a country; four digits long.

- **Private network identification code (PNIC)**

  Assigned by the CCITT registration agent or by the public or private data network to which your equipment is connected; three digits long.

- **Address within the private network**

  The network manager assigns this address for each node. Choose one node's address and use it to create a unique IDI value for your network.

  The node whose X.25 address you choose does not have to be part of the network for which you are creating the IDI. The address serves simply as a unique number that has been assigned to you.

Example:

```
Company's subnetwork address prefix (DNIC and PNIC)   = 8107654
Address within company's subnetwork                   = 1234
```

Resulting IDP value = `37:81076541234:`

4.7.5. Using a Telex Number for the IDI (AFIs 41 and 55)

The IDI value is based on CCITT Recommendation F.69 and is created from a Telex number that has been assigned to your company or organization. The parts of this IDI are:

- Destination code (country or network number), either two or three digits

  See the appropriate CCITT reference for the list of destination codes.

- Local Telex number, six or five digits (depending on the destination code)

  Choose one Telex number and use it to create a unique IDI value for your network. This number serves simply as a unique number that has been assigned to you. When choosing a Telex number, make sure it conforms to CCITT Recommendation F.69.

Example:

```
Destination code (for Switzerland)  = 45
Local Telex number                  = 43 61 92
```

Resulting IDP value = `41:45436192:`

4.7.6. Using a Telephone Number for the IDI (AFIs 43 and 57)

The IDI value is based on CCITT Recommendation E.163 and is created from a **public switched telephone network (PSTN)** number that has been assigned to your company or organization. The parts of this IDI are:

- World zone number
- Country or geographic area number
- Local number

Consult your local PSTN for the world zone and the country numbers.

Choose one telephone number and use it to create a unique IDI value for your network. This number serves simply as a unique number that has been assigned to you.

Example:

```
World zone number (for U.S.A.)               = 1
Geographic area number (for Massachusetts)   = 508
Local number                                 = 555-1192
```

Resulting IDP value = `43:15085551192:`

4.7.7. Using an ISDN Number for the IDI (AFIs 45 and 59)

The IDI value is based on CCITT Recommendation E.164 and is created from an **integrated services digital network (ISDN)** number that has been assigned to your organization. ISDN is a technology offered by the telephone carriers of the world that combines voice and digital network services in one media, allowing for digital data services through a single "wire." The standards that define ISDN are specified by CCITT.

The parts of this IDI are:

- Country code (CC)

The country code is one to three digits and is administered by CCITT.

- National destination code (NDC)

  For the U.S.A. and Canada, the NDC is the 3-digit area code.

- Subscriber number (SN)

  For the U.S.A. and Canada, the SN is the local telephone number.

The NDC and SN, which together make up the national significant number, are administered within each country, usually by the PTT.

Choose one ISDN number and use it to create a unique IDI value for your network. This number serves simply as a unique number that has been assigned to you.

Example:

```
Country code (for U.S.A.)                   = 1
National destination code (for New York)    = 212
Subscriber number                           = 555-6172
```

Resulting IDP value = `45:12125556172`

4.8. Example: NSAP Fields

This section shows examples of how an ANSI-formatted NSAP is divided into its component fields. For all examples, except for the one that shows a Phase IV prefix, the fields have the following values:

Field	**Value**
AFI	`39`
IDI	`840`
DFI	`80`
ORG	`01E240`
RES	`0000`
RD	`0000`
LocArea	`0001`
Node ID	`08002B143136`
SEL	`21`

The example showing the Phase IV prefix uses node ID and SEL values that are Phase IV-compatible, and these fields have the following values:

Field	**Value**
NodeID	`AA0004000104`
SEL	`20`

Figure 4.5 shows the example NSAP using DNA format.

Figure 4.5. NSAP Fields: DNA Format Examples

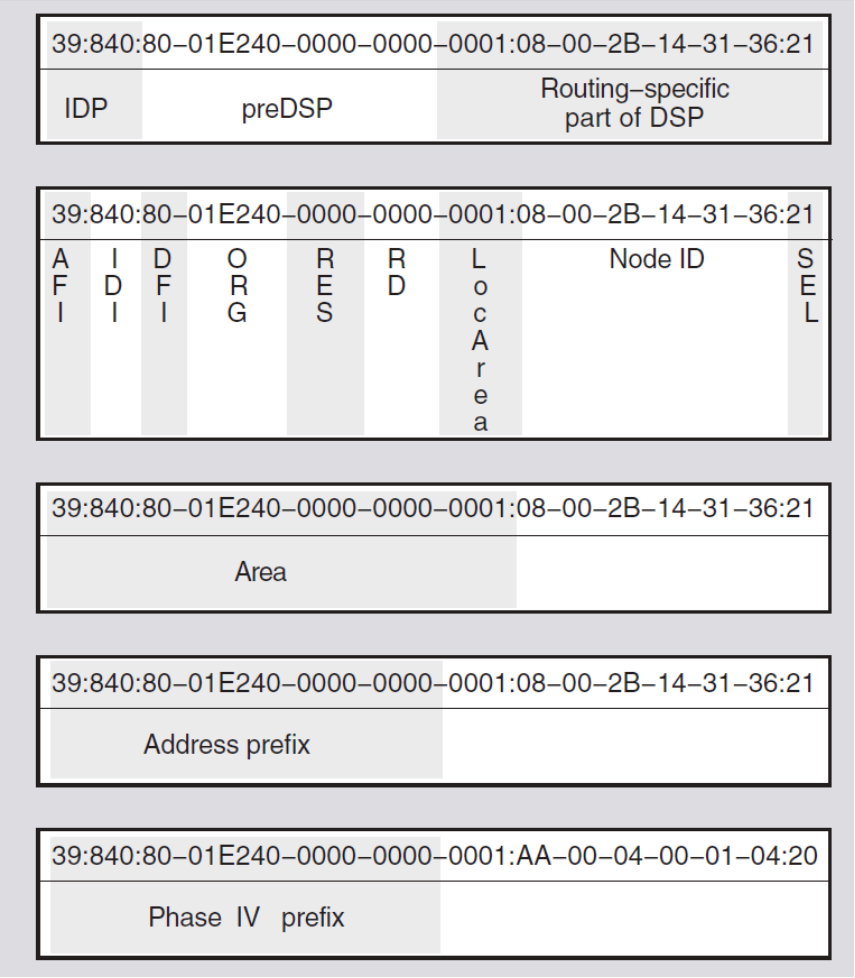
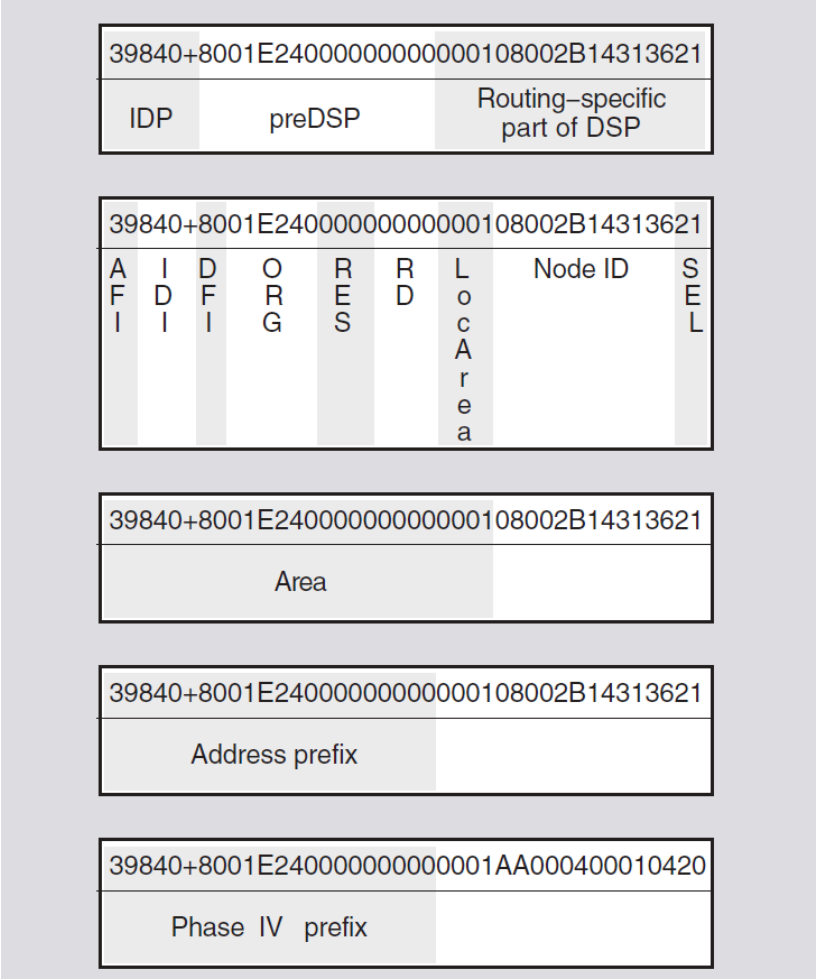


Figure 4.6 shows the same examples using OSI format.

Figure 4.6. NSAP Fields: OSI Format Examples



4.9. Example: Using Values Allocated by ANSI

This section applies to users in the United States who have applied to ANSI for a unique network identifier in the ISO DCC format. With values assigned by ANSI, an NSAP has the following DNA format:

```
AFI:IDI:DFI-ORG-RES-RD-LocArea:NodeID:SEL
```

Table 4.2 explains the NSAP fields used in this format.

Table 4.2. Entering NSAP Field Values

Field	Value	Meaning
AFI	`39`	ANSI provides a decimal value. Enter as two decimal digits.
IDI	`840`	ANSI provides a decimal value. Enter as three decimal digits.
DFI	`128`	ANSI provides a decimal value. Enter as two hexadecimal digits.
ORG	*dddddd*	ANSI provides a decimal value. Enter as six hexadecimal digits.

Field	Value	Meaning
RES	`0000`	Value reserved by ANSI for future use. Enter as four zeros.
RD	*dddd*	Enter as four hexadecimal digits.
LocArea	*dddd*	Enter as four hexadecimal digits.
Node ID	*dddddddddddd*	Enter as 12 hexadecimal digits.
SEL	*dd*	Enter as two hexadecimal digits.

All values supplied by ANSI are decimal numbers. Use the AFI and IDI in your NSAP as decimal values. However, due to the encoding rules of binary syntax NSAPs, you must convert the DFI and ORG values to hexadecimal. You can easily convert these two values from decimal to hexadecimal using a calculator or using the operating system's calculator feature.

For UNIX: Use the UNIX `bc` command. For example, with an ORG value of `123456`, use the following commands to convert the values:

```
# bc
obase=16        /* Set the output radix to base 16 (hex) */
128             /* Enter the decimal DFI value */
80              /* Resulting hex value */
123456          /* Enter the decimal ORG value */
1E240           /* Resulting hex value */
```

For OpenVMS: Use an OpenVMS symbol. For example, with an ORG value of `123456`, use the following commands to convert the values:

```
$dfi = 128
$org=123456
$sh sym dfi
  DFI = 128   Hex = 00000080  Octal = 00000000200
$sh sym org
  ORG = 123456   Hex = 0001E240  Octal = 00000361100
$
```

Using these values, the resulting NSAP has the following DNA NSAP format:

```
39:840:80-01E240-0000-RD-LocArea:NodeID:SEL
```

While this section discusses NSAP values from ANSI, the same information also applies to other Phase V addressing values, including the NET, Phase IV prefix, and address prefixes. In addition, this information is based on the draft ANSI standard:

"Data Communications — Structure and Semantics of the Domain Specific Part (DSP) of the OSI Network Service Access Point (NSAP) Address"

ANSI might review and modify this standard.

Chapter 5. DECdns, Local Namespace, and DECdts Concepts

The Local namespace is a discrete, nondistributed namespace that exists on an individual node and provides that node with a local database of name and addressing information. The Local namespace replaces functionality previously provided by the DECdns Local Naming Option. Depending on the number of address towers stored, the Local namespace is designed to scale to at least 100,000 nodes.

DECnet-Plus recognizes that when a node full name begins with `local:`, information for that node is stored in a Local namespace. The following are typical node full names properly formatted for the Local namespace: `local:.xyz.abc` and `local:.maximum`.

Unlike DECdns, the Local namespace does not employ backtranslation directories for address-to node-name translation.

The Distributed Name Service (DECdns) is a networkwide service that makes it possible to use network resources without knowing their physical location. Users and applications can assign DECnet-Plus names to resources such as nodes. The creator of a name also supplies other relevant information, such as the resource's network address, for DECdns to store. Users then need to remember only the name, and DECdns acts as a lookup service, providing the rest of the data when necessary. DECnet/OSI for UNIX does not contain DECdns server software.

The Distributed Time Service (DECdts) is a networkwide service that synchronizes the system clocks in the network's computers. DECdts enables distributed applications to execute in the proper sequence even though they run on different systems.

5.1. How DECnet-Plus Uses the Local Namespace and DECdns

All DECnet-Plus systems require a name service because the Session layer of DECnet-Plus uses it to map node names to addresses.

DECnet-Plus provides access to the node name and addressing information stored in one or more of the following name services.

- Local namespace – A discrete, nondistributed namespace that stores name and address information locally in database files.
- DECdns – Distributed Name Service, a distributed, global name service.
- Domain Name System – The Domain Name System (DNS/BIND) supported forstorage of IP addresses.

While configuring DECnet-Plus, the system administrator specifies one or more of the following name services to use on the node: the Local namespace; DECdns; or Domain. Note that in this version of DECnet-Plus, even when you are using the Local namespace the DECdns clerk software is still required on each node by DECnet-Plus.

By using DECdns, you can save network resources, manage node names automatically, and scale up to thousands of nodes.

When you make the transition from DECnet Phase IV to DECnet-Plus, the nodes in your network are given DECnet-Plus full names. Along with the nodenames, the name service stores the address and protocol information that DECnet needs to make connections between nodes. The DECnet-Plus software includes a tool, `decnet_register`, to help you create and manage node information in DECdns and the Local namespace.

The major benefit of using DECdns in a distributed namespace is the ease of storing node names. With DECdns, it is not necessary to maintain a node database on every system in the network. A few DECdns servers store node names, and all other nodes in the network can depend on those servers for node-name-to-address mapping. When data associated with the node name changes, DECdns propagates the change automatically to all servers that store that node name.

Another benefit of DECnet-Plus node full names is that they can be longer and hence more descriptive than Phase IV node names. Whereas Phase IV node names are limited to six characters, any full name, including a node name, can be as long as 255 characters. See Chapter 6 for complete guidelines for all node names.

The Distributed Time Service (DECdts) is another important user of DECdns in DECnet-Plus. DECdts and DECdns actually depend on each other. DECdts uses DECdns as a networkwide registry for global time servers that synchronize system clocks in the network. DECdns uses timestamps to determine the order in which changes to its data occur, and it depends on DECdts to synchronize time on DECdns servers so their timestamps are consistent. Synchronized clocks are important to any distributed application that needs to keep track of the order in which events occur across multiple systems.

5.2. The Name Service Search Path

DECnet-Plus constructs a name service search path file from information entered during configuration. This file determines the order in which the namespaces available on the node will be searched and, for each namespace, any defaulting rules to allow users to enter abbreviated node names.

The name service search path applies system wide and allows DECnet-Plus to search a list of name services in a predetermined order when looking up names or addressing information. The *primary* name service (the name service to be searched first) is listed before the *secondary* name services. The secondary name services are listed in the order in which they are to be searched after the primary name service.

If you choose to use a search path and configure more than one name service on your system, the ordering of the name services is *very* important.

The search path also contains a list of name service keywords, each followed by a *naming template* that specifies a "defaulting rule" so users can enter shorter node names. In each template, the user-supplied portion of the name (usually the node's terminating name or rightmost simple name) is indicated with an asterisk (*). For example, if the DECdns template is: `"ABCDE:.xyz.*"` and a user supplies the name `foo`, then the following full name: `ABCDE:.xyz.foo` will be looked up in namespace ABCDE in the DECdns name service.

Only one asterisk should be supplied per template. Only the first occurrence of an asterisk (*) in the template is substituted with the user-supplied name. Any additional asterisks are passed to the name service as part of the full name. When you specify a template without an asterisk, the template string is passed to the name service unchanged.

If the user-supplied name should be passed to the name service as entered by the user, the template should simply be specified as follows: `"*"`.

Note that the system maintains two separate search paths:

- One search path supports forward translation or naming (node-name-to-address translation).
- Another separate search path supports backtranslation (address-to-node-name translation).

During DECnet-Plus configuration, the system administrator uses `net$configure.com` (on OpenVMS systems) and `decnetsetup` (on UNIX systems) to set up one or more name services on each node and set up the search path. The procedure for setting up and maintaining the search path on your node is specific to the OpenVMS and UNIX systems. See your installation and configuration guide for more information.

5.3. How DECdns Works

Operation of the name service involves several major participants:

- Client applications
- Servers
- Clerks
- Clearinghouses

DECdns uses a client/server model: an application, such as DECnet-Plus, that depends on DECdns to store and retrieve information because it is a **client** of DECdns. Client applications create names for resources on behalf of their users. Through a client application, a user can supply other information for DECdns to store with a name. This information is stored in data structures called **attributes**. Then, when a client application user refers to the resource by its DECdns name, DECdns retrieves data from the attributes for use by the client application.

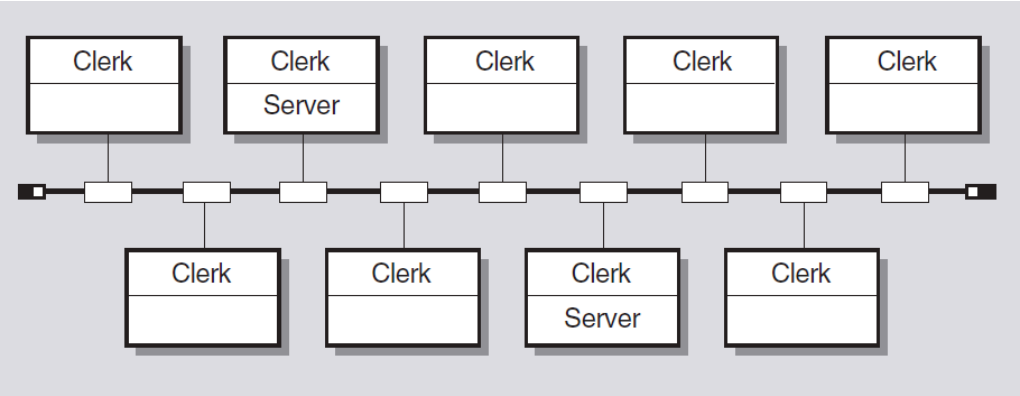
A system running DECdns server software is a **DECdns server**. A DECdns server stores and maintains DECdns names and handles requests to create, modify, or look up data. You designate a system as a DECdns server during configuration of the network software.

A component called the **clerk** is the interface between client applications and DECdns servers. A clerk must exist on every DECnet-Plus node and is created during configuration of the network software. The clerk receives a request from a client application, sends the request to a server, and returns the resulting information to the client. This process is called a **lookup**. The clerk is also the interface through which client applications create and modify names. One clerk can serve many client applications.

The clerk **caches**, or saves, the results of lookups so that it does not have to go repeatedly to a server for the same information. The cache is written to disk periodically so the information can survive a system reboot or the restart of an application. Caching improves performance and reduces network traffic.

Figure 5.1 shows a sample configuration of DECdns clerks and servers on a nine-node LAN. Every node is a clerk, and DECdns servers run on two selected nodes.

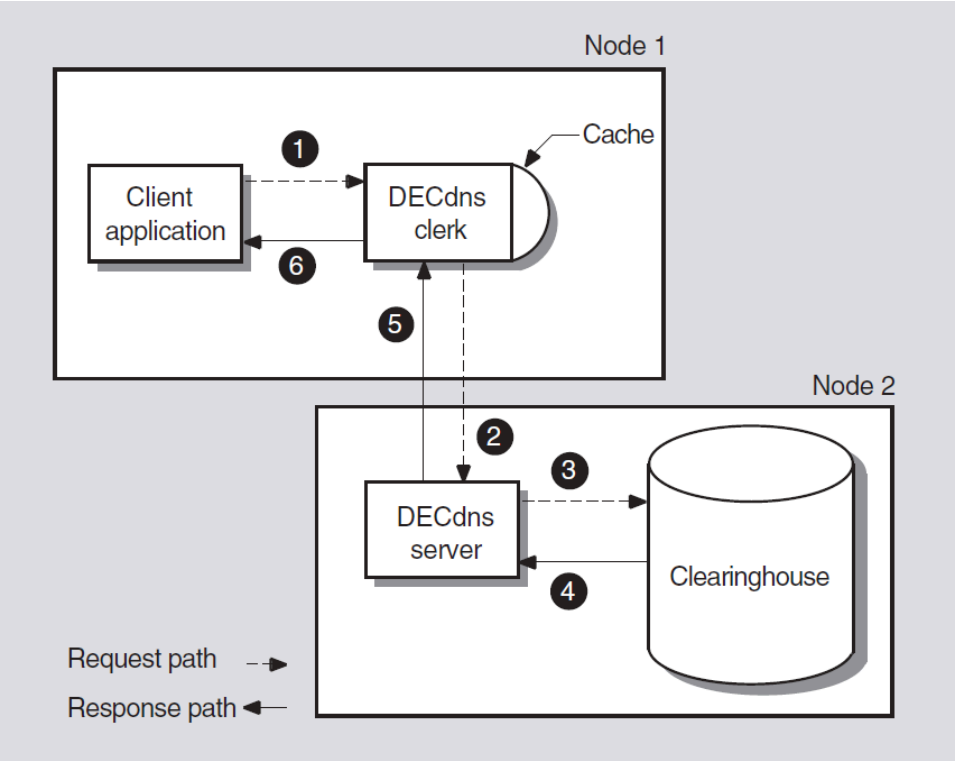
Figure 5.1. Sample DECdns LAN Configuration



Every DECdns server has a database called a **clearinghouse** in which it stores names and other DECdns data. The clearinghouse is where a DECdns server adds, modifies, deletes, and retrieves data on behalf of client applications. Although more than one clearinghouse can exist at a server node, does not recommend it as a normal configuration.

Figure 5.2 shows the interaction between a DECdns client, clerk, server, and clearinghouse during a simple lookup. First, the clerk receives the lookup request from the client application (Step 1) and checks its cache. Not finding the name there, the clerk contacts the server on Node 2 (Step 2). The server finds the name in its clearinghouse (Steps 3 and 4) and returns the requested information over the network to the clerk (Step 5), which passes it to the client application (Step 6). The clerk also caches the information so it does not have to contact a server the next time a client requests a lookup of that same name by the same user.

Figure 5.2. Simple Lookup



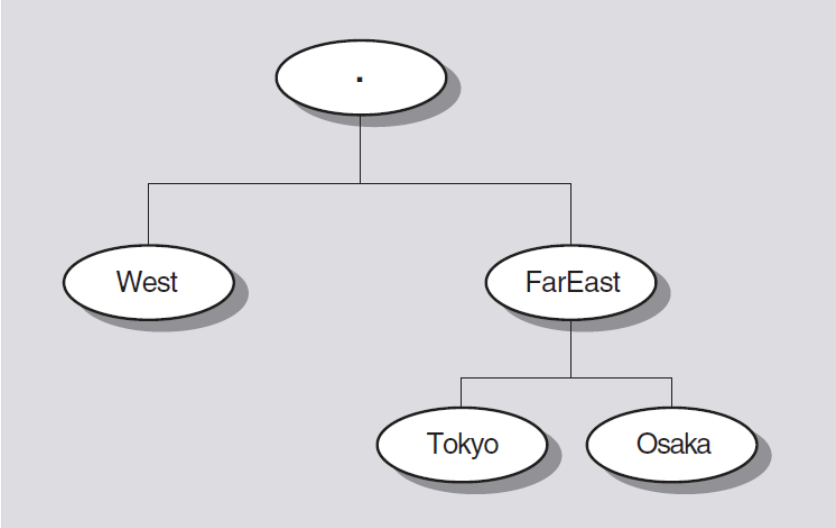
5.4. Contents of a DECdns Namespace

The total collection of names that one or more DECdns servers know about, look up, manage, and share is called a DECdns **namespace**. When you create a DECdns namespace, you organize DECdns names into a hierarchical structure of **directories**. DECdns directories are conceptually similar to the directories that you create in your operating system's file system. They are a logical way to group names for DECdns-specific management or usage purposes.

The highest-level directory in the namespace is denoted by a dot (.) and is called the **root** directory. The root is created automatically when you initialize a namespace. You can then create and name other directories below the root. Any directory that has a directory beneath it is considered the **parent** of that directory. Any directory that has a directory above it is considered a **child** of the directory above it.

Figure 5.3 shows a simple hierarchy of directories. The root directory (.) is the parent of the directories named `West` and `FarEast`. The `FarEast` directory is a child of the root directory and the parent of the `Tokyo` and `Osaka` directories.

Figure 5.3. Example Namespace Directory Hierarchy



The complete specification of a DECdns name, going left to right from the root directory to the entry being named, is called the **full name**. Each element within a full name is separated by a dot and is known as a **simple name**. For example, the `Osaka` directory in the preceding figure might contain an entry for a node whose simple name is `Node01`. The node's full name would be `.FarEast.Osaka.Node01`. A full name also can include a namespace nickname, but this is not needed when only one namespace exists in a network. When you must specify a namespace, use the following format:

```
NamespaceNickname:.DirectoryPath.NodeObject
```

5.4.1. Replicas and Their Contents

Each physical copy of a directory, including the original copy, is called a **replica**. Directory replicas are the units by which you distribute names in clearinghouses throughout the namespace. You can think of a clearinghouse as a collection of directory replicas at a particular server. After you create a directory in one clearinghouse, you can create replicas of it in other clearinghouses.

All of the replicas of a specific directory in the namespace constitute that directory's **replica set**. DECdns performs a periodic **skulk operation** to ensure that all replicas of a directory remain consistent.

During a skulk operation, DECdns collects all changes that have been made to the directory since the last skulk completed and disseminates the changes to all replicas of the directory. Every replica must be available for the skulk to be considered successful. Two types of replicas can exist:

- Master
- Read-only

A replica's type affects the processing that can be done on it and the way DECdns updates it. The type of replica DECdns uses when it looks up or changes data is invisible to users. However, it helps to understand how the two types differ.

The **master replica** is the first instance of a specific directory in the namespace. After you make copies of the directory, you can designate a different replica as the master, if necessary, but only one master replica of each directory can exist at a time.

The master replica is the only directly modifiable replica of a directory. DECdns can create, change, and delete information in a master replica. Because it is modifiable, the master replica incurs more overhead than read-only replicas, which DECdns keeps up-to-date but never modifies directly. The master replica also is the place where skulks originate.

A **read-only replica** is a copy of a directory that is available only for looking up information. DECdns does not create, modify, or delete names in read-only replicas; it simply updates them with changes made to the master replica. Read-only replicas save resources because DECdns does not make changes directly in them, nor does it have to gather changes from them to disseminate to other replicas.

Directory replicas can contain three kinds of entries:

- Object entries
- Soft links
- Child pointers

5.4.1.1. Object Entries

An **object** is any real-world network resource, such as a disk, application, or node, that is given a DECdns name. When an object name is created, client applications and the DECdns software supply additional data in the form of **attributes** to be stored with the name. The name and its attributes make up the **object entry**. When a client application requests information associated with the name, DECdns returns the value of the relevant attribute or attributes.

Every object has a defined class, which is stored as an attribute of the object entry. For example, a node object's class is `DNA$Node`. Another important class of object is the group object (class `DNS$Group`). **Groups** let you associate, or manage as a group, names that have something in common. They do this by mapping a specific name (the group name) to a set of names denoting the group members. DECdns managers can create groups that assign several users a single set of access rights to names. Two such access control groups are created automatically during configuration of the DECnet-Plus and DECdns software. See Section 5.6.2 for a description of them and recommendations on their use.

5.4.1.2. Soft Links

A **soft link** is a pointer that provides an alternate name for an object entry, directory, or other soft link in the namespace. You can restructure a namespace on a minor scale by creating soft links that point

from an existing name to a new name. Soft links can be permanent, or they can expire after a period of time that you specify. If the name that a soft link points to is deleted, DECdns deletes the soft link automatically.

DECnet-Plus uses **backtranslation** soft links to map a node address to a node's full name when necessary. Another kind of soft link, called a **node synonym**, enables applications that do not support the length of a DECdns full name to continue using six-character Phase IV-style node names. Section 5.5 explains how DECnet-Plus uses backtranslation and node synonym soft links.

5.4.1.3. Child Pointers

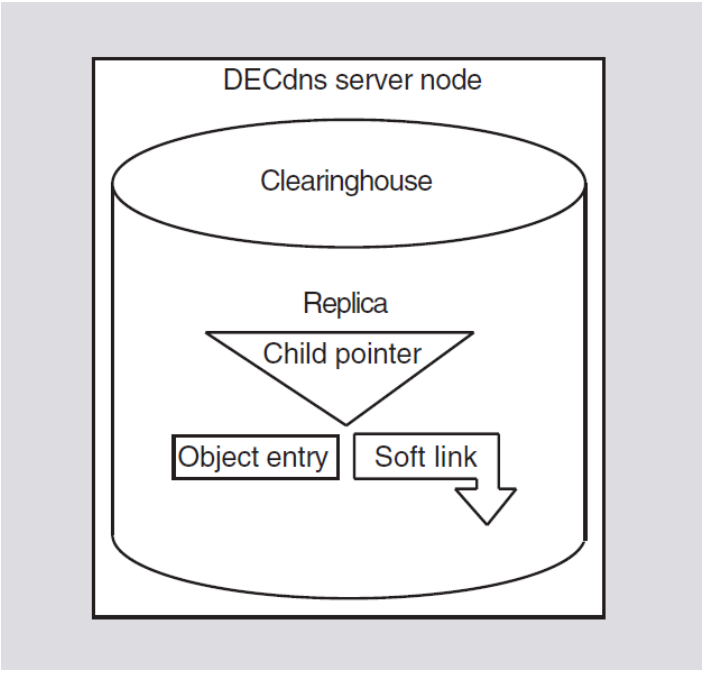
A **child pointer** connects a directory to another directory immediately beneath it in a namespace. Users and applications do not create or manage child pointers; DECdns creates a child pointer automatically when someone creates a new directory. The child pointer is created in the directory that is the parent of (one level above) the directory to which it points. DECdns uses child pointers to locate directory replicas when it is trying to find a name in the namespace. Child pointers do not require management except in rare problem-solving situations.

5.4.2. Putting It All Together

To summarize, a DECdns namespace consists of a complete set of names shared and managed by one or more DECdns servers. A name can designate a directory, object entry, soft link, or child pointer. The logical picture of a DECdns namespace is a hierarchical structure of directories and the names they contain. Every physical instance of a directory is called a replica. Names are physically stored in replicas, and replicas are stored in clearinghouses. Any node that contains a clearinghouse and runs DECdns server software is a server.

Figure 5.4 shows the components of a DECdns server. Every server manages at least one clearinghouse containing directory replicas. A replica can contain object entries, soft links, and child pointers. The figure shows only one replica and one of each type of entry possible in a replica. Normally, a clearinghouse contains many replicas, and a replica contains many entries.

Figure 5.4. Components of a DECdns Server Node



5.5. How DECnet-Plus Uses the DECdns Namespace

DECnet-Plus Session Control uses DECdns to store three kinds of entries:

- Node object entries
- Backtranslation soft links
- Node synonym soft links

The `decnet_register` tool is available to help you create these entries in the distributed namespace. The DECnet configuration procedure starts the tool automatically after creating a new name space. If you choose to use the Local namespace, the `decnet_register` tool also enables you to create entries in the Local namespace. You also can run `decnet_register` separately to create and manage node information in the namespace. This section introduces each type of node entry and describes how it is created and used.

5.5.1. Node Object Entries

In a Phase V network using DECdns, every DECnet node has a corresponding object entry in the DECdns namespace. During the DECnet configuration procedure, you must supply a DECdns full name for your node (the full name includes the complete path from the root directory to the node's simple name) if you decide to use a distributed namespace. The full name becomes the name of the node's object entry in the namespace. If you decide to use a Local namespace, you must enter `local` as your namespace name during the DECnet configuration procedure.

The Local namespace exists on a single node and provides a local database of names. The local name file, called `SYS$SYSTEM:DECNET_LOC_NODE_DEFINITIONS.TXT` on OpenVMS and `/var/dna/decnet_loc_node_definitions` on UNIX, contains a list of nodenames, node synonyms, and network addresses. You can also use the `decnet_register` tool to verify or modify node addresses as your network changes.

In a distributed namespace, the node object entry has an attribute called `DNA$Towers`, in which DECnet-Plus stores the node's address. The attribute name reflects the fact that a DECnet Phase V address consists of separate layers, sometimes called tower floors, in the DNA and OSI network models. A DECnet-Plus node communicates with another DECnet-Plus node by using the address information stored in the node's `DNA$Towers` attribute.

You can store DECnet-Plus node object entries in any directory in the distributed namespace. When you register a DECnet-Plus node in a distributed namespace, the `decnet_register` tool creates any directories in the node's full name that do not already exist. You also can create directories with the DECdns Control Program. However, recommends that you use `decnet_register` to create directories that will contain node object entries. Using `decnet_register` ensures that the directories are created with the proper access control; see Section 7.4 for details.

If your network includes nodes that are not yet upgraded to DECnet-Plus, their names and addresses also must be in the namespace in order for a DECnet-Plus node to communicate with them. You can use the `decnet_register` tool to register each Phase IV node individually in any directory in the namespace.

Alternatively, you can use the `decnet_register export` and `import` commands to extract the node information from the Phase IV database and enter it in the namespace.

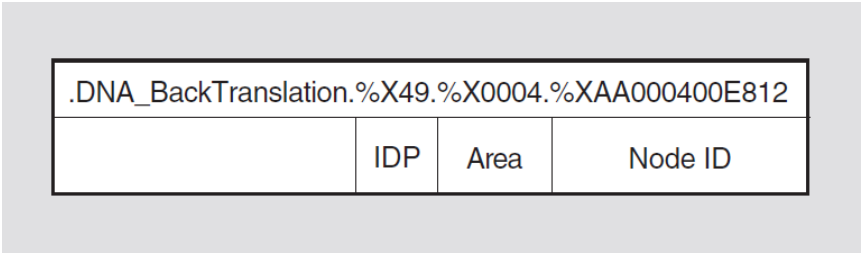
5.5.2. Backtranslation Soft Links

Backtranslation soft links enable Session Control to translate anode address to the full name of a node object entry. For example, end user applications often require the node name of the source of incoming connect requests. A DECnet-Plus node can include its name in the data it sends to request a connection, but a Phase IV node provides only an address. A backtranslation soft link contains the data necessary to make the translation from an address to a DECdns node name.

Backtranslation soft links are stored in a hierarchy of directories beginning with a directory named `.DNA_BackTranslation`. The `.DNA_BackTranslation` directory contains one child directory for each IDP in the network. Each IDP directory contains one child directory for each local area defined within that IDP. Each local area directory contains one soft link for each node in that area.

Unlike DECdns, the Local namespace does not use backtranslation directories for address to node name translation. In a DECdns namespace, the `decnet_register` tool automatically creates the `.DNA_BackTranslation` directory, as well as IDP and area directories derived from all existing Phase IV addresses. In addition, the tool has an option for creating new IDP and area directories for DECnet-Plus nodes.

The following illustration shows a backtranslation soft link:



The %X symbols in the IDP, area, and node ID segments of the soft link indicate to DECdns that the numbers that follow are in hexadecimal notation. For a complete explanation of the DECnet Phase V address structure, see Section 1.3.1.

When you register a Phase IV node in a distributed namespace, `decnet_register` constructs a backtranslation soft link based on whatever address the user specifies in the node registration information. When you register a DECnet-Plus node, the node creates its own backtranslation soft link.

When you register a node in a local namespace using `decnet_register`, you can specify a Phase IV address and/or one or more towers in the local name file. Backtranslation information is maintained in the database.

5.5.3. Node Synonyms

A node synonym is a soft link that points to the full name of anode object entry. Node synonym soft links, in the form of a Phase IV-style node name, exist so that Phase IV applications that do not support the length of DECdns full names can continue to use six-character node names. A node synonym is optional; the system manager can supply it during DECnet-Plus configuration. Node synonyms are stored in a required directory called `.DNA_NodeSynonym`, which `decnet_register` creates automatically during DECnet-Plus configuration of a distributed namespace. In the Local namespace, synonym mapping is maintained by the database.

When DECnet-Plus Session Control receives a node name of six characters or fewer that does not include a leading dot or namespace nickname, it tries to look up that name in the

`.DNA_NodeSynonym` directory in a distributed namespace. Suppose, for example, DECnet-Plus is given the node name MIS01. Session Control sends a request to DECdns to look up `.DNA_NodeSynonym.MIS01`. The node synonym exists and points to the full name of the node, `ABC:.Geneva.Admin.MIS01`. Session Control can then ask DECdns to look up the node object entry named `ABC:.Geneva.Admin.MIS01` and get the address information it needs to contact the node.

Node synonyms are not for users to avoid typing the DECdns full name of a node. Local names, and a feature called the **local root**, offer that capability on a per-node basis. For details, see the *VSI DECnet-Plus for OpenVMS DECdns Management Guide*.

5.6. How DECdns Protects Names

The `decnet_register` tool `manage` command lets you control access to clearinghouses, directories, and their contents. Access to clerks and servers is determined by operating system rights identifiers.

5.6.1. Access Rights

The DECdns access rights are read, write, delete, test, and control. Each right has a slightly different meaning, depending on the kind of name with which it is associated. In general, the meanings are as follows:

- Read access lets users view data.
- Write access lets users change data.
- Delete access lets users remove data.
- Test access lets users test whether an attribute of a name has a specific value without being able to see any values (that is, without having read access to the name). DECdns uses test access internally to test for group membership during access checking.
- Control access lets users change the access control on a name and grants other powers normally given only to an owner, such as the right to replicate a directory or relocate a clearinghouse.

You assign access in the form of an **access control entry** (ACE), which consists of two parts: the **principal** portion and the list of access rights that the principal has to the associated name in the namespace. The principal part of the ACE can be the name of an individual user, a group name, or a name with one or more wildcard characters to denote several users. For example, the following wildcard notation denotes all users on all nodes in the specified namespace:

`*...`

A name can have multiple ACEs associated with it. The complete set of ACEs for a particular name is called an **access control set** (ACS) and is an attribute of that name.

5.6.2. Access Control Groups

It is often useful to set up an access control policy when you plan your namespace, choosing a select group of users who will have control over the root of the namespace and deciding how to delegate access to other directories. Two DECdns groups are available to help you implement such an access control policy: `.DNS_Admin` and `.DNA_Registrar`.

The `.DNS_Admin` group is an access control group meant to contain the names of people responsible for managing the entire namespace. The DECnet configuration procedure creates this group when it creates a new namespace. If you want to use the `.DNS_Admin` group, you must explicitly add members and create ACEs for the directories that you want the group to control.

You can create ACEs for the `.DNS_Admin` group that propagate from the root directory down to all subsequent child directories and their contents. Propagating access rights allows `.DNS_Admin` members to monitor any changes or problems in the namespace. Consider a policy requiring that the `.DNS_Admin` group also be given access to all clearinghouses. A clearinghouse is named in a directory but not contained in that directory, so it does not inherit access control from a directory's ACS. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for details on creating ACEs, managing groups, and implementing a namespacewide access control policy.

The `.DNA_Registrar` group is an access control group for people who manage node object entries and node soft links. You must explicitly add members to this group as well. You might want to add the `.DNS_Admin` group as a member of this group, thereby automatically giving `.DNS_Admin` members access to whatever `.DNA_Registrar` members can manage.

The `decnet_register` tool creates the `.DNA_Registrar` group and grants the group full access to all directories, object entries, and soft links that the tool creates. Section 7.4 describes the access rights that the tool automatically assigns, and suggests ways to modify the default access control policy.

5.7. DECdns Management

DECdns provides two user interfaces for examining and managing the components of a namespace: a windows-based Browser utility and a DECdns Control Program, called `DNS$CONTROL` on OpenVMS systems and `dnscp` on UNIX and ULTRIX systems. The DECdns Control Program is an interface that accepts commands targeted for specific entities, such as clerks, clearinghouses, or directories. The browser is a tool for viewing the content and structure of a namespace. It runs on workstations with DECwindows or Motif software.

5.8. How DECnet-Plus Uses DECdts

DECnet-Plus software includes DECdts. DECdts is self-configuring, but offers a command-line management interface that allows system managers to display and update the time on any system in the network.

The DECdts application programming interface (API) allows applications (DECdns, for example) to obtain, modify, and compare timestamps. DECdts servers also accept time values from external sources through the DECdts time-provider interface; these sources can be radio receivers, data communications software, or other time services, such as NTP (Network Time Protocol). Sample time-provider programs are also supplied with the DECdts software.

DECdts consists of software components on a group of cooperating systems. In a typical DECdts implementation, a few servers supply the time to many client systems and applications through intermediaries called clerks. Clerks reside on their client systems.

Because no device can measure the exact time at a particular instant, DECdts expresses the time as an **interval** containing the correct time. DECdts clerks obtain time intervals from several servers, and compute the intersection where the intervals overlap. Clerks then adjust the system clocks of their client systems to the midpoint of the computed intersection. When clerks receive a time interval that does not intersect with the majority, they declare the nonintersecting value to be faulty. Clerks ignore faulty values when computing new times, thereby ensuring that defective server clocks do not affect clients.

DECdts provides many valuable services for computer networks running distributed applications. A summary of its major features and benefits follows:

- Correctness

  DECdts maximizes the probability that a client will receive the correct time. DECdts uses Coordinated Universal Time (UTC) as a base reference, and defines any time interval containing UTC as correct.

- Fault Tolerance

  DECdts reports faulty servers, and does not use their time values during clock synchronizations.

- Automatic Configuration

  DECdts entities use multicast messages to obtain the locations of servers in a LAN, and the Distributed Name Service (DECdns) to obtain the locations of servers in a WAN.

- Application Programming Interface

  DECdts provides an interface that allows applications to obtain, compare, and calculate UTC time values.

- Automatic Local Time Updates

  After the initial configuration of a DECnet-Plus system, DECdts sets the local system clock automatically according to the time zone rule in effect for the system.

- Local Time Translation

  When displaying time values, DECdts translates the UTC times that it uses internally into local time values.

- Quantitative Time Measurement

  DECdts uses specific measurement and manufacturer's specifications to determine the accuracy of the times reported by servers.

- NCL Management

  Through its management interface, DECdts offers Network Control Language support for controlling and monitoring the software.

- Efficiency

  Complexity is placed in the servers; network overhead is minimal.

- Monotonicity

  DECdts provides unidirectional clock adjustment to ensure that clocks do not run backwards.

5.9. DECdts Advantages

DECdts offers all the features generally provided by a time service, but it also has several unique features that enhance network performance. This section describes these DECdts features.

5.9.1. Applications Support

Operating systems and distributed applications require synchronized time measurements to coordinate their processes. DECdts synchronizes the system clocks in a network with each other, and in the presence of an external time-provider, to the UTC time standard. Any distributed application that reads the system clock (the majority of applications) needs DECdts. As the number of distributed applications and systems in a network increases, DECdts becomes increasingly vital to process coordination.

For new applications, DECdts provides an application programming interface (API) that provides routines applications can use to obtain and manipulate binary timestamps. For example, routines are available to convert the OpenVMS local time format used in DECnet-Plus for OpenVMS systems to UTC. The DECdts API supports ANSI C language constructs; you can use the same calls on VSI DECnet-Plus for OpenVMS and DECnet/OSI for UNIX systems.

5.9.2. Automatic Time Zone Changes

On DECnet-Plus systems, DECdts automatically sets the system time according to the time zone rule you specify during the initial configuration procedure. DECdts communicates by using UTC; it uses the time zone rules to translate from UTC to local time. In addition, the time zone rules can define seasonal time changes. For example, you can specify that when daylight saving time takes effect, the system clock should be updated automatically.

5.9.3. External Time-Provider Support

For most networks, it is desirable to synchronize the system clocks with the UTC time standard. Many commercial devices are available for obtaining the UTC time provided by standards organizations;these devices receive signals by short-wave radio, satellite, and telephone. If your network is larger than a single LAN, recommends that you use at least one external time-provider in combination with the DECdts software. Sample time-provider programs for use with various time-providers are supplied with the DECdts kit and are available on line. The DECdts kit also contains a time-provider interface that describes the interprocess communications between a DECdts server and a time-provider.

5.9.4. Quantitative Inaccuracy Measurement

Unlike other network time services, DECdts uses manufacturer's specifications and direct observation to determine the inaccuracy of system clocks relative to UTC. DECdts appends an inaccuracy measurement to each time value that it uses internally; this measurement takes into account cumulative clock error, communications delays, and processing delays. DECdts uses combined time and inaccuracy measurements from one or several sources to calculate the most accurate new clock settings for client systems.

5.10. How DECdts Works

DECdts has two major software components:

- Clerks
- Servers

The following sections describe each of these components and tell how they interact to provide time to client applications and to synchronize system clocks.

5.10.1. Clerks

Any system that is not a DECdts server is a DECdts **clerk**; most network systems run clerk software. Clerks are the intermediaries between clients and servers. Clerks also maintain server lists and perform the synchronization functions for DECdts client systems. Clerks send multicast queries to all servers and build server lists from the servers that respond. Clerks contact known servers for new server lists periodically, or whenever the number of servers on their lists falls below the minimum specified by the `servers required` attribute. This attribute is one of several settable DECdts characteristics available through the management interface.

After building a server list, a clerk can directly request time values from the number of servers determined by the **servers required** management parameter. The clerk then receives these time values and uses them to compute a new system time for its client system.

5.10.2. Servers

Servers provide many of the communications functions for DECdts. They provide time intervals to clerks and other servers, and advertise their presence on the LANs of which they are members.

External time providers can be connected to servers, which then propagate the precise time intervals they obtain from the time-provider throughout the network. The rest of this section describes the three types of DECdts servers.

5.10.2.1. Local Servers

A set of **local servers** reside on the same LAN and maintain their clocks by synchronizing with each other. Local servers use the IEEE 802.2 (CSMA-CD) protocol to communicate; due to the high throughput on this type of network, the skews between the local servers on the LAN are normally maintained at under 200 milliseconds. If at least one of the servers in the local set synchronizes with an accurate external time-provider, inaccuracies at each server may be even less.

Each local server advertises itself on the LAN by sending multicast messages containing its address. Each server builds lists of the other servers in the LAN by listening to the multicast messages. When a server synchronizes, it queries all the other servers on its server list for time values. Servers that do not respond are deleted from the list of servers. Local servers perform time-interval computations, adjust their clocks, and provide time values to each other for synchronization purposes. Each server must synchronize with every other server in the local set at periodic intervals.

5.10.2.2. Global Servers

WANs and many extended LANs require **global servers**. Local servers are available only to the servers and clerks in a single LAN, but global servers are available across an extended LAN or WAN. Any server can be configured as both a local and a global server. The number of global servers is usually small, but global servers have several important functions that enable DECdts to synchronize every node in the network. Global servers are necessary in the following situations:

- When a network has multiple LANs or an extended LAN with bridges that block multicast messages.
- When systems that are not on LANs have access to LANs through point-to-point links.
- When clerks or local servers cannot access the required number of local servers determined by the **servers required** management parameter.

A list of DECdts global servers is always available to network systems through DECdns, which maintains a list of the DECdts global servers. All DECdts clerk and server systems are also DECdns clients. DECdts servers and clerks also maintain their own lists of global servers, which are periodically refreshed from the directories maintained by DECdns.

Local servers and clerks request time values from global servers when they cannot obtain the number of local server responses mandated by the `servers required` attribute. Certain local servers also regularly request the time from global servers; see the following section.

5.10.2.3. Couriers

Designated local servers called **couriers** regularly communicate with global servers. Couriers are local servers that request time values from one global server at every synchronization. To ensure that all parts of the network contribute their time values to the LAN, couriers randomly select a global server from the DECdns namespace before each synchronization. Couriers use the responses of both local and global servers when synchronizing their own clocks.

For a network containing multiple LANs or point-to-point links, configure one server on each LAN or segment as a courier; this configuration ensures that various portions of the network remain synchronized and are not isolated from each other.

Using the management interface, you can also designate one or more servers to be **backup couriers**. These local servers temporarily assume courier functions in the event that no courier servers are available on the LAN. In such a case, the backup courier with the lowest numbered Ethernet address regularly synchronizes with global servers until a courier is again available.

5.11. DECdts Management

DECdts is managed with NCL. System managers can adjust system clocks, change the values of the DECdts management parameters, and add or subtract servers from the network. To aid in solving problems with system clocks, DECdts also provides event reporting that notifies system operators and managers in the rare event that a system clock is inaccurate or fails to synchronize.

Chapter 6. Naming Guidelines

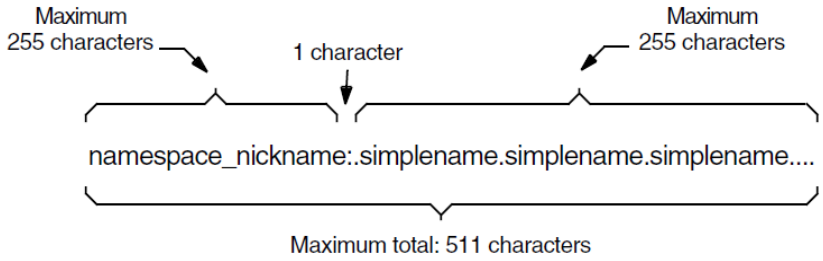
This chapter provides general guidelines for all DECnet-Plus names as well as specific guidelines for naming clearinghouses and namespaces.

6.1. General Naming Guidelines

As you choose all DECnet-Plus names, consider the following guidelines:

- A name should be meaningful and easy for its users to remember and type. Give meaning to a name by making it descriptive of the resource it names. For example, `.Aero.Dev.Project_disk` could be the name of the VAX Distributed File Service (DFS) access point used by a company's aerospace development division, and `.Sales.Topdollar` could be the name of a node in the company's sales division.
- A name should be as short as is practical without causing it to lose meaning or uniqueness.
- If you create a DECdns directory hierarchy, choose directory names that are stable and are not likely to be affected by reorganizations or changes in product strategy. It is difficult to change existing directory names, especially at high levels in the namespace. Also, a change in a directory name affects all applications that use names in that directory, as well as any users who memorized the full name of a resource or created an abbreviated name for it. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for details on how local name substitution works.
- DECdns preserves the case of names as they are entered. Lookups, however, are case sensitive so is not possible to create two names that differ only in their case. For example, requests to lookup `.MYNODE`, `.mynode`, and `.MyNode` would all produce the same result. Similarly, if someone attempted to create all three of those names, only the first attempt would succeed.
- A name can be any combination of letters, digits, and certain punctuation characters from the ISO Latin-1 character set.
- A full name, including the namespace nickname, can have as many as 511 characters, as illustrated in Figure 6.1.

Figure 6.1. Full Name Example



6.2. Guidelines for Naming Clearinghouses

If you configure your node as a DECdns server, you must supply a DECdns name for the clearinghouse that the server will manage. Use the following guidelines for naming clearinghouses:

- In networks of fewer than 50 DECdns servers, name all clearinghouses in the root directory (for example, `.Bonn_CH`). This enables you to automatically comply with two special clearinghouse

rules that protect the name service's ability to find names. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* guide for details. In networks of 50 servers or more, you should name some clearinghouses at levels below the root. See Section 7.2 for an explanation of the reasons.

- Include a suffix such as `_CH` at the end of clearinghouse names (for example, `.Paris_CH` or `.Sales_CH`) to help you easily identify clearinghouse names and avoid accidentally deleting them. Tightly controlling access to clearinghouse names also can help prevent accidental deletion.

- Avoid naming a clearinghouse based on the name of the server node where it exists. Although using the node name conveniently indicates the exact location of a clearinghouse, it can make things confusing if you later move the clearinghouse to another node. You cannot rename a clearinghouse once it is created, so does not recommend giving it a name based on the node where it currently exists. Instead, consider naming clearinghouses based on the site, city, or region where they are located. Even if you move a clearinghouse, you are not likely to move it far from the site or city where it resides.

6.3. Guidelines for Naming Namespaces

When you create a new namespace, you must supply both a namespace nickname and a clearinghouse name (the first node in a new DECdns namespace must configure as a DECdns server). Use the following guidelines for choosing a namespace nickname:

- Although nicknames can contain up to 255 characters, you should choose a short and simple namespace nickname. When you refer to a name in your clerk's default namespace, you do not need to specify the namespace nickname. However, because the nickname still appears in command output, it makes sense to keep the nickname short.

- Because a namespace has no relationship to any individual node, you should give the namespace a nickname that is meaningful to users networkwide. You might base the namespace nickname on your company's name. For example, a company called International Air Freight could have a namespace nickname of `IAF`. Choose the namespace nickname carefully; it is difficult to change. Changing the namespace nickname involves creating a new namespace and copying all the entries.

- You might decide to create a test namespace so you can become familiar with the DECdns software. If you do, give the namespace a nickname like `Test` to clearly indicate its purpose. Do not give a test namespace a nickname that you eventually want to use for a networkwide namespace. Namespace nicknames are extremely difficult to reuse once they have been established in clerk caches throughout the network.

- The following namespace nicknames have special meaning to DECnet-Plus. The nickname `LOCAL` for a node full name indicates that DECnet-Plus should look up information for the node in the Local namespace. Similarly, the nickname `DOMAIN` for a node full name indicates that DECnet-Plus should look up information for the node in the DNS/BIND namespace.

Note that the use of the underbar (or underscore) character (_), maybe restricted by certain send mail implementations. As DECnet MAIL-11 addresses may be processed by send mail on some systems (for example, UNIX), you should take this restriction into consideration.

Chapter 7. Basic DECdns Namespace Planning for DECnet-Plus

This chapter contains basic DECdns planning guidelines if you choose to plan and implement a DECdns distributed namespace now. See Section 3.2.2 if you plan to use a Local namespace.

If you plan to use a DECdns distributed namespace, DECdns is a networkwide service that allows users and applications to assign unique names to such resources as nodes, disks, and files, and then use these resources without knowing their physical location. If your network consists of about 100 nodes and you do not expect it to increase over time, this chapter provides the necessary information to plan for DECdns.

The basic DECdns planning steps are as follows:

1. Decide whether you should have multiple DECdns namespaces.
2. Choose a namespace nickname and establish a general naming policy.
3. Determine whether you need a directory hierarchy. If so, plan the directory structure.
4. Plan an access control policy.
5. Plan how to replicate directories.
6. Select DECdns server nodes.

7.1. Step 1: Should You Have Multiple DECdns Namespaces?

VSI recommends using a single namespace for your entire network. However, if your organization has special requirements that justify the use of multiple namespaces, there is nothing to prevent you from creating them. They can coexist without harm to each other.

If multiple namespaces do exist, they are separate and do not share information. You cannot replicate data across namespaces. Users can refer to a name in a namespace other than their own by including the namespace nickname in the full name reference.

Using multiple namespaces might be justified if you have two or more unconnected networks that you plan to always keep separate, or if your organization plans to acquire or connect with separate networks that have existing namespaces. Multiple namespaces might also be a solution for a company with a strong tradition of separate, autonomous departments that would find it difficult to plan a single, companywide namespace.

Another valid reason for multiple namespaces is if you decide to create a test namespace for the purpose of becoming familiar with the DECdns software. In that case, give the namespace a nickname like Test to clearly indicate its purpose. Do not give a test namespace a nickname that you eventually want to use for a networkwide production namespace. Namespace nicknames are difficult to reuse once they have been established in clerk caches throughout the network.

Creating multiple namespaces is not a way to separate names for security purposes. The access control capabilities of DECdns are sufficient to protect names from people and applications that should not see or use them. The following are some additional arguments against using multiple namespaces:

- They make name references less convenient.

  In a single-namespace network, you can define a default namespace nickname and then no longer have to use it in name references. In a multiple-namespace network, references to a name outside of the default namespace must include the namespace nickname.

- They increase management complexity and overhead.

  Multiple namespaces involve additional overhead in tracking and maintaining entries in each namespace. In particular, if you store node information in more than one namespace, and you want the nodes in different namespaces to be able to communicate, you greatly increase the complexity of creating and managing node object entries and node soft links. You also increase the complexity of managing access control.

  If there is a need for multiple namespaces in your network, yet you want all of the nodes in each namespace to be able to communicate, you must run `decnet_register` once for each namespace. If you use a Phase IV node registration script that the tool creates, you must edit the data files that the script uses. You will also need to manually maintain and update some of the node information in each namespace. For details, see Section 2.4.2.

What To Do If a Namespace Already Exists

One or more namespaces might already exist in your network as a result of client applications that use an earlier version of DECdns, called VAX Distributed Name Service (DNS) Version 1. DNS Version 1 and DECdns can interoperate in the same namespace, with a few restrictions and considerations. An appendix in the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* contains details on interoperability.

If you have one or more existing Version 1 namespaces, plan to configure the first DECnet-Plus node in one of them. Eventually you should create a Version 2 namespace and merge the other namespaces into it. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for details on how to merge namespaces.

7.2. Step 2: Plan a Naming Policy

Before you configure the first DECnet-Plus node, choose a namespace nickname and establish a naming convention for clearinghouses, directories, and object entries. Chapter 6 contains naming rules and guidelines.

If you decide to create a directory hierarchy, your naming policy can be influenced by the directory structure that you plan and by the way in which names will be used and managed. Your directory structure and naming policy might therefore evolve at the same time.

7.3. Step 3: Should You Create a Directory Hierarchy?

Small networks should not need to create a hierarchy of directories other than those required for use by Session Control and DECdts. As long as you do not anticipate major growth beyond your network's current size, you can name all of your clearinghouses and DECnet-Plus nodes in the root directory.

Naming everything in the root keeps names short and simple, and it eliminates the planning and management tasks associated with multiple directories. However, it means every node name in your

namespace will be in one directory. If you want to protect some names from general access, you need to grant access to individual names rather than granting access to the entire contents of the root directory. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for a detailed description of access control.

If your network meets any of the following conditions, read Chapter 8 for guidelines on planning a directory structure:

- It is larger than about 100 nodes.
- It is smaller than 100 nodes now but you expect growth over time.
- It includes OpenVMS systems already using DNS Version 1, and you want to plan a unified namespace that contains more than just node entries.

7.4. Step 4: Plan an Access Control Policy

Before you configure your first DECnet-Plus node, you should plan an overall node-name administration policy. This section presents guidelines for controlling access to the directories containing node object entries, node synonym soft links, and backtranslation soft links. If your namespace will contain entries other than node names, you should plan and implement an access control policy that includes those entries as well.

Note

The *VSI DECnet-Plus for OpenVMS DECdns Management Guide* has details on how access control works and shows sample commands for implementing varying degrees of security in your namespace. Read the access control chapter before you configure the first DECnet-Plus node. VSI recommends that you implement an access control policy immediately after you create a new namespace.

The `decnet_register` tool automatically grants certain access rights for directories, object entries, and soft links that it creates. Depending on the degree of security and centralization you want for node-name management, you can modify the default access control scheme. By default, the tool assigns the following access:

- Every user on a node registered in this namespace gets read access to all directories, object entries, and soft links that the tool creates.
- Every user on a node registered in this namespace gets read and write access to the children of the `.DNA_BackTranslation` directory and their contents. Write access is necessary so a node can automatically keep its own address data up-to-date.
- The `.DNA_Registrar` group gets complete access to all directories, object entries, and soft links that the tool creates.
- Session Control on each node gets the access rights it needs to maintain that node's DNA$Towers, synonym, and backtranslation data.

The following three suggested policies offer varying degrees of access control over node data in the namespace.

- Secure, centralized: The default policy created by `decnet_register` grants the highest degree of centralized control over node names and soft links. Under this policy, only members of the `.DNA_Registrar` group are able to register nodes in the namespace. Only Session Control

and members of the `.DNA_Registrar` group are able to modify node data. To complete implementation of this policy, all you have to do is add the names of designated node-name administrators to the `.DNA_Registrar` group (the group is created empty).

- Moderately secure: To maintain centralized control over the namespace but allow people other than the `.DNA_Registrar` group to register and modify node information, you can grant access as needed to specific individuals and remove the access when the task is done. This policy means users still must contact a member of the `.DNA_Registrar` group when they want access to the namespace. However, it allows the `.DNA_Registrar` group to temporarily delegate node-name registration and management tasks.
- Accessible: To completely decentralize, you can change the default directory access control to grant read and write access to directories that are to contain node names. This allows nodes to autoregister themselves. It also allows any user to register a node in those directories without intervention by a namespace administrator. The `decnet_register` tool provides a convenient means for allowing or disallowing node autoregistration.

Note

The default access control policy established by `decnet_register` applies only to directories created with that tool. The tool always creates the `.DNA_NodeSynonym` and `.DNA_BackTranslation` directories. Directories to contain DECnet-Plus node object entries can be created with either the tool or the DECdns Control Program. By default, the control program grants full access for the creator of a directory. Therefore, on directories created with the control program, you must add or modify access if you want it to parallel the access granted by `decnet_register`.

7.5. Step 5: Plan the Replication of Directories

A basic DECdns guideline is to create at least two replicas of every directory (including the original, or master replica). Replication ensures an alternate source of information in the event that one of the replicas is temporarily unavailable. Replication also creates alive backup of DECdns data in the event that a clearinghouse becomes permanently unavailable.

Replicating data requires at least two clearinghouses, and thus two servers. In very small namespaces, you can configure only one server and rely on traditional operating system backups to preserve DECdns information. However, backups are not a good way to preserve data in namespaces with more than one clearinghouse. If you restore a clearinghouse whose data is replicated else wherein the namespace, unexpected and confusing results may occur. For example, recently created names can disappear, recently deleted names can reappear, and recent modifications can be reversed. Replication provides reliable, real-time backup of information and protects you against disk failures and common problems that may occur. Backups of the master replica protect you against deleting object entries. Therefore, the best way to back up DECdns data is to create at least two replicas of every directory.

DECdns creates the root directory automatically when you configure the first server and clearinghouse in the namespace. The root also is replicated automatically at any subsequent server whose clearinghouse you name in the root.

After `decnet_register` creates and populates the node directories, you should replicate them for availability and backup. Because DECnet-Plus systems are the only ones that use the directories created by the tool, you can replicate them at the same pace at which you migrate nodes to DECnet-Plus. In fact, if you do not have an existing DNS Version 1 namespace, replication must wait until you configure at least one other DECdns server. In small networks, two replicas of every directory should be sufficient. See Chapter 8 for additional replication guidelines for large networks.

7.6. Step 6: Select DECdns Servers

Server nodes are the systems that store and maintain the clearinghouses in your namespace. Your main goals in choosing DECdns servers should be to achieve availability of data, reliability, and optimum performance. Study your network and keep in mind the following guidelines to help you achieve those goals:

- Choose stable nodes.

  DECdns servers should be stable systems that experience minimal downtime and restart quickly. It is important that a DECdns server be one of the first systems available in the network, because client applications all over the network are limited to the information in clerk caches until a DECdns server is available.

- Use reliable network connections.

  Reliable connections between DECdns servers are important because, during skulks, DECdns must be able to contact all servers that maintain a replica of the directory involved. Reliable connections between clerks and servers are not essential after a clerk develops a cache of frequently used information. However, they do increase the chances for success on initial lookups and on rare long-distance lookups.

- Estimate disk space, CPU, and memory requirements.

  In small namespaces, the impact of DECdns on timesharing systems should be minimal. The impact depends partly on the number of node object entries and soft links in the namespace, the frequency of use of those names, and the load placed on the system by other applications. If you are concerned with DECdns capacity requirements, see Section 8.3 for additional guidelines.

Unless you already have a DNS Version 1 namespace, the first system in your network to make the transition to DECnet-Plus must be a DECdns server. Many additional factors can influence the choice of the first DECnet-Plus system (see Chapter 2 for details).

7.6.1. Server Placement Guidelines for LANs and Extended LANs

To ensure access to data in the event a server becomes unavailable and to distribute lookup load, consider using at least two servers for every thousand nodes. If a server is connected to other servers through high-speed, reliable links, you might be able to use just one server on a LAN. Factors influencing your decision can include the expected lookup load and how you want to distribute it, and the capacity of the systems that you plan to use as DECdns servers.

On extended LANs, consider the reliability of the bridge connecting LANs. If the bridge is frequently unavailable, or if it filters DECdns multicasts, you might want two servers on each side of the bridge. (DECdns servers use multicasts to advertise their existence to clerks and other servers, so LAN bridges that filter multicasts prevent the automatic passage of this information from one LAN to another.) However, if the bridge is reliable and does not filter multicasts, then one server on each side of the bridge is adequate.

Note that with the LAN Bridge 200 product and Remote Bridge Management Software, you can control filtering to allow DECdns multicast IDs to traverse the bridge. The following are the DECdns multicast IDs:

NSAdvertisement	09-00-2B-02-01-00
NSSolicitation	09-00-2B-02-01-01

7.6.2. Server Placement Guidelines for Sites Connected by a WAN

When planning the placement of DECdns servers in a wide area network (WAN) environment, try to avoid connections through WAN links that are not usually stable. Place DECdns servers so that most systems can access at least one server even if a WAN connection is unavailable.

At small sites connected to the rest of the network through a WAN, a DECdns server is not necessary if the small site only occasionally uses resources on the other side of the WAN link. For example, if users at a small site sometimes contact nodes at the company's headquarters, it is probably sufficient to store the node names at headquarters, and it is not necessary to configure a DECdns server at the small site. Remember that once clerks at the small site cache frequently used names, they will rarely need to cross a WAN link for lookups anyway.

On the other hand, if the small site has many DECdns names referring to object entries at the site, it might make sense to configure a server there. Consider storing the master replica at the small site's server if you expect users there to make frequent changes to the names. Doing so further reduces WAN traffic and improves performance.

Consider that when you configure clerks at sites connected by a WAN link, you must manually enter the address of a server that the clerk can contact. This is less convenient than configuring clerks on a LAN, where the clerk automatically receives advertisements from servers on the LAN and caches their addresses.

Chapter 8. Advanced DECdns Namespace Planning

If your network is large, or if you have a small network now but want to plan ahead for growth, use the additional guidelines in this chapter to plan your distributed namespace. These guidelines also are useful for planning a namespace into which a DNS Version 1 namespace and its entries will eventually merge. The chapter offers guidelines in three main areas:

- Planning a directory hierarchy
- Replicating directories
- Calculating server capacity needs

8.1. Planning a Directory Hierarchy

VSI recommends a multilevel directory structure for large and growing networks. Multiple levels of directories offer the following advantages over a root-only namespace:

- Greater opportunity to distribute information in the network, decreasing the processing load on any one system.
- Ability to separate object entries based on where they are used, their frequency of use, the people who use them, or whatever criteria suit your needs.
- Decreased probability of duplicate names (for example, `.Eng.Aero.Dev_disk` and `.Eng.Plastics.Dev_disk` are unique).
- Ease of delegating access control and administrative responsibility for subsets of directories and object entries.

Overall, try to create no more than three or four directory levels. In very large networks, people at individual sites might create and own lower-level directories that are minimally replicated (for instance, not beyond a LAN) and used mainly by a specific group of local users. In such cases, users can create as many levels of directories as they deem necessary and convenient to manage.

However, remember that every directory requires additional network resources for skulking and increases administrative overhead in areas such as access control and replication. Also, a long directory path results in names that are hard to remember and type. Do not create empty directory levels to serve as place markers in names.

8.1.1. Consider Geographic and Functional Directory Names

You can design a directory structure based on functional areas of your organization, geographic areas of your network, or a mixture of both. Choose whatever scheme best suits your organization's needs and preferences. The decision should be influenced partly by how you want to populate and replicate the directories. For example, you could populate a geographic directory with names used mainly by a group of users concentrated in one geographic area, and replicate the directory close to those users. Similarly, you could populate a functional directory with names used mainly by people in a particular branch of an organization.

Functional directory names can be useful if your organization:

- Has a well-established and stable functional structure that is not likely to change much over time.
- Consists of several independently functioning units that each use their own resources.

You can use your company's organization chart as a template for designing functional directories. Remember to choose names that are not likely to change; derive directory names from a department's business function, not its current title. Some sample functional directory names are `.Sales`, `.Admin`, `.Mfg`, and `.Eng`, for storing names used by the Sales, Administrative, Manufacturing, and Engineering branches of an organization.

In addition to or instead of creating functional directories, you can create geographic directories. For example, you could name upper-level directories `.NY`, `.Paris`, and `.Geneva`, with lower-level directories based on site codes or some other more specific geographic name. The following are reasons for using geographic directory names:

- To organize names that are used mainly by a concentration of users in certain geographic areas.
- To avoid skulks of the root directory across wide geographic areas.

Every clearinghouse named in the root directory must store a replica of the root; see Section 8.2.2 for details. By naming clearinghouses in geographic directories below the root, you can give an indication of their location and also help reduce skulking of the root directory across long distances.

If node names are traditionally managed by site or geographic area, you might group them in geographic directories. If a node communicates most frequently with other nodes in its geographic area, this scheme would also result in more efficient lookups, provided that you replicate the geographic directory at servers in the location that the directory covers. Any other names that are used or managed primarily by location can be stored in geographic directories as well.

8.1.2. Plan Access Along with Directory Structure

While you plan the structure and contents of directories, consider an overall name administration scheme. Your plan for who will use and manage names can have a strong influence on directory structure. For example, the root and other upper-level directories should be stable, widely replicated, and limited in content, and only a few trusted people should be able to create or modify their contents.

As part of namespace creation, the DECnet configuration procedure creates an access control group called `.DNS_Admin`, which you can use to control access to the namespace. You can limit access to top-level directories by granting only the `.DNS_Admin` group the more powerful access rights (write, delete, and control) to those directories. You also can set up access control entries (ACEs) that propagate down to newly created directories and their contents, giving the `.DNS_Admin` group a window into activities or problems that occur anywhere in the namespace.

More flexible administration and access control is possible at lower levels of the namespace hierarchy, where directories might be created, controlled, and written to by a limited number of local users. The content and structure of these directories can be determined by the people who will use and manage them.

Keep in mind that it is possible to share a directory. Because access control is assignable to both object entries and directories, several different client applications or user groups can share a directory without being able to change or delete each other's names. A namespace administrator or server manager with access to the entire contents of a shared directory should monitor it routinely for growth.

8.1.3. Other Directory Planning Tips

The following are some additional guidelines for planning directories and their contents:

- Keep the root small and stable.

  The root directory will probably be one of the most widely replicated directories in the namespace, especially if you name most or all of your clearinghouses in the root (when a clearinghouse is named in the root directory, the root is automatically replicated in that clearinghouse). To minimize overhead from skulks of the root directory, limit the contents of the root to object entries that will not change often.

  The root should contain a maximum of 100 widely used object entries (including clearinghouse object entries and node object entries) and should not be cluttered with names used by only a small number of people. This restriction is less important for networks with only a few servers. Consider a root directory that contains 250 object entries, replicated in a namespace that contains only two servers. The impact of skulking two replicas of the directory would be less than the impact of skulking 20 replicas of that same directory. Note also that the recommended limit does not apply to the number of child directories the root can have.

- Limit other directories to a manageable size.

  Directories have a practical size limit based on manageability. Factors to consider include the following:

  - How easy is it to assign and control access to a directory's contents? If you want to implement a simple access control scheme (for example, assigning default access to the entire contents of a directory), your decision may limit a directory's contents, and thus its size.

  - How widely must a directory be replicated? A directory containing many names may need to be replicated in several places to locate the names close to their users. Update propagation and skulking on large, widely replicated directories adds overhead, especially across WAN links. For example, if you have three sites and 1,000 object entries, and all of the sites need those object entries, keep them in one directory. On the other hand, if specific object entries are used only by users at one site, you can save overhead by locating those object entries in a directory at that site.

- Anticipate growth.

  Try to design a namespace that allows horizontal expansion (more directories directly under the root) and avoids vertical expansion (more levels of directories). This design will help limit the number of directory levels and make the namespace easier to manage, even as it grows.

8.2. Replicating Directories in a Large Network

Your strategy for replicating directories is especially important in the initial stages of namespace operation. Once a clerk establishes a cache of frequently used names, the clerk will rarely need to locate a replica for the information it needs. However, well-planned replication of directories can make the clerk's process of learning about names in a large network easier and more efficient. The number and size of replicas that you plan in the namespace can also affect your choice of servers.

When planning replication in large networks, consider the following guidelines to enhance DECdns performance:

- Ensure availability.

  Every directory should have at least two replicas (including the master replica). Replication ensures an alternate source of information in the event that one of the replicas is temporarily unavailable.

Very small namespaces with one DECdns server can use traditional operating system backups to preserve DECdns information. However, backups are not a good way to preserve data in namespaces with more than one clearinghouse. Multiple replicas are especially important in these namespaces because you cannot depend on traditional operating-system backup methods to preserve replicated data. Data restored from backups might contain outdated timestamps, which cause replicas to lose synchronization and can cause DECdns to return outdated information. Replicating is the most reliable way to create a backup of data in a clearinghouse.

- Distribute the lookup load.

  Identify directories that will get the most use for lookups and updates, and replicate them on several different servers. This helps to distribute the lookup load.

- Locate a replica where its users are.

  If you design a directory so that its most frequent users are widely dispersed geographically, replicate it accordingly. Alternatively, you can avoid wide replication of a directory by populating it with names used mainly by a group of users who are concentrated in one geographic area.

- Locate a name close to the resource it describes.

  Create an object name in a directory that is replicated close to (for example, on the same LAN as) the resource it describes. Because a resource is most often used by people close to it, replicating its name nearby can save network resources and make lookups more efficient. Usually, it is practical to limit the replication of lower-level directories to clearinghouses that are closest to where they will be used.

- Help DECdns find a name efficiently.

  Replicate the root directory and other high-level directories widely, because the clerk must often find the root and work down from it in initial attempts to look up names. Also, you can reduce WAN traffic by placing the root and other high-level directories at sites that frequently refer to names stored across a WAN link.

8.2.1. Replicating the DECnet-Plus Directories

The following are specific guidelines for replicating the directories that contain node object entries, node synonyms, and backtranslation soft links:

- If a directory contains node object entries, locate at least one replica (preferably the master)on the same LAN as the nodes the entries describe.
- Locate at least one replica of the `.DNA_NodeSynonym` directory on every LAN that includes DECnet-Plus nodes.
- Locate the master replica of each `.DNA_BackTranslation` area directory in the area it is named after, if that area contains DECnet-Plus nodes. Also replicate an area directory in other areas likely to communicate most frequently with the nodes whose addresses it contains.
- In a solely WAN environment, try to replicate every directory so it can be reached reliably by any DECnet-Plus system that needs it.

8.2.2. How Clearinghouse Names Affect Replication

To ensure its ability to find names, DECdns enforces some rules regarding clearinghouses and their contents. The *VSI DECnet-Plus for OpenVMS DECdns Management Guide* contains an appendix that

explains the rules in detail. For planning purposes, the main rule to remember is that whenever a clearinghouse is named in the root directory, the root is automatically replicated in that clearinghouse. In a network with more than 50 DECdns servers, naming all clearinghouses in the root could, therefore, cause replication of the root beyond a practical limit. Naming some clearinghouses in directories below the root helps to avoid this problem.

For instance, if you have 50 clearinghouses in a network and you name them all in the root, the root directory is replicated in at least 50 places to comply with the rule. Then, every time DECdns skulks the root directory it must be able to contact and process all 50 replicas. The greater the number of replicas, the greater the demand on system and network resources, and the greater the chance for skulks to fail. Skulks also can fail if a directory is replicated across an unreliable WAN link.

Therefore, if your namespace will have more than 50 clearinghouses, you can help avoid problems with replicating and skulking the root by naming some clearinghouses in lower-level directories. Naming a clearinghouse in a directory other than the root eliminates the requirement that a replica of the root be stored there, thus reducing the total number of root replicas in the network.

If your namespace is easily divisible by geographic area, you could create some geographic directories under the root and name clearinghouses in those directories; for example, `.Tokyo.Site1_CH`, `.Wash.Seattle_CH`, or `.Paris.Branch3_CH`. The geographic directories in each clearinghouse could also contain data used primarily by the people and applications in that geographic area, and the work of skulking replicas would be more evenly distributed than if all of the data were in the root.

8.3. DECdns Server Capacity Planning

To determine whether a node has the capacity to be a DECdns server, consider several factors, including the system's disk space and processing power, the demand that other applications already place on the system, and the demand you expect from DECdns.

If you expect high DECdns lookup and update demand, consider using a dedicated server. VSI recommends using dedicated servers for large networks, because a dedicated system generally boots faster than a timesharing system, and you can select it specifically for optimum DECdns server performance. For example, you will probably have to increase the swap space on a server with a clearinghouse that contains many directory replicas or replicas with many entries. Keep in mind that a few large, dedicated servers can perform better than many small timesharing systems.

To estimate capacity requirements for a server, assess the number of directory replicas to be stored at each server and the number of entries in those replicas. Table 8.1 contains estimates of memory usage for ACEs, directories, node object entries, and soft links.

Table 8.1. DECdns Memory Requirements

Item	Memory
ACE	60 bytes
Node object entry	750 bytes
Node synonym soft link	520 bytes
Backtranslation soft link	520 bytes
Directory	4700 bytes

The numbers in the table are based on the following assumptions:

- Node name length is 30 characters or fewer.

- ACE estimates are based on a fixed requirement of about 40 bytes plus a variable requirement for the length of the principal specification. The 60-byte total is derived from an estimated average principal length of 20 bytes (for example, `.Sales.Node01.Miller` is a principal requiring 20 bytes).

If you plan node names of 30 characters or longer, or if you expect principal names to be longer or shorter than 20 characters, adjust the capacity estimates accordingly. Also remember that a lengthy list of ACEs can quickly increase the space requirements of a directory, object entry, or soft link.

One way to conserve space on ACEs is to use the `.DNS_Admin namespace` administrator group, created automatically as a part of namespace creation. You also can create and use similar administration groups for each directory you create. Then each entry needs only one ACE per group, and you can control access to the entry by adding and removing members of the group. In addition to saving space, the use of groups is easier and more efficient than adding and removing individual ACEs. See Chapter 9 and the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for a complete discussion of the use of groups.

Add a 30 percent margin of safety after figuring the requirements for nodes and directories. This margin is necessary because of the way storage is allocated for object entries in the database. Even if only one object entry exists, a fixed space is allocated for it that is more than the entry actually requires.

Now that you know the amount of space your clearinghouse requires, you need to make sure that your server's disk, memory, and (on ULTRIX or UNIX systems only) swap space can support it. The server loads the entire clearinghouse into memory. You can obtain the best performance by providing enough physical memory to contain the entire clearinghouse. Very large ULTRIX or UNIX servers need to be configured for additional swap space to support the virtual memory use of the server. Similarly, OpenVMS VAX servers require sufficient paging file space to support the server. The OpenVMS VAX server startup uses the `VIRTUALPAGECNT` SYSGEN parameter to determine its maximum page file quota. It also uses `WSMAX` as its working set extent and `WSMAX*.3` for the working set quota.

On ULTRIX, UNIX, or OpenVMS VAX servers, the disk with the clearinghouse files themselves (the `/var` partition or the `[DNS$SERVER]` directory) must be able to hold two copies of the clearinghouse.

Capacity Planning Example

The following hypothetical example is a DECdns capacity plan for a small organization, ABC Company, whose network consists of 30 nodes. The nodes are all on the same LAN and in DECnet area 1, and the planners have chosen to use a default initial domain part (IDP) of 49.

The namespace planners decided to configure two servers, each containing the same directory replicas. They decided to create no additional directories other than those required by DECnet-Plus and DECdts. So each clearinghouse would contain replicas of the following six directories:

- . (the root)
- `.DNA_NodeSynonym`
- `.DNA_BackTranslation`
- `.DNA_BackTranslation.%X49`
- `.DNA_BackTranslation.%X49.%X0001`
- `.DTSS_GlobalTimeServers`

Given that all of their nodes would be named in the root, and that most of their node names would not exceed 10 or 12 characters, the planners thought that the estimate of 20 bytes required for a principal

was high. They decided to reduce the estimate for principals to 15 bytes, thus reducing the total storage requirements for ACEs to 55 bytes each. They estimated an average of four ACEs on each directory and entry in the namespace. They also estimated that the server needs an additional 134,000 bytes for swap space for ULTRIX. Based on these considerations, the ABC Company planners produced the following node and directory capacity worksheets:

ABC Company
Node Requirements

Node object entry	750 bytes
+ Node synonym soft link	520 bytes
+ Backtranslation soft link	520 bytes
Per node subtotal	1790 bytes
+ ACE consumption per node	660 bytes
Per node total	2450 bytes
x number of nodes	30
Total node consumption	73,500 bytes

ACE consumption	55 bytes
x est. ACEs per node entry	4
x 3 entries per node (1 object, 2 soft links)	220 bytes 3
Total ACE consumption per node	660 bytes

ABC Company
Directory Requirements

Directory consumption	4700 bytes
+ ACE consumption per directory	220 bytes
Per directory subtotal	4920 bytes
x number of directories	6
Total directory consumption	29,520 bytes

ACE consumption	55 bytes
x est. ACEs per directory	4
Total ACE consumption per directory	220 bytes

The planners then added the node total and the directory total and factored in a 30 percent margin of safety, as shown in the following worksheet:

ABC Company
Total Capacity Requirements

Total node consumption	73,500 bytes
+ Total directory consumption	29,520 bytes
Subtotal	103,020 bytes
+ 30% safety margin	30,906 bytes
Total capacity requirements	133,926 bytes

As a final step in capacity planning for ULTRIX or UNIX systems, the planners estimated that the server needs an extra 134,000 bytes for swap space and the clearinghouse database needs 268,000 bytes in the

`/usr/var/dss/dns` partition. For OpenVMS VAX systems, the planners estimated that they need a minimum of 525 blocks in the `DNS$SERVER` default directory.

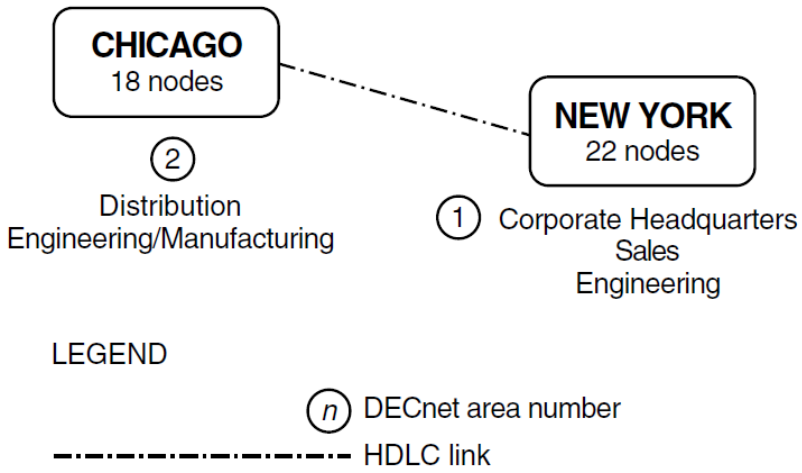
Chapter 9. DECdns Namespace Design Example

International Air Freight (IAF) Corporation is a hypothetical air freight company that does business in the United States, Europe, and Japan. The company plans to expand into other countries in the near future and anticipates considerable growth in its network.

The company consists of a New York office and a Chicago office. The New York office is corporate headquarters and is also the base for IAF's sales organization. A small engineering group in New York develops and maintains software applications specific to the company's business and distribution needs. Chicago is the hub of IAF's distribution operations. Also in Chicago is a small engineering and manufacturing group. This group designs and manufactures the packaging that IAF uses to transport freight.

The IAF network currently consists of 22 DECnet nodes on a local area network (LAN) in New York and 18 DECnet nodes on a LAN in Chicago. The two LANs are connected by DEC WAN routers that are communicating by means of a High-level Data Link Control (HDLC) link. DECnet-Plus software is in use, and the hardware includes workstations, personal computers, and large VAX systems. DECdns servers are running on DECnet-Plus for ULTRIX and DECnet-Plus for OpenVMS VAX systems. Figure 9.1 is a representation of the IAF network.

Figure 9.1. The IAF Network



IAF's introduction to DECdns came when the company decided to upgrade its network to DECnet Phase V. Network administrators learned that DECnet-Plus Session Control can use DECdns to store and look up node names. They saw the ability to distribute nodenames, combined with the automatic updating capability of DECdns, as a time and resource saver. In the beginning, DECnet would be the primary user of DECdns and node names would be the primary entries in the namespace. The Distributed Time Service (DECdts), another component of DECnet-Plus, would use DECdns to store the names of global servers that synchronize system clocks in the network.

When planning for DECdns, IAF anticipated future use by applications in addition to those that would be in place immediately after the transition to DECnet-Plus. Other client applications that IAF planned to use included VAX Distributed File Service (DFS), VAX Notes, and Remote System Manager (RSM), products that use Version 1 of the name service.

IAF engineers also planned to develop their own client applications specific to the company's distribution, manufacturing, and administrative needs. For instance, IAF had areal-time application that

collected data on the location and flight times of air carriers and tracked the status of freight at each of the company's distribution sites. The application used a message bus for task-to-task communications and message queuing. The message bus, in turn, would be adapted to store the names and addresses of its message queues in DECdns.

IAF assembled a planning committee to plan the transition from DECnet Phase IV to DECnet-Plus and to plan the DECdns namespace. The committee decided to make a staged DECnet transition. First, they would upgrade a portion of their LAN in New York and configure two DECdns servers there. Six to eight months later, they would begin to upgrade the Chicago LAN, configuring two DECdns servers there as well. Transition of the remaining nodes in New York would be gradual, as time and resources allowed.

This chapter discusses decisions the IAF planners made in designing and configuring their namespace. It also describes some of the business changes the company went through after initially setting up its namespace, and explains the effects those changes had on the namespace. The discussion refers to other manuals or other chapters in this manual for more information on some of the activities described.

9.1. Part 1: Planning and Configuring the Namespace

The planning committee chose a namespace nickname based on the initials that stand for the company's name: `IAF`. For naming clearinghouses, the committee decided to identify a clearinghouse by the city where it was located and adopted the recommended convention of adding a `_CH` suffix to a name. They knew they would have fewer than 50 clearinghouses, so they also followed the recommendation for naming all clearinghouses in the root directory. For example, clearinghouses in New York would be called `.NY1_CH` and `.NY2_CH`.

Next, the committee decided to assess the size and contents of the IAF namespace to determine whether they needed to create a directory structure. They knew that, for DECnet-Plus, every node in the network would have three entries in the namespace: an object entry and two soft links. After considering the needs of other applications, the committee produced the following list of projected namespace contents:

- 40 node names (for the 22 nodes in New York and 18 in Chicago)
- 40 node synonym soft links
- 40 node address backtranslation soft links
- 15 VAX Notes conference names
- 4 RSM server names and 35 client names
- 2 DECdts server names
- 4 DFS access point names
- 10 message bus queue names
- Miscellaneous names created by IAF-written applications, number to be determined

It was clear that node names and soft links would have the most significant impact on the namespace; the number of names created by other applications would be minimal. Even the impact of node names and soft links would be minimal in a network the size of IAF's. However, the company and its network

were expected to grow. Therefore, the planning committee decided to prepare for future expansion by designing a directory hierarchy.

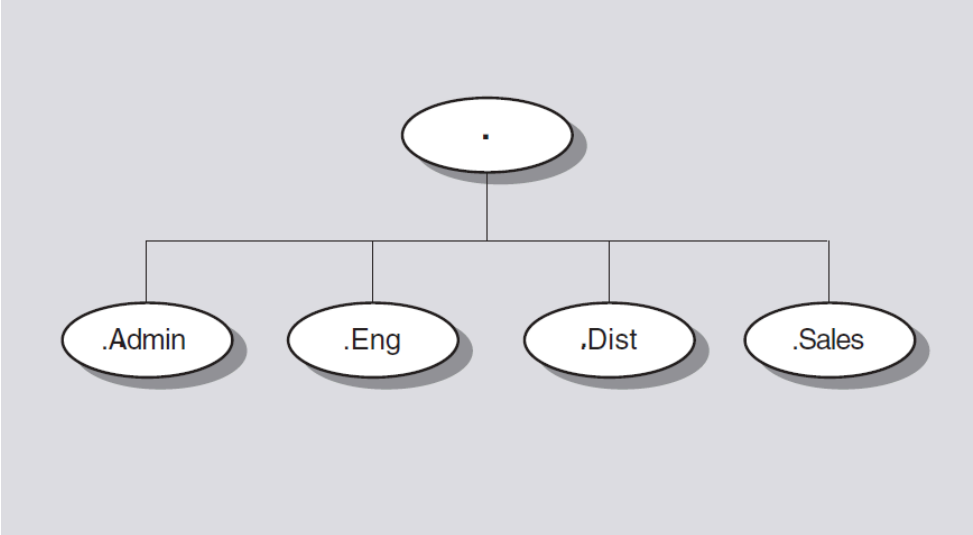
9.1.1. Designing the Directory Structure

The committee decided that creating several directories, all one level below the root, would be more than sufficient to handle the size of the IAF namespace. They wanted to avoid the longer names and the additional management tasks associated with multiple directory levels. They also felt confident that it would be a long time before any one directory contained more than 5000 names: a number that they considered a practical directory size limit.

The planners decided on a functional directory naming scheme. The decision was easy, because IAF is a functionally oriented organization. Resources, including nodes, are allocated and managed on a departmental basis, regardless of geographic location. For that reason, the company's organization chart could serve as a template both for the directory structure and for designing an access control policy.

Using the organization chart, the committee planned functional directories for administration, engineering, distribution, and sales, resulting in the hierarchy shown in Figure 9.2.

Figure 9.2. IAF Namespace Hierarchy



9.1.2. Planning Access Control

The namespace access control policy would be closely tied to the directory structure. The committee decided to implement a policy of world read and test access on the entire namespace, but to limit other access rights for each functional directory to the main users and managers of the names in those directories. They would use DECdns access control groups to implement this policy.

The `.DNS_Admin` group, a namespacewide administrative group, would consist of the network manager, who was the designated namespace administrator, and other staff members from IAF's Network Control Center. The members of the `.DNS_Admin` group would be the only people with full access to the root directory. The planners also decided to give the group access that propagates down to all subsequent directories and their contents. The propagating access would allow the `.DNS_Admin` group members to monitor any changes or problems in the namespace.

The `.DNA_Registrar` group is an access control group created during configuration of the first DECnet-Plus node in the namespace. Its purpose is to contain the names of people responsible for

managing node object entries and soft links. The planners had read the DECnet transition documentation and decided to use this group as well. They decided to make the `.DNS_Admin` group a member of the `.DNA_Registrar` group, automatically granting the namespacewide administrators access to all node-related entries in the namespace. Additionally, the `.DNA_Registrar` group would contain the names of people at each site who were traditionally responsible for assigning and monitoring node names.

In addition to the `.DNS_Admin` and `.DNA_Registrar` groups, the planners decided to create separate access control groups to manage each functional directory. The directory management groups would be named `.Admin.Dir_Admin, .Sales.Dir_Admin,.Eng.Dir_Admin`, and `.Mfg.Dir_Admin`. Each group would have full access to the contents of the directory it was associated with, and would contain the names of the current managers of that directory. With groups as the main method of granting access control, the namespacewide administrative group could simply add and remove members of a group instead of creating and deleting individual ACEs whenever management responsibility for a directory changed.

9.1.3. Assessing Directory Contents

Based on their decisions, the planners outlined the contents of each directory as follows:

- Root directory (.)
  - Clearinghouse names (object entries).
  - The `.DNS_Admin` namespace administrator group.
  - The `.DNA_Registrar` node entry administrator group.
- `.Admin` directory
  - Names of DECnet-Plus nodes used by administrative groups, including the executive office, personnel, finance, accounting, payroll, and management information systems.
  - Names of RSM client and server systems.
  - The `.Admin.Dir_Admin` group.
- `.Sales` directory
  - Names of DECnet-Plus nodes used by the sales groups.
  - Names of RSM client and server systems.
  - Names of sales-related Notes conferences.
  - Names translating to file specifications of price databases, to be created and used by an IAF-developed application.
  - The `.Sales.Dir_Admin` group.
- `.Eng` directory
  - Names of DECnet-Plus nodes used by the New York engineering group. Names of nodes from the Chicago engineering and manufacturing groups would also be stored in this directory, but not until 8 months to a year after the New York LAN began the transition to DECnet-Plus.

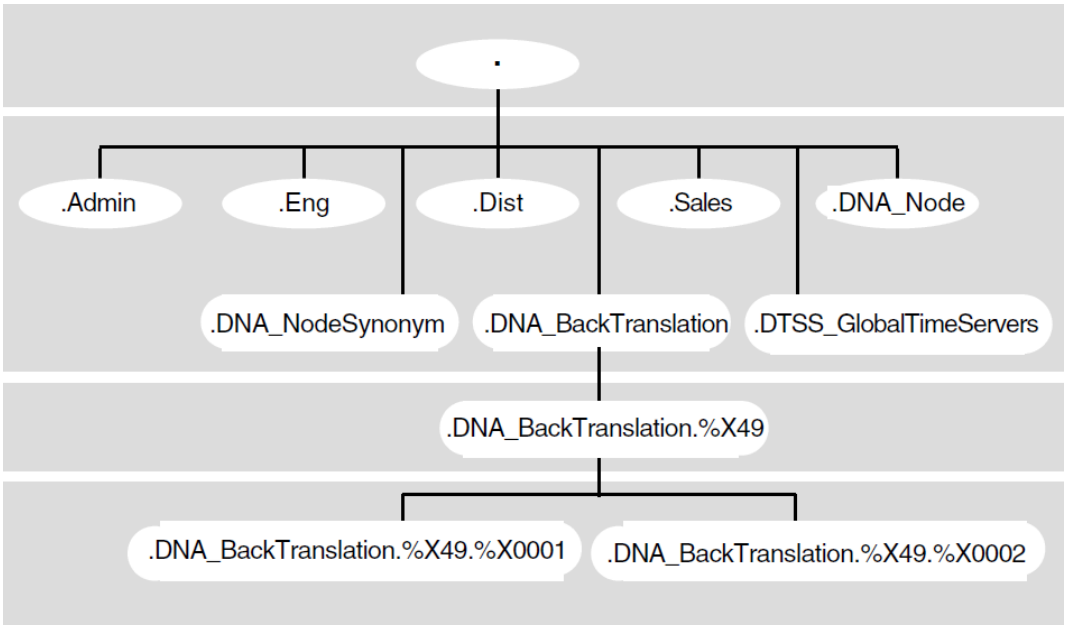
  - DFS access point names.
  - Names of RSM client and server systems.
  - Names of code libraries to be created and used by an IAF-developed application.
  - Names of development Notes conferences.
  - Names of industrial processes used in a distributed manufacturing work flow controller, to be designed by an IAF engineer.
  - The `.Eng.Dir_Admin` group.
- `.Dist` directory
  - Names of DECnet-Plus nodes used by the distribution organization.
  - Names of RSM client and server systems.
  - Message bus queue names.
  - The `.Dist.Dir_Admin` group.

In addition, the following directories would be created during or soon after configuration of the first DECnet-Plus node:

- `.DNA_Node` directory, a directory to contain node object entries for systems still running DECnet Phase IV.
- `.DNA_NodeSynonym` directory, to contain soft links pointing from a Phase IV-style version of a node name to the node object entry.
- `.DNA_BackTranslation` directories, to contain soft links pointing from the address of a node to its node object entry. The top-level `.DNA_BackTranslation` directory would have a child directory for each initial domain part (IDP) in the network, and the IDP directory would have a child for each area in the network. IAF planned to use only one IDP (the default IDP of 49) because the IAF network is private and the company is not interested in communicating with other OSI networks. Two area child directories would be created, for the New York LAN (Area 1) and the Chicago LAN (Area 2). The area directories would contain the node address-to-name soft links.
- `.DTSS_GlobalTimeServers`, a default directory that DECdts uses for storing information about global time synchronization servers. Based on the recommendation in Chapter 10 for one global server on each LAN, the planners anticipated this directory would contain two entries.

With these additional directories added into the hierarchy, the complete picture of the namespace would look like the one in Figure 9.3. The shaded areas separated by white lines indicate levels of the hierarchy.

Figure 9.3. Complete IAF Namespace Hierarchy



The committee could see that, at least in the initial stages of DECdns usage, the sales and engineering directories and the nodes oft link directories would be the most heavily populated. However, the planners were not concerned that some directories would be larger than others. They knew that load balancing is determined not by directory size but by the number of servers in the namespace and how directories are replicated on those servers. Additionally, in such a small namespace, none of the directories would contain enough entries to have significant impact relative to any other directory.

9.1.4. Choosing Servers and Planning Replication

The next step for the committee was to plan how they were going to replicate directories and how many DECdns servers they would need.

After considering the guidelines in Chapter 8, the planners decided to configure two servers on the New York LAN and two servers on the Chicago LAN. The clearinghouses at the two New York servers would both contain an identical set of directory replicas. The planners felt this replication scheme would help to reduce confusion during the initial configuration of DECdns in the network. They also knew it would help to balance the work load between the two servers, increase the likelihood of finding a name without going off the LAN, and provide automatic, real-time backup of data in case a clearinghouse or replica became corrupted or temporarily unavailable.

The committee selected systems based on estimates of 1 MIPS of CPU power, 3,000 disk blocks, and 1 megabyte (2,000 pages) of memory required to run DECdns in the IAF network. In choosing the first server to be configured in New York, the committee also considered additional requirements, because that system would be the first system in the network to make the transition to DECnet-Plus. Table 9.1 indicates the systems selected to be DECdns servers.

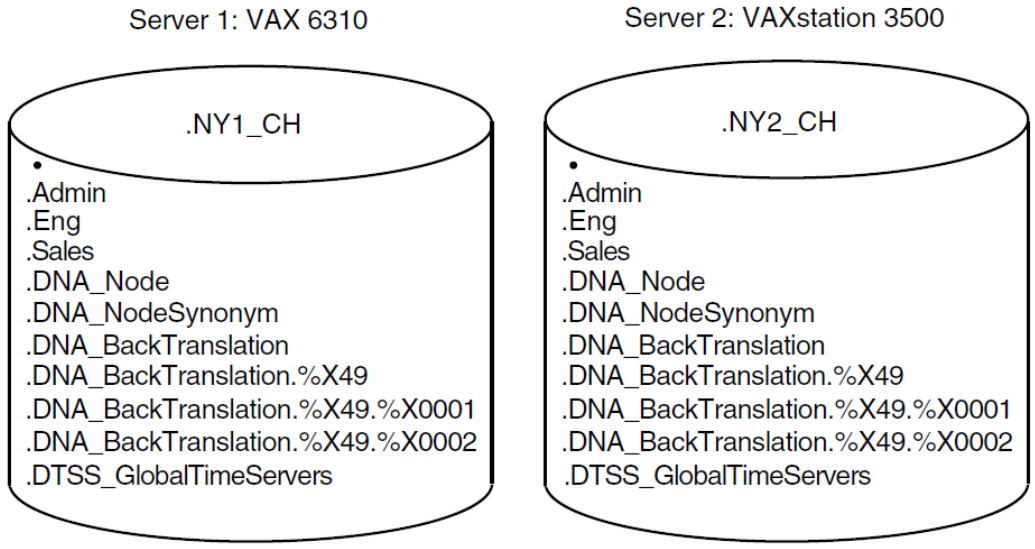
Table 9.1. IAF Corporation DECdns Server Plan

Clearinghouse Name	Server Hardware
.NY1_CH	VAX 6310
.NY2_CH	VAXstation 3500

Clearinghouse Name	Server Hardware
.Chicago1_CH	DECsystem 5400
.Chicago2_CH	DECstation 3100

Figure 9.4 shows the plan for the first two servers in the namespace and the directory replicas their clearinghouses would contain.

Figure 9.4. Replication Plan for New York Servers



Because the `.Dist` directory would not be used until the Chicago LAN began the transition to DECnet-Plus, the committee decided not to create the directory until that time.

IAF was ready to configure its namespace. The planning committee had established a general naming policy, planned their directory structure and decided on an access control policy, and planned to replicate the same set of directories at each of the first two servers in the namespace. The committee had chosen the first node to migrate to DECnet-Plus and had selected three additional systems to be DECdns servers. More decisions could come later, as the Chicago LAN began the transition and the namespace grew. IAF was ready to begin building its namespace.

9.2. Part 2: The Namespace Grows

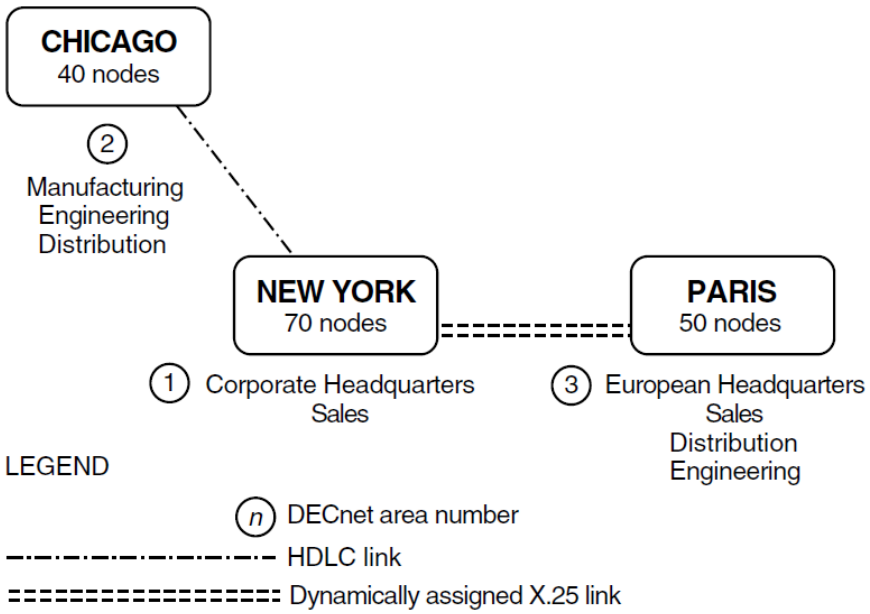
Eight months after IAF completed the first phase of its transition to DECnet-Plus, it was ready to migrate the nodes on the Chicago LAN. In the meantime, some major changes occurred within the company as well. As part of its plan to expand its international operations, IAF bought a similar company in Paris, France, called AeroFrance. The AeroFrance network was a LAN consisting of 100 DECnet nodes running Phase IV software. It connected to New York headquarters through an X.25 link.

The AeroFrance company had a very strong sales and distribution organization, whose headquarters remained in France. It had a small engineering group that also remained in France. Some administrative functions stayed at the Paris site, which became IAF's European headquarters. Other functions moved to corporate headquarters in New York, increasing the number of nodes there to 70 and reducing the number of nodes in Paris to 50.

When IAF acquired AeroFrance, IAF consolidated its U.S. engineering operations. IAF thought that the New York engineering group should be physically closer to the distribution organization that it

served. Also, both the New York and Chicago engineering groups were too small to be under separate administrators. Therefore, the two groups were combined in Chicago, increasing the number of nodes there to 40. Figure 9.5 shows the new network topology after the acquisition of AeroFrance and the combination of the domestic engineering groups in Chicago.

Figure 9.5. Expanded IAF Network

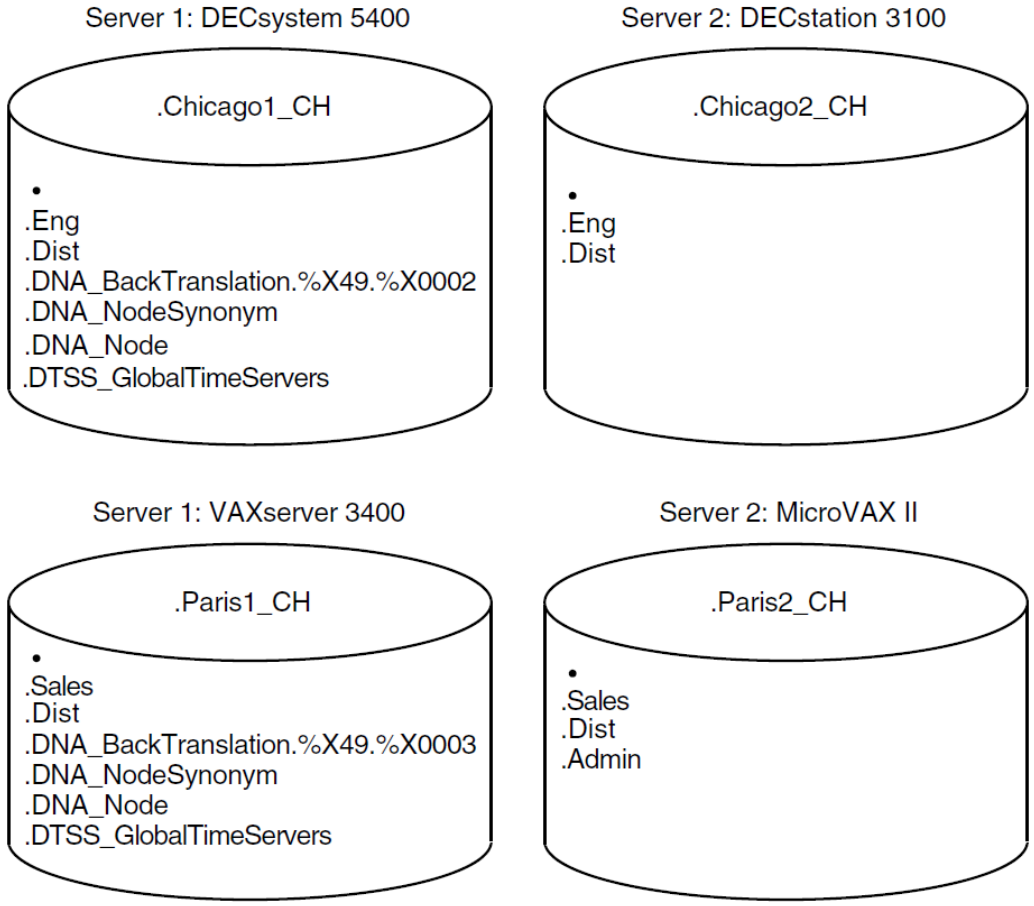


IAF decided to start migrating the AeroFrance nodes to DECnet-Plus on a parallel transition schedule with the Chicago LAN, so that the Paris LAN could also take advantage of DECdns and other features of DECnet-Plus. The acquisition, the engineering reorganization, and the DECnet-Plus transition caused several changes to the namespace over the next year.

One major change was the addition of DECdns servers. Because the newly acquired French company was a single LAN, IAF took the same approach it had planned for New York and Chicago and configured two servers in Paris. The new network, combined with transition of the Chicago site, resulted in four additional servers, with clearinghouses named `.Chicago1_CH`, `.Chicago2_CH`, `.Paris1_CH`, and `.Paris2_CH`.

Figure 9.6 shows the four new DECdns servers — two in Chicago and two in Paris — and their contents at the end of the year.

Figure 9.6. New Servers and Their Contents

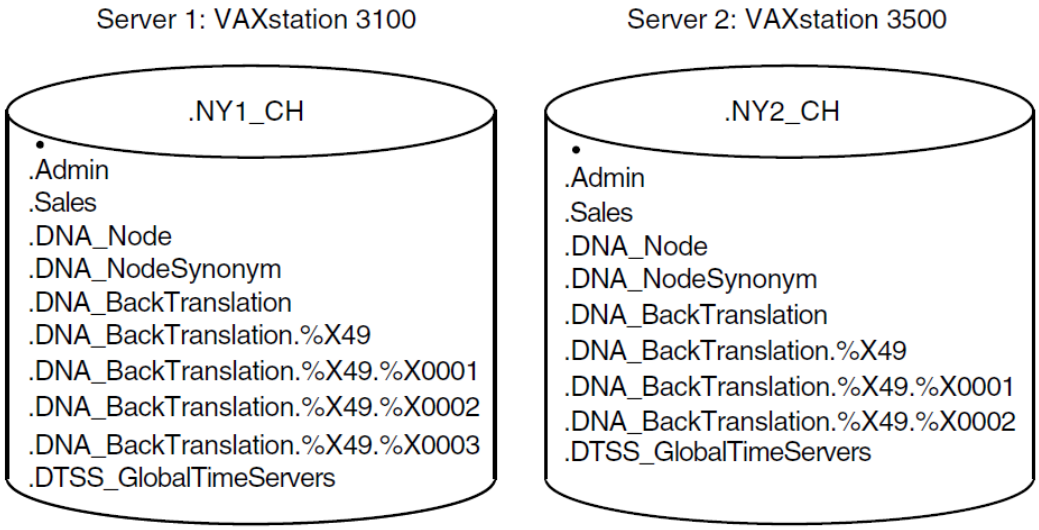


The figure reflects the following changes:

- Creation and replication of the distribution directory (`.Dist`).
- Replication of other directories that were created at New York servers.

Figure 9.7 shows the status of the two New York clearinghouses at the end of the year.

Figure 9.7. New York Server Contents a Year Later



The figure reflects these changes:

- Creation of a new backtranslation area directory at Server 1 to accommodate soft links for the Paris nodes in Area 3.
- Transfer of the `.NY1_CH` clearinghouse to a VAXstation 3100 system.
- Removal of the `.Eng` replicas from both New York clearinghouses.

The remainder of this chapter explains in detail each of the changes that occurred and refers to other documentation for details on the activities described.

- After nearly a year of using DECdns, the namespace administrator decided that it would be better to move at least one of the servers in New York off of a timesharing system. The Network Control Center wanted a central node from which they could manage the network and possibly add some new routing capabilities. So IAF purchased a VAXstation 3100 and moved `.NY1_CH` from the VAX 6310 to the new machine.

  See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for more on **moving a clearinghouse**.

- The newly acquired AeroFrance LAN was in a new DECnet Phase V area, number 3. Therefore, a new backtranslation directory was necessary for the addresses of nodes in Area 3. At the DECdns server in New York, the namespace administrator ran the DECnet node registration tool to create the new area directory and fill it with Area 3 addresses. The tool was also used to

  - Populate the `.DNA_Node` directory with AeroFrance nodes still running Phase IV.
  - Populate the `.Sales`, `.Dist`, and `.Admin` directories with AeroFrance nodes as they upgraded to DECnet-Plus.
  - Fill the `.DNA_NodeSynonym` directory with node synonym soft links for the AeroFrance nodes.

  See the *VSI DECnet-Plus for OpenVMS Network Management Guide* for more on **registering node names and soft links**.

- Once DECdns servers were configured in Area 2 (Chicago) and Area 3 (Paris), the namespace administrator replicated the Area 2 backtranslation directory at a server in Chicago and the Area 3 backtranslation directory at a server in Paris. This was in accordance with the recommendation that the master of the replica of a backtranslation area directory should reside in the area it describes.

  See Section 8.2.1 for detailed **replication guidelines** for the node name and soft link directories. See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for more on **replicating a directory**.

- The `.DTSS_GlobalTimeServers, .DNA_NodeSynonym`, and `.DNA_Node` directories were replicated at one server in Chicago and one server in Paris for availability on those LANs.

  See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for more on **replicating a directory**.

- Because many administrative functions and resources stayed in Paris after IAF bought AeroFrance, the namespace administrators decided it would be efficient to replicate the `.Admin` directory in Paris.

See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for more on **replicating a directory**.

- The namespace administrator created the `.Dist` directory in `.Chicago1_CH` and populated it with names of distribution nodes as they migrated to DECnet-Plus, names of message bus queues, and RSM client and server names. The `.Dist` directory was replicated at the second DECdns server in Chicago.

  See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for more on **creating a directory**.

- As a result of the New York engineering group's move to Chicago, the namespace administrators replicated the engineering directory at the two Chicago DECdns servers. They also realized that it was no longer necessary to have two replicas of the engineering directory in New York, and decided to delete them from the New York servers. However, one of the replicas in New York was the master replica, since the directory had been created there. So they declared a new epoch for the directory, redesignating the master replica in Chicago. Then they were able to delete the two replicas from the New York clearinghouses.

  See the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for more on **setting a new epoch for a directory**.

Chapter 10. Preparing for DECdts

This chapter describes how to plan your DECdts implementation, including personnel selection for the planning process, and planning for DECdts on a LAN, an extended LAN, or a WAN. The DECdts software is shipped on the same media as the DECnet-Plus software, so hardware installation considerations are only included by reference. Some of the planning considerations for DECdts are tied to the planning of DECdns. Plan how you will use the two products simultaneously. For DECdns planning considerations, see Chapter 7 and Chapter 8.

Two main categories of personnel will interact with the DECdts software: system managers and applications programmers. Programmers do not usually need to be involved in the planning stages of the DECdts implementation. If you are writing a program to import a source of Coordinated Universal Time (UTC) into the service, however, you may wish to locate the time-provider at the server that is closest to the programmer. Close proximity to the time-provider will help the programmer when testing the software application with the time-provider hardware.

System managers or network architects usually plan the DECdts implementation. System managers also install the software and maintain DECdts. They decide which nodes will be servers and which will be clerks, and decide how the DECdts implementation will grow with the network. DECdts is fully scalable for networks of any size, so expanding the implementation to include new nodes is relatively simple.

All references to DECdts servers apply to those servers running on ULTRIX, UNIX, or OpenVMS VAX systems.

10.1. Planning a DECdts Implementation

Consider the following questions as you plan your DECdts implementation:

- Is your network a single LAN, an extended LAN, a WAN, or a combination of LANs and WANs?
- What is the current or proposed network topology (component placement)?
- How many servers will be required? Where will they be located?
- Will global servers be required? Where will they be located?
- Will you need to configure some local servers as couriers if you are using global servers?
- Will you use an external time-provider to obtain UTC?
- What will the time zone be for each system?

The following sections will help you answer these questions. The worksheet at the end of the chapter is provided to help you map your time service plan before you begin the installation.

Although there are many network configurations that affect DECdts planning, several general rules apply regardless of your network configuration or the number of nodes in the network. These guidelines are summarized as follows:

- Your network should have a minimum of three DECdts servers.
- Each LAN should have a minimum of one server.

- You should locate the servers at the sites with the greatest number of nodes or at backbone sites with high-speed links to the rest of network.

Although you must consider many other factors when planning your network, these factors depend on your network topology and configuration. The following sections present some typical network arrangements to help you implement DECdts on your own network.

10.2. Configuration Planning on a LAN

If your nodes are in a single local area network, regardless of the number of nodes, planning your DECdts implementation is relatively simple. You can configure a minimum of one system as a server, but to enhance reliability and to detect faulty time servers, configure at least three systems as servers. If you want to provide redundancy for your DECdts implementation, plan to install four or more servers in the network. That way, if one of the servers fails, DECdts can still synchronize with reliable results.

To ensure the reliability of your DECdts implementation, make sure that the network connections between server nodes are stable. If you plan to add WAN links to your LAN, do not move the servers to the remote nodes, because WAN links are usually less reliable than the LAN.

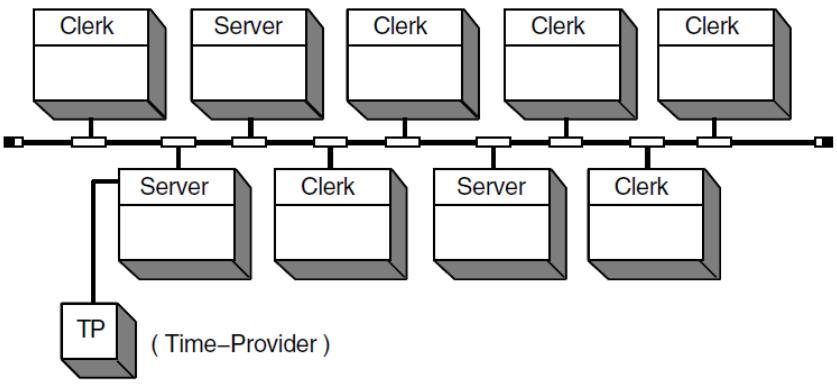
If you have a single LAN, the location of the servers on the LAN is not critical. You can locate one of the servers on a readily accessible node to aid in troubleshooting, but there are no other recommended server locations. Neither global servers nor couriers are required.

For all network configurations, you must install DECdns concurrently with DECdts to run both services. DECdns is usually configured with two name servers per LAN. To ease installation, management, and maintenance, you can locate the DECdns global name servers and two of your local DECdts servers on the same nodes. The DECdts servers will not add significantly to the processing or memory demands on anode.

If you are planning to use one or more time-providers, locate the mat easily accessible systems to ease startup and maintenance. If your network requires only synchronized clocks, but does not need to closely follow a time standard such as UTC, you may not require a time-provider. If you do not use a time-provider, we recommend that you use the **update** command to manually set the time approximately once each week.

Figure 10.1 shows a simplified LAN configuration. Your LAN maybe much larger, but the figure should resemble a portion of your network.

Figure 10.1. DECdts LAN Configuration



10.3. Configuration Planning on an Extended LAN

Configuring an extended LAN can be as simple as configuring a single LAN, or it can be more complex, depending on the type of bridge that connects each LAN segment and how many nodes are in each segment. For the purposes of DECdts, extended LANs fall into one of the following categories:

- Extended LANs that use the LAN Bridge 100 or METROWAVE bridges.
- Extended LANs that use the Vitalink TransLAN or the high-performance bridges but block multicast messages between segments.

The following sections describe possible configurations for each type of extended LAN.

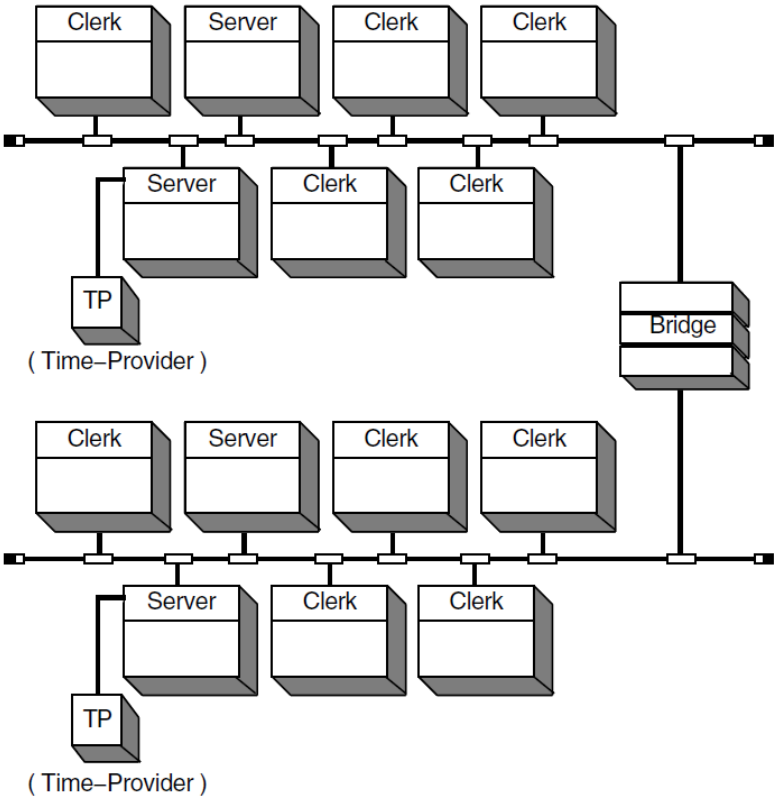
10.3.1. Extended LANs Using High-Performance Bridges

If you are using the LAN Bridge 100 or METROWAVE bridges to connect the different segments of your LAN, and you have not configured the bridges to block multicast messages between segments, the bridges can be transparent to DECdts. Neither of these bridges introduces a significant amount or variation in delay during intersegment synchronization, provided that the following conditions are also met:

- The bridges are not bottlenecks, due to extremely high volumes of traffic.
- There are few redundant bridges connected in series.
- The network is small or medium sized.

If you follow these guidelines, you can spread the time servers over the network at whatever locations are convenient, just as you would for a single LAN. You need not set up three servers on each segment, because multicast synchronization messages will be relayed over the network interface without delay. When placing the servers (including name servers), you need only consider the quantity of servers, not their locations. Figure 10.2 illustrates this type of configuration.

Figure 10.2. DECdts Configuration — Unified Extended LAN



10.3.2. Extended LANs Using Medium-Speed Bridges

If you are using TransLAN bridges between your extended LAN segments, or if any of the following conditions exist, the number and placement of servers in each extended LAN segment is critical:

- Multicast messages are blocked at the bridge.
- The bridge is frequently unavailable.
- Delays are sometimes incurred due to high intersegment traffic.

If the bridge does not forward multicast messages, local servers in each segment cannot receive advertisement messages from local servers in other segments. Even if a bridge forwards multicast messages, the variable delays between the LAN segments can lead to high clock skews between local servers on opposite sides of a bridge. In all cases, treat each segment of the extended LAN as though it were a separate LAN.

Because you should consider each segment of your extended LAN as a separate entity while planning your DECdts implementation, configure each segment according to the following guidelines. Note that you must also take DECdns configuration into account.

- Create a minimum of one server in each segment.
- For the lowest skew between nodes, create three or more servers in each segment.
- Configure one server on each segment as a courier.
- Use the `advertise` command to configure one server on each segment as a global server.

- If you are using time-providers, connect them to global servers.

10.4. Configuration Planning on WANs and WAN Links

Because there are many variations on WAN configurations (especially in combination with LANs and extended LANs), it is impossible to describe every case where a WAN link can be used to disseminate time. This section does not give recommendations for every case involving a WAN link, but describes how you can set up your DECdts implementation using several generic configurations as examples.

Due to the variable delay inherent in any WAN link, it is difficult to maintain a consistent skew between clocks on opposite sides of the link. DECdts synchronizes clocks across WAN interfaces, but larger inaccuracies occur between the clocks to account for the worst case transmission delay during each synchronization.

A reliable and robust DECdts implementation is important any time WAN links are part of your network. Because WANs are less reliable than LANs, plan for some redundancy in any DECdts implementation involving WAN links. Try to place servers so there will always be three or more available, even if one of the WAN links goes down.

The following sections give recommendations for three basic WAN configurations:

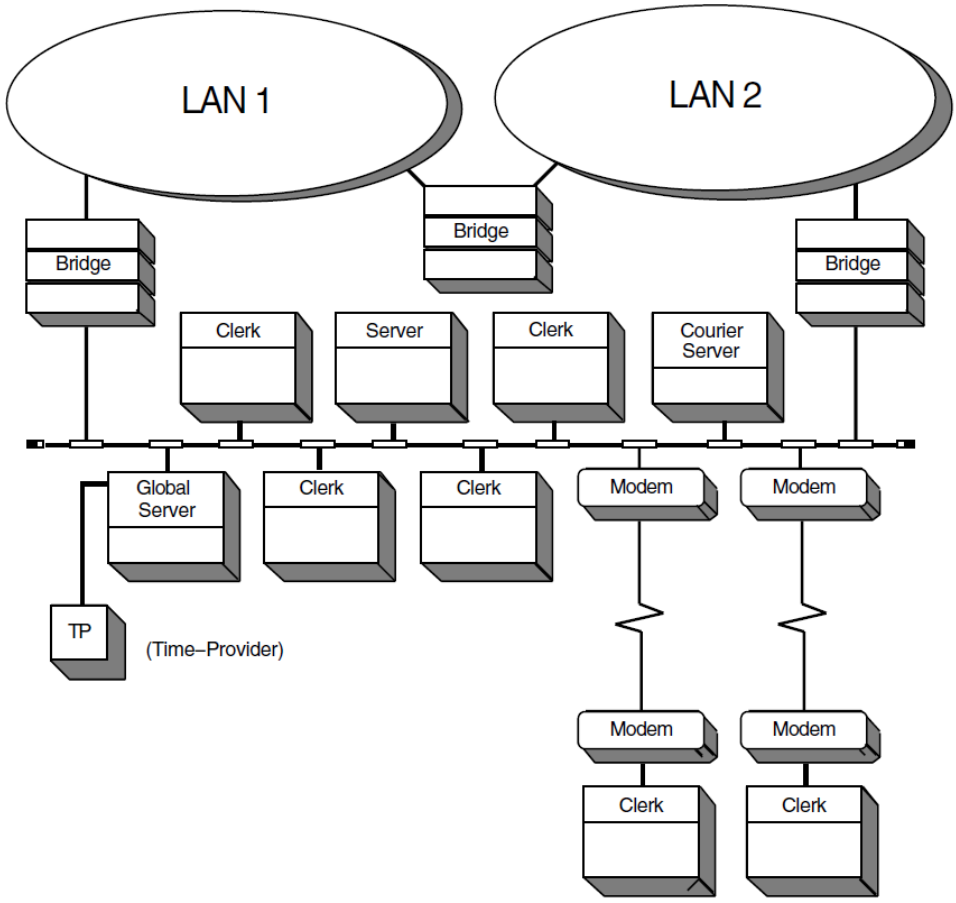
- A LAN or extended LAN with WAN links to remote nodes
- LANs connected by WAN links
- An all-WAN network with a central host or cluster

Your network may not exactly match any of the configurations, but you will be able to plan your network by following the recommendations for each example.

10.4.1. LANs with WAN Links to Remote Sites

Figure 10.3 shows a LAN that incorporates several remote nodes by using WAN links.

Figure 10.3. DECdts Configuration — LAN with WAN Links



In this configuration, follow the basic recommendations for a single LAN, but also adhere to the following rules:

- Create a minimum of one server in each LAN; to increase reliability, create three or more servers in each LAN.
- Configure servers at remote sites as global and courier servers.
- For each remote clerk, set the `servers required` attribute to (3).
- If you are using a single time-provider, locate it at a global server on the LAN with the greatest number of clerks.

The network configuration resulting from the rules outlined above concentrates the servers on the LAN, so clock skews are kept to a minimum and the service is not dependent on remote nodes that may be physically inaccessible to the system manager. Each remote clerk node synchronizes with the global servers on the LAN in order to satisfy the `servers required` requirement.

10.4.2. LANs Connected by WAN Links

The rules outlined for extended LANs that use the Vitalink bridge also apply to LANs connected by WAN links. Each LAN in such a network is a separate entity, so several DECdts components must be configured on all of the LANs. For the best performance, configure each LAN according to the following guidelines:

- Configure at least three DECdts servers on each LAN.

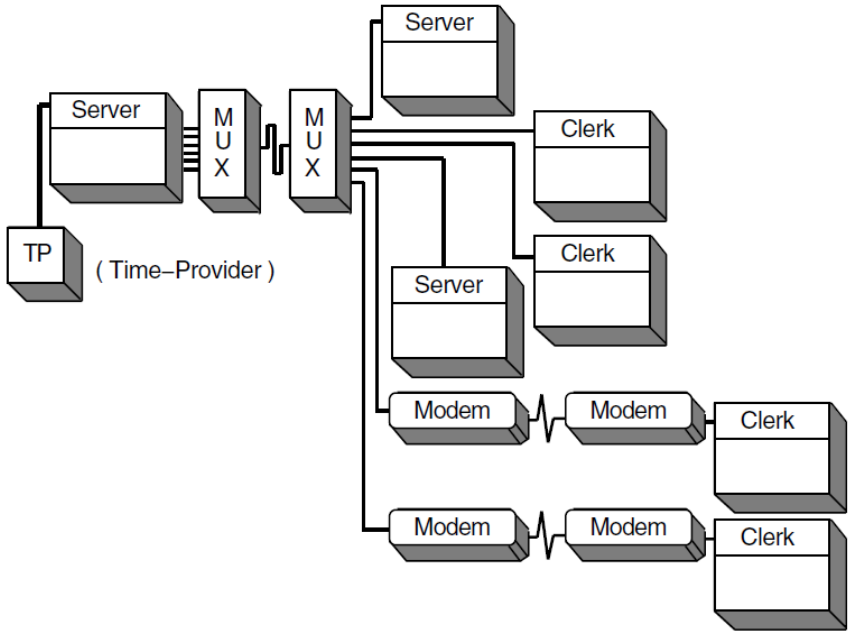
- Configure at least one server on each LAN as a courier.
- Configure at least one global server on each LAN.
- If you are using time-providers, install them in the LANs that contain global servers.

These recommendations will result in peak DECdts efficiency and availability despite the irregular delays associated with WAN links.

10.4.3. WAN Networks

Figure 10.4 shows a geographically distributed network with no LANs. DECdts delivers higher clock skews in an all-WAN environment, but still provides synchronization adequate for most distributed applications. In such a network, clock skews approximate the round trip delay between nodes.

Figure 10.4. DECdts Configuration — WAN Networks



Many of the same recommendations for a LAN with WAN links also apply to the network that does not have any LANs. Keep the following considerations in mind when planning your all-WAN network:

- The network should have at least three servers (preferably four or more).
- Every server should be configured as a global server.
- Couriers are not required, however, you can configure any or all of the servers as couriers. The `servers required` attribute will force each global server to synchronize with at least two others.
- Local servers are not required.
- You can place the servers anywhere in the network, but place at least one at the central site; choose the most active remote nodes connected by the most reliable links for the rest of the servers.
- If you are using time-providers, place one at the central site. Connect each time-provider to a global server.

10.5. Planning for External Time-Providers

To closely synchronize your systems with UTC and with each other, you can place one or more time-providers in your network. Time-providers have many forms: they can be radio receivers, software/modem combinations, or satellite receivers. See the *VSI DECnet-Plus DECdts Programming* guide for additional information about time-providers and the time-provider interface that you can use to integrate these devices in your network.

If you plan to use time-providers in your network, you can write a time-provider program to match the time-provider interface or use one of the sample programs that are supplied with the DECdts software. After you select your time-provider device and program, plan where to install the device in your network.

It is relatively simple to locate time-providers to your best advantage. Time-providers must be located at servers; if your network has several segments and you are using global servers to maintain synchronization across the network, locate the time-providers at the global servers. Regardless of your network configuration, place the time-providers where they will have the highest availability and use.

Note

You cannot configure a server connected to a time-provider as a courier. A server connected to a time-provider never assumes the courier role, because the server process only solicits time values from the time-provider. For additional information about courier servers, see the *VSI DECnet-Plus DECdts Programming* guide.

10.6. DECdts Planning Worksheet

This section provides a worksheet for you to fill out before you install the DECdts servers. The worksheet provides a place to keep track of the server systems so you can manage them remotely using the NCL management interface (you must have applicable privileges).

Local Servers:				
Location	**Node Name**	**Node Number**	**Manager**	**How to Contact**
Global Servers:				
Location	**Node Name**	**Node Number**	**Manager**	**How to Contact**

Local Servers:				
Network DECdns Information:				
DECdns manager:				
Closest DECdns server node:				
Format for global server name:				