

VSI OpenVMS

VSI DECnet-Plus for OpenVMS DECdts Management Guide

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

VSI DECnet-Plus for OpenVMS DECdts Management Guide



VMS Software

Copyright © 2025 VMS Software, Inc. (VSI), Boston, Massachusetts, USA

Legal Notice

Confidential computer software. Valid license from VSI required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

The information contained herein is subject to change without notice. The only warranties for VSI products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. VSI shall not be liable for technical or editorial errors or omissions contained herein.

HPE, HPE Integrity, HPE Alpha, and HPE Proliant are trademarks or registered trademarks of Hewlett Packard Enterprise.

Intel, Itanium and IA-64 are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Table of Contents

Preface	vii
1. About VSI	vii
2. Intended Audience	vii
3. Document Structure	vii
4. Related Documents	vii
5. VSI Encourages Your Comments	vii
6. OpenVMS Documentation	viii
7. Typographical Conventions	viii
Chapter 1. Introduction to the VSI DECnet-Plus Distributed Time Service	1
1.1. DECdts Advantages	2
1.1.1. Applications Support	2
1.1.2. External Time-Provider Support	3
1.1.3. Manageability	3
1.1.4. Quantitative Inaccuracy Measurement	4
1.2. Basic DECdts Concepts	4
1.2.1. Time Measurement Factors	4
1.2.1.1. Clock Error	4
1.2.1.2. Communications and Processing Uncertainties	4
1.2.2. Inaccuracy	5
1.2.3. Synchronizing System Clocks	6
1.2.4. How DECdts Adjusts System Clocks	8
1.2.5. DECdts Time Representation	9
1.2.5.1. Absolute Time Representation	9
1.2.5.2. Relative Time Representation	10
1.3. How DECdts Works	11
1.3.1. Clerks	11
1.3.2. Servers	13
1.3.2.1. Local Servers	13
1.3.2.2. Global Servers	13
1.3.2.3. Couriers	14
1.4. Interoperation with DECdns	14
Chapter 2. Managing DECdts	15
2.1. The DECdts Management Interface	15
2.1.1. Interface Components and Command Syntax	15
2.1.2. DECdts Timestamp Syntax	15
2.1.3. The DECnet Startup Script	16
2.1.4. Accessing and Exiting the Management Interface	17
2.2. Creating and Enabling DECdts	17
2.2.1. Creating DECdts	17
2.2.2. Enabling DECdts	18
2.2.3. Controlling LAN Devices Used by DECdts	18
2.2.4. Designating Global Servers and Couriers	19
2.2.4.1. Advertising Global Servers to DECdns	19
2.2.4.2. Designating Global Servers Using the BIND Database	19
2.2.4.3. Designating Global Servers with the Site-Specific Template	21
2.2.4.4. Assigning the Courier Role to Servers	22
2.2.5. Changing the DECdts Entity Type	22
2.2.6. Option to Disable DECdts at System Boot	23
2.3. Summary of Management Commands and Characteristics	23

2.4. Routine Management Tasks	25
2.4.1. Critical Commands and Characteristics	25
2.4.1.1. The Servers Required Characteristic	25
2.4.1.2. Use with Global Servers	26
2.4.1.3. Use with Systems on Point-to-Point Lines	27
2.4.1.4. The Maximum Inaccuracy Characteristic	27
2.4.1.5. The Synchronization Hold Down Characteristic	28
2.4.1.6. The Error Tolerance Characteristic	28
2.4.1.7. The LAN Timeout, WAN Timeout, and Query Attempts Characteristics	30
2.4.2. Server Management	31
2.4.2.1. Matching the Epoch	31
2.4.2.2. The Advertisement Interval and Check Interval Characteristics	31
2.4.3. Modifying the System Time	32
2.4.3.1. Updating the Time Monotonically	32
2.4.3.2. Changing System Time Nonmonotonically	33
2.4.3.3. Forcing System Synchronization	34
2.4.3.4. Setting the System Clock to Local Time	34
Chapter 3. Problem Solving	37
3.1. A Clerk System's Clock is Incorrect	37
3.2. A Server System's Clock is Incorrect	38
3.2.1. Large Inaccuracy On One Server	38
3.2.2. Large Inaccuracy On All Servers	39
3.3. Local Servers are Unavailable or Unresponsive	40
3.4. Global Servers are Unavailable or Unresponsive	41
3.5. The Advertise Command Fails	41
3.5.1. Global Directory Not Configured	41
3.5.2. Access Violation Messages	42
Chapter 4. DECdts Command Dictionary	45
4.1. The DTSS Entities	45
4.2. Command Format	46
4.2.1. Attribute Groups	46
4.2.2. Prepositional Phrases	47
4.2.3. NCL Access Control	47
4.2.4. Wildcards	48
4.3. Editing the Commands	48
4.4. Accessing and Exiting the Management Interface	49
4.5. Commands	49
Appendix A. Event Messages	71
Appendix B. Time-Providers and Time Services	75
B.1. Criteria for Selecting a Time Source	75
B.2. Sources of Coordinated Universal Time	76
B.2.1. Telephone Services	76
B.2.2. Radio Broadcasts	76
B.2.3. Network Time Protocol	79
B.2.4. Satellite	79
B.3. World Time Zone Map	79
Appendix C. Interoperation with NTP	81
C.1. Getting the Time from Remote NTP Time Sources	81
C.2. Giving the Time to NTP Nodes	83

C.3. Preventing Loops	84
-----------------------------	----

Preface

The VSI DECnet-Plus Distributed Time Service (DECdts) is a networkwide service that runs on the OpenVMS operating system. DECDts enables systems to synchronize their clocks with all other clocks in the network. This manual describes DECDts concepts, how to use the DECDts software to manage system clocks, and how to solve problems once you have configured the software. For information on installing and configuring DECDts, see the *VSI DECnet-Plus for OpenVMS Introduction and User's Guide*, the *VSI DECnet-Plus Planning Guide*, and the *DECnet-Plus for OpenVMS Installation and Basic Configuration Manual*.

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 manual is intended for two audiences: network managers who administer DECDts, and server managers who are in charge of day-to-day operations of DECDts server systems. Both audiences should have a sound understanding of DECDts concepts before using the software. The management and problem-solving presentations may apply to either of the two audiences, depending on the topology of the network. (For small networks, one person often takes the roles of network manager and server manager.)

Network managers who are planning a DECDts implementation should refer to the DECDts information in the *VSI DECnet-Plus for OpenVMS Introduction and User's Guide*, the *VSI DECnet-Plus Planning Guide*, and the *DECnet-Plus for OpenVMS Installation and Basic Configuration Manual* before installing and using the software.

The concepts portion of the manager's guide may also be useful as background information for programmers or managers of client applications that use DECDts. Programmers should read the *VSI DECnet-Plus DECDts Programming* for DECDts internals information and programming concepts.

3. Document Structure

This manual is organized into four chapters. The first chapter introduces DECDts and basic time synchronization concepts. The second chapter describes how to manage DECDts, the third chapter describes common DECDts problems and how to solve them, and the fourth chapter describes management commands. The appendices contain reference information including event messages, time-provider sources, and interoperation with Network Time Protocol (NTP).

4. Related Documents

For a list of additional documents that are available in support of this version of the operating system, refer to the *VSI DECnet-Plus for OpenVMS Introduction and User's Guide*.

5. 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.

6. OpenVMS Documentation

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

7. Typographical Conventions

The following conventions may be used in this manual:

Convention	Meaning
<code>special type</code>	Indicates a literal example of system output or user input. In text, indicates command names, keywords, node names, file names, directories, utilities, and tools. On a DECnet-Plus for OpenVMS system, enter the word or phrase in the exact case shown. You can abbreviate command keywords to the smallest number of characters that OpenVMS, NCL, DECdns, DECdts, and the other utilities accept, usually three characters.
<i>italic</i>	Indicates a variable.
text style	Indicates a new term defined either in the text or in the <i>VSI DECnet-Plus for OpenVMS Introduction and User's Guide</i> glossary or important text.
Return	Indicates that you press the Return key.
Ctrl/ x	Indicates that you press the Control key while you press the key noted by <i>x</i> .
[]	In command format descriptions, indicates optional elements. You can enter as many as you want.
{ }	In command format descriptions, indicates you must enter at least one listed element.

Chapter 1. Introduction to the VSI DECnet-Plus Distributed Time Service

This chapter gives a conceptual overview of the VSI DECnet-Plus Distributed Time Service (DECdts) and its functions. Some basic time and clock concepts, DECdts time representation, and basic DECdts operation are also presented.

DECdts is a software-based service that provides precise, fault-tolerant clock synchronization for systems in local area networks (LANs) and wide area networks (WANs). The clock synchronization provided by DECdts enables distributed computing applications to determine event sequencing, duration, and scheduling. DECdts software is packaged with DECnet-Plus for OpenVMS systems.

DECdts consists of software components on a group of cooperating systems; it conforms to the client/server model used in many of VSI's distributed processing products. In the DECdts implementation, each server supplies the time to many client systems and applications through intermediaries called **clerks**. Clerks reside on their client systems. (Note that throughout this document, the term **entity** is used to refer to the server or clerk process when they have the same functions.)

Most DECnet nodes have a clerk that adjusts the clock on its client system; at regular intervals, clerks obtain time values from one or several servers in the network. The nodes that do not have DECdts clerks have DECdts servers; in addition to providing time values to clerks, servers also adjust the system clocks on their host systems. Servers can also obtain reference time values from sources of standardized time that are outside the network.

Because no device can measure the exact time at any particular instant, DECdts expresses the time as an **interval** containing the correct time. In the DECdts model, the DECdts entity (clerk or server) on a given system obtains time intervals from several servers, and computes the intersection where the intervals overlap. The DECdts entity then adjusts the clock of its client system to the midpoint of the computed intersection. When the entity receives a time interval that does not intersect with the majority of received intervals, it rejects the nonintersecting value. This mechanism allows a DECdts entity to ignore random faulty values when computing new times, thereby ensuring that defective server clocks do not affect its client system.

DECdts also permits the importation of time values from outside sources, such as the U.S. National Institute for Standards and Technology (NIST). Many standards bodies disseminate time signals by radio, telephone, and satellite; commercial devices (time-providers) are available to receive and interpret these signals. DECdts offers a time-provider interface (TPI) that describes how a time-provider process can pass time values to a DECdts server and propagate them in the network. The time-provider interface also permits distributed time services from other vendors to interoperate with DECdts.

DECdts provides many other 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.
- **Quantitative Time Measurement**— DECdts uses specific measurement and manufacturer's specifications to determine the accuracy of the times reported by servers.

- **Fault Tolerance**— DECdts reports servers whose time values do not intersect with the majority, and does not use their time values during clock synchronizations.
- **NCL Management**— Through its management interface, DECdts offers Network Control Language (NCL) support for controlling and monitoring the software.
- **Applications Programming Interface (API)**— DECdts provides an interface that allows applications to obtain, compare, and calculate UTC time values.
- **Local Time Translation**— When displaying time values, DECdts translates the UTC times it uses internally into local time values.
- **Automatic Local Time Updates**— After the initial configuration of a DECnet-Plus system, DECdts updates the system time automatically according to the rule in effect for the geographical location of the system.
- **Monotonicity**— DECdts usually provides unidirectional clock adjustment to ensure that clocks do not run backwards. The `enable` and `change` commands also allow nonmonotonic clock adjustment if desired.
- **Automatic Configuration**— DECdts entities use multicast messages to obtain the locations of servers in a LAN, and the DECdns (DECdns) or a site-specific configuration file to obtain the locations of servers in a WAN.
- **Efficiency**— Complexity is placed in the servers; network overhead is minimal.

1.1. DECdts Advantages

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

1.1.1. 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 increase, DECdts becomes increasingly vital to process coordination.

There are several types of existing applications that use the synchronized time DECdts provides to system clocks. These applications must reference synchronized system clocks to coordinate the events that occur throughout the network. Applications use synchronized clocks for the following functions:

- **Event measurement.** Applications can read the system clock to start and stop timers and measure the elapsed time between events.
- **Event reporting.** Applications can read the clock when an event occurs and append a timestamp to the event report.
- **Event scheduling.** Applications can read the system clock and add a relative time to determine the occurrence of a future event.
- **Event sequencing.** Applications can determine the order of events by reading the event report timestamps derived from the synchronized system clock.

For new applications, DECdts provides an applications programming interface (API). The API provides routines that new applications can use to obtain and manipulate binary timestamps. The DECdts API supports ANSI C language constructs. See *VSI DECnet-Plus DECdts Programming* for further information on the DECdts API.

1.1.2. 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, VSI recommends that you use at least one external time-provider in combination with the DECdts software. For a list of time-provider hardware suppliers, see *Appendix B, "Time-Providers and Time Services"*. Sample time-provider programs are provided with the DECdts software on DECdts server systems. The programs are located in `sys$common:[syshlp.examples.dtss]`.

See *Table 1.1, "Time-Provider Programs and Related Time-Provider Suppliers"* for a list of currently available time-provider programs and related time-provider hardware or software suppliers.

Table 1.1. Time-Provider Programs and Related Time-Provider Suppliers

File Name	Related Supplier	Time-Provider Type
<code>dtss\$ntp_provider.c</code>	Various	Time-provider program for Internet Network Time Protocol
<code>dtss\$provider_acts.c</code>	U.S. NIST (North America)	TPP for data communications
<code>dtss\$provider.c</code>	Traconex, Spectracom, Heath, Hopf (North America & Europe)	TPP for RF receivers

DECdts servers can synchronize with time-providers by means of the time-provider interface (TPI), which is described in the *VSI DECnet-Plus DECdts Programming*. The TPI specifies the communications between the DECdts server process and the time-provider process.

When a DECdts server attempts to synchronize, it uses the TPI to check for a time-provider process. If one is available, the server synchronizes only with the time-provider. If no time-provider is connected to the server, the server synchronizes with other servers in the network.

By connecting a time-provider to a DECdts server, you can ensure that the server is synchronized closely with UTC. When other servers request a time from the server with the time-provider (the **TP server**), the TP server's precise time is propagated throughout the network. See *Section 1.2, "Basic DECdts Concepts"* for further information on time-providers and the server synchronization process.

1.1.3. Manageability

The DECdts entity runs as a background process; little or no input is required from the system manager to synchronize the system clock after DECdts is initially configured. DECdts is also fault tolerant; it prevents malfunctioning clocks from providing the wrong time to other clocks in the network. Occasionally, however, system managers may need to perform the following functions:

- Identify system clock problems
- Adjust system clocks
- Change DECdts attributes due to varying WAN conditions

DECdts provides a full-featured management interface that allows system managers to 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 provides event reporting, which notifies system operators and managers in the rare event that a system clock is inaccurate or fails to synchronize. For further information on solving synchronization problems with DECdts, see *Chapter 3, "Problem Solving"* and *Appendix A, "Event Messages"*.

1.1.4. 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. See *Section 1.2.3, "Synchronizing System Clocks"* for further information on the DECdts synchronization process.

1.2. Basic DECdts Concepts

This section describes time measurement factors, DECdts synchronization concepts, DECdts system clock adjustment, and DECdts time representations. Read this section to gain a basic understanding of DECdts concepts before progressing to *Chapter 2, "Managing DECdts"*.

1.2.1. Time Measurement Factors

This section describes the factors that affect time measurement and explains how DECdts handles them.

1.2.1.1. Clock Error

DECdts uses UTC as its time standard, and defines any difference between UTC and the time of a system clock as **clock error**.

All system clocks have common properties that contribute to clock error and interfere with the synchronization process. This section describes some of these properties; *Section 1.2.2, "Inaccuracy"* explains how DECdts handles these problems to accurately synchronize networked system clocks.

System clock error tends to increase over time; the rate of change of error is known as **drift**. If each system clock in a network started at the same time and ran at the same rate, the clocks would remain synchronized. Because each system clock drifts at a different rate, however, the system clocks throughout a network tend to become desynchronized. The difference between any two clock readings is known as the **skew** between the clocks. The clocks used in many OpenVMS computer systems have a specified maximum drift of 8 seconds per day. If uncorrected for several days, the skew between networked system clocks can inhibit the performance of distributed applications.

DECdts constantly tracks the drift of each system clock in the network and periodically synchronizes the clocks to reduce the skew among them. The DECdts server and clerk adjust the clocks in their client systems as the final step in this repeating synchronization process.

1.2.1.2. Communications and Processing Uncertainties

Communications delays can inhibit the synchronization process, especially when two systems communicate over a WAN or low-speed link. DECdts can cope with the known processing

delays required to send and receive messages between systems. Because of the varying quality of communications links, however, the time required to send, receive, and acknowledge messages varies from one message to the next. These delays cannot be known exactly, because transit over the network and the time required to read an incoming timestamp both vary.

Rather than using estimates of communications and processing delays, DECdts records all known error factors that accompany a time measurement sent over the network. This measurement enables DECdts to determine the relative quality of a time source regardless of its geographic location or changing conditions on communications links.

1.2.2. Inaccuracy

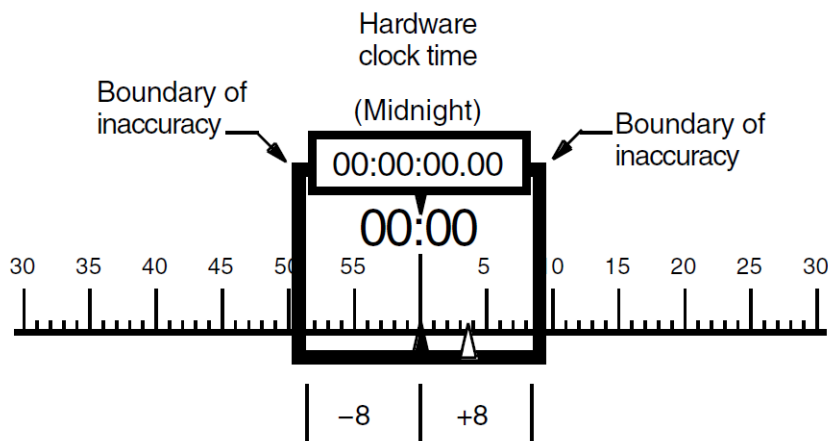
To synchronize system clocks to the most accurate settings, DECdts needs a way to determine the accuracy of time sources relative to each other and to UTC. This section describes how DECdts determines the relative accuracy of any time source available in the network.

DECdts uses an **inaccuracy** value, or tolerance, to determine the relative accuracy of time values that it obtains from system clocks and external time-providers. This DECdts feature effectively transforms each time value into an interval, or range, rather than a point on a continuum.

Inaccuracy values are determined by three factors:

- **Drift.** When reading a clock, DECdts calculates the amount of time that the clock may have drifted since DECdts previously synchronized and adjusted the clock. Drift is the largest component of most inaccuracy values.
- **Communications delay.** The inaccuracy also consists of the uncertain portions of the communications delays between systems. Although DECdts compensates for processing delays, it cannot predict or directly measure the varying delays that occur on network links. The inaccuracy values that a clerk or server obtains from collocated systems on a LAN tend to be much lower than those obtained from servers outside the LAN.
- **Leap seconds.** UTC time is measured by atomic clocks, which are extremely stable. The standard, however, keeps time based on the earth's position. Because of the slowing of the earth's rotation, it occasionally becomes necessary to advance UTC time by 1 second; these events are known as **leap seconds**. Leap seconds can occur in the final second of any month, and normally occur about once every 18 months. At the end of each month, DECdts accounts for leap seconds by increasing the inaccuracy measurements of client system clocks by 1 second; DECdts later adjusts the clocks to remove the extra second of inaccuracy if an external time-provider determines that a leap second did not actually occur.

Without DECdts to correct it, a system clock's inaccuracy is always increasing. For example, suppose that a clock starts with a UTC time of 00:00:00.000 (midnight) and zero inaccuracy. Because of drift, when the clock next shows a time of 00:00:00.000, the inaccuracy is 8 seconds. UTC time can be 23:59:52.000 or 00:00:08.000, but is probably somewhere in between. Therefore, the system time is an interval containing UTC time and bounded by the inaccuracy, as shown in *Figure 1.1, "Time and Inaccuracy"*. Using the DECdts format for displaying time, the combined time and inaccuracy interval is expressed as follows: 1990-08-03-00:00:00.000I08.000.

Figure 1.1. Time and Inaccuracy

Key:

△ = Coordinated Universal Time (UTC)

ZK-1996A-GE

1.2.3. Synchronizing System Clocks

To maintain uniform system times, DECdts servers and clerks periodically **synchronize** the clocks in all network systems. The DECdts entity on each system performs these synchronizations by requesting that servers send their combined clock and inaccuracy values (time intervals) to the originating system. The entity then uses the values sent by the servers to compute a new system time.

DECdts servers and clerks have slightly different synchronization procedures. Before attempting to synchronize with other systems, a DECdts server always checks to see whether an external time-provider is connected to the system. If no time-provider is available, the server requests times from other servers. When the server synchronizes with its peer servers, the server uses its current system time as one of the input values for computing a new system time.

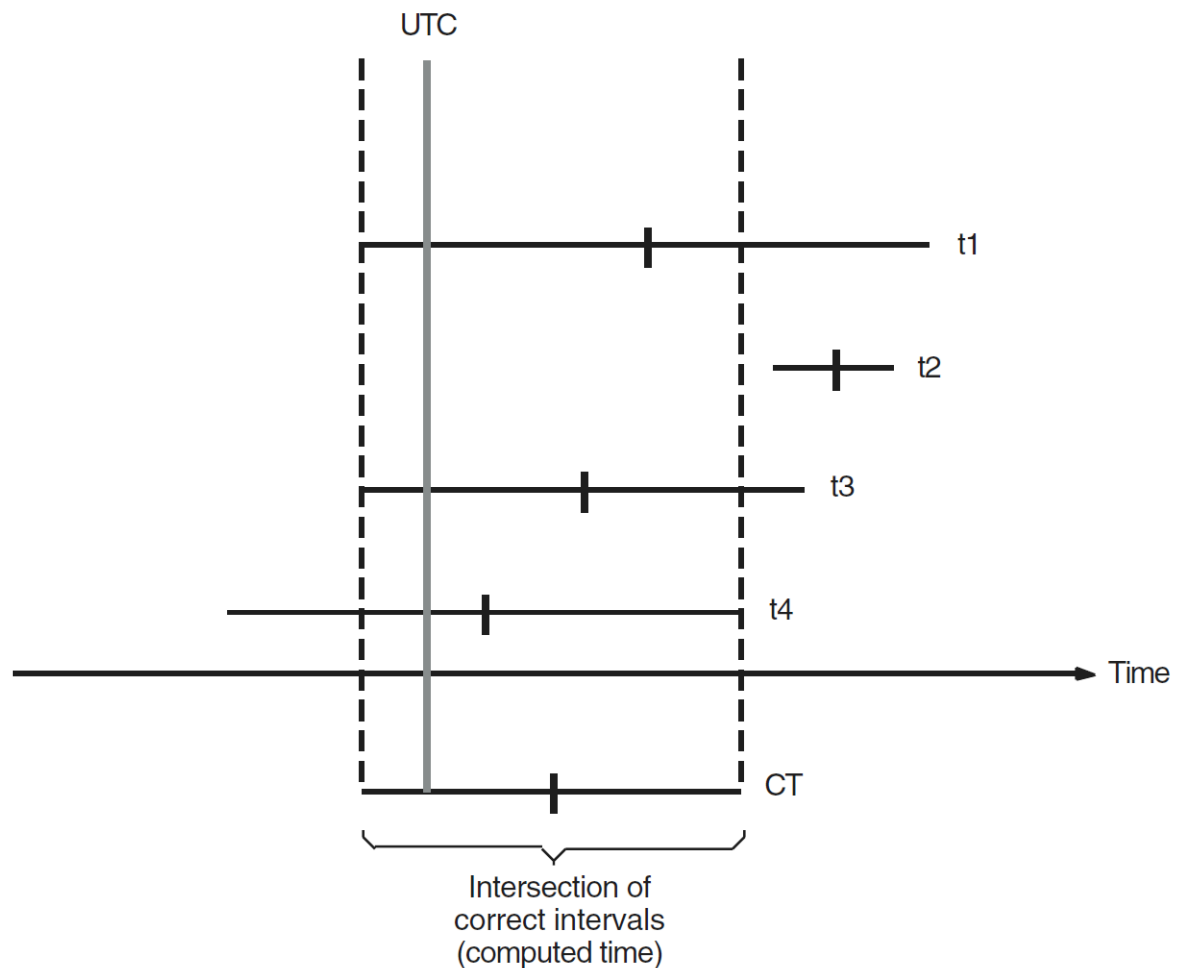
Most network systems run the DECdts clerk process. Clerks cannot have time-providers, and they do not use the system time of their client systems to compute new times. When a clerk is synchronizing its client system's clock, the clerk uses only the time values that it obtains from servers to compute a new system time.

When a DECdts clerk or server process requests several time intervals from servers, it uses them to calculate a new time that is correct (contains UTC) and that minimizes inaccuracy. When the servers respond and the DECdts entity has calculated network communications uncertainties for each of the time values, the entity has a set of intervals (t1 through t4 in *Figure 1.2, "Computed Time"*). Intervals t1, t3, and t4 are correct; because each of these intervals contains UTC, their intersection is the smallest interval the entity can choose that also contains UTC. This intersection is the **computed time**. The DECdts process uses the computed time interval to adjust the clock on the system that receives the server values.

In addition to eliminating large inaccuracy values during synchronization, DECdts also discards intervals received from potentially faulty clocks (t2 in *Figure 1.2, "Computed Time"*). DECdts detects and rejects time intervals that do not intersect with the majority of the intervals. When DECdts detects a nonintersecting interval, it notifies the system manager by displaying an event message identifying the server that sent the interval.

A server that has a high-drift clock or is far away in the network submits its time to the DECdts entity (t_1 in *Figure 1.2, "Computed Time"*), but the large time interval is ignored because more accurate times are available. Note that in *Figure 1.2, "Computed Time"*, the midpoint of correct time t_1 is further from the computed time than that of the interval that is declared nonintersecting (t_2).

Figure 1.2. Computed Time



ZK-3855A-GE

During the synchronization process, servers with the best accuracy have the most influence in determining new system times throughout the network. In *Figure 1.2, "Computed Time"*, the server that submitted time value t_3 has the smallest correct interval, and is therefore the closest to the computed time. Server systems with external time-providers are usually the servers with the most accurate times; beyond TP servers, those servers with the highest quality clocks and best communications links tend to influence the time on other systems to the greatest degree.

The synchronization process also reduces the skew between systems. The computed time interval is often smaller than the interval supplied by any single clock. Note that the computed time in *Figure 1.2, "Computed Time"* is a smaller interval than any of the source intervals. As the synchronization procedure is constantly repeated on each network system, the skew between systems is reduced and they are more closely synchronized. (However, if a time-provider is absent from the network, the clocks may collectively drift away from UTC.)

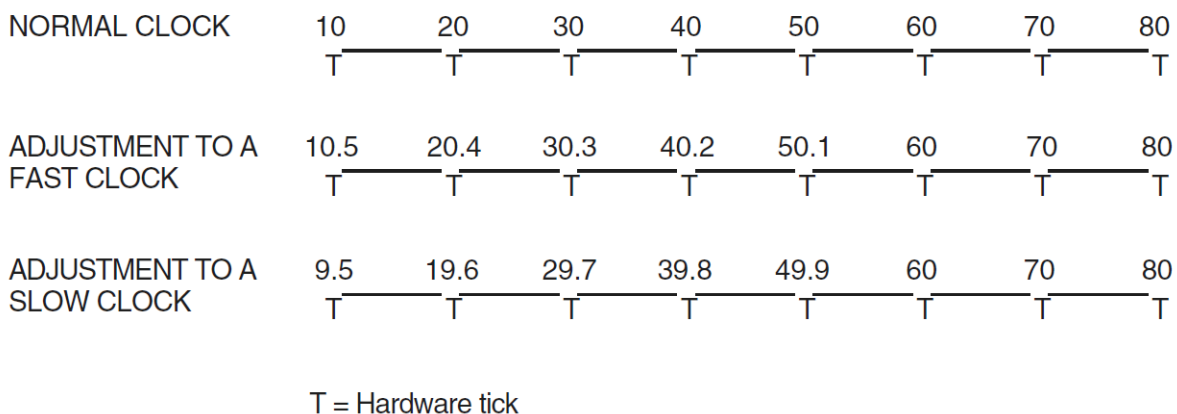
1.2.4. How DECdts Adjusts System Clocks

Each system clock in your network is based on an oscillator and operates with a combination of hardware and software. The clock's hardware contains a timer that sends interrupts to the operating system at fixed intervals; each interrupt is a single **tick**. A software register that contains the current value of the time is incremented by a fixed amount (for example, 10 milliseconds) at each tick. DECdts adjusts the rate of the clock by changing only the incremental value that is added to the software register; DECdts does not directly affect the ticks of the hardware clock. The management attribute **clock resolution** displays the tick length of a system in nanoseconds.

DECdts adjusts system clocks at the rate of 100 to 1: it requires 100 time units to adjust 1 time unit of error. For example, it takes 100 seconds to correct a 1-second error. This rate of adjustment exceeds the normal rate of drift, so that synchronization is carried out without further significant interference from the clock.

Figure 1.3, "Adjustment of the Clock" illustrates how DECdts changes the increment to the software register. The top line represents a 10-millisecond increment to the normal clock at every 10-millisecond tick. The middle line illustrates the adjustment to a fast clock: DECdts slows the clock by incrementing the register by 9.9 milliseconds instead of 10 at each tick. The bottom line illustrates the adjustment to a slow clock: DECdts speeds it up by incrementing the register by 10.1 milliseconds instead of the usual 10 at each tick.

Figure 1.3. Adjustment of the Clock



ZK-2323A-GE

It is occasionally preferable to set the system clock immediately, rather than adjusting it gradually. DECdts provides this option for the following situations:

- During system startup when you want to set the initial system time.
- If it has been a long time since the last synchronization, and you decide that the skews between system clocks are too large to wait for a gradual adjustment.
- When a network has had catastrophic hardware problems, causing a large number of the clocks to become faulty.
- When the time interval for a given clock does not intersect with the intervals of other clocks, and the error exceeds a predetermined tolerance.

Note that this feature disrupts monotonicity, and it may affect other distributed applications that depend upon DECdts.

1.2.5. DECdts Time Representation

Coordinated Universal Time (UTC) is the international time standard that has largely replaced Greenwich Mean Time (GMT). The standard is administered by the International Time Bureau (BIH), and is in widespread use. For all its internal processes, DECdts uses opaque binary timestamps representing UTC. You cannot read or disassemble a DECdts binary timestamp; the DECdts applications programming interface (API) allows applications to interpret, convert, or manipulate the timestamps. DECdts also translates the binary timestamps to or from ASCII text for display on a client system.

Although DECdts uses UTC time internally, it converts UTC to the local time for display on a client system. On OpenVMS VAX systems, you can determine the offset of the displayed time from UTC by selecting a time zone when you install the DECdts software. After the initial software installation, you can subsequently display the local time for any system with the appropriate NCL command. See *Chapter 2, "Managing DECdts"*, and *Chapter 4, "DECdts Command Dictionary"*, for further information on displaying times.

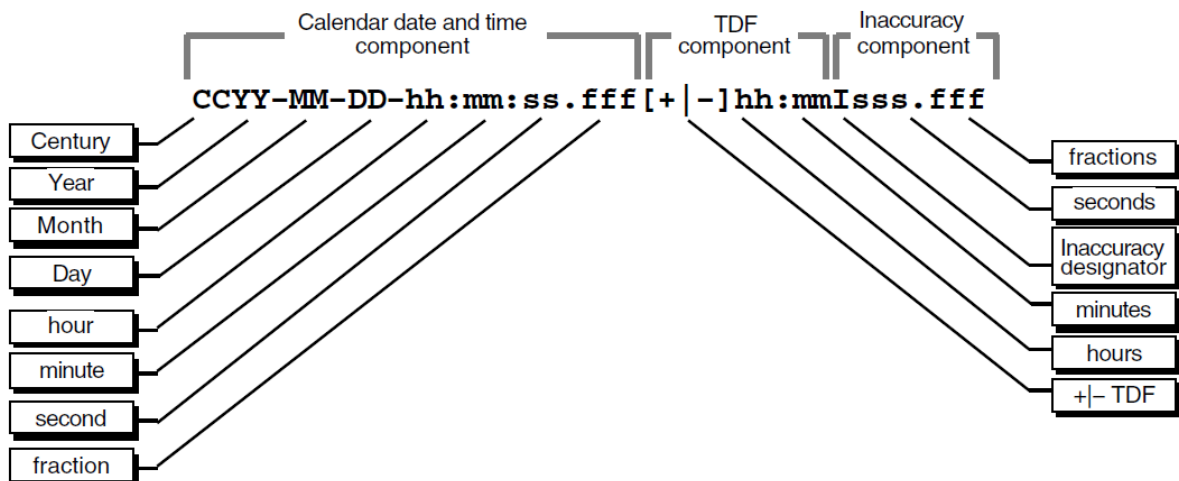
1.2.5.1. Absolute Time Representation

An **absolute time** is a point on a time scale. For DECdts, absolute times reference the UTC time scale; absolute time measurements are derived from system clocks or external time-providers. When DECdts reads a system clock time, it records the time in an opaque binary timestamp that also includes the inaccuracy and other information. When you use the DECdts management interface to display an absolute time, it is converted to ASCII text as shown in the following example:

```
2019-11-21-13:30:25.785-04:00I000.082
```

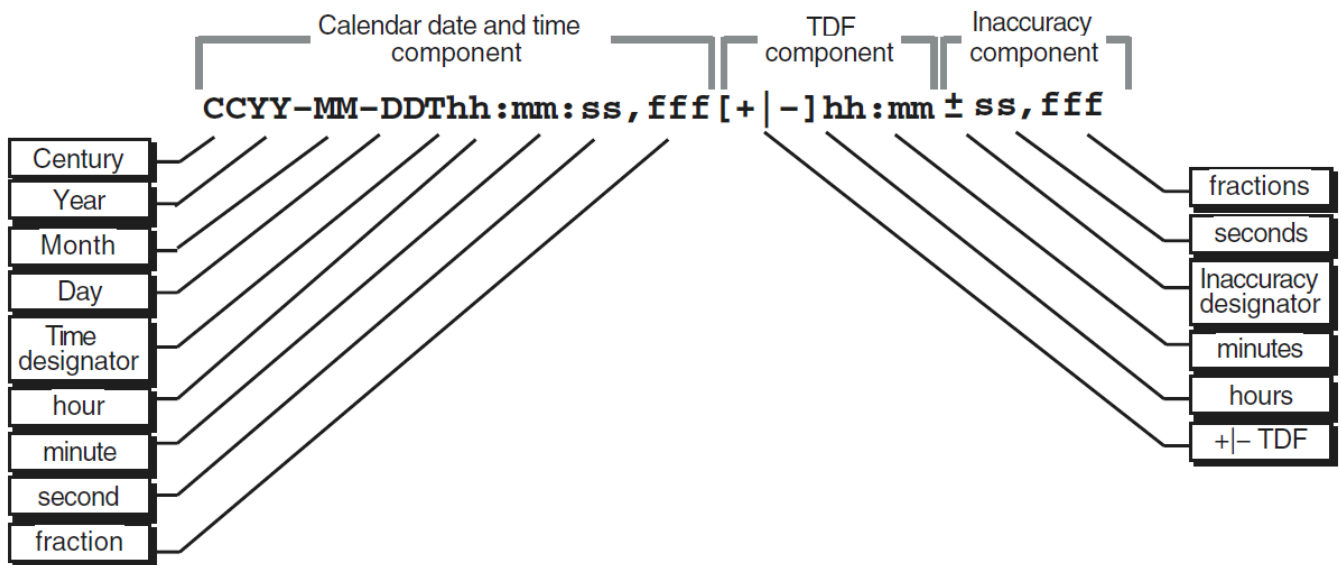
DECdts displays all times in a format that complies with the International Organization for Standardization (ISO) Standard 8601 (1988). The ISO format that generated the previous display example is detailed in *Figure 1.4, "Time Display Format"*.

Figure 1.4. Time Display Format



In *Figure 1.4, "Time Display Format"*, the relative time preceded by the plus (+) or minus (–) character indicates the hours and minutes that the calendar date and time are offset from UTC. The presence of this **time differential factor (TDF)** in the string also indicates that the calendar date and time are the local time of the system, not UTC. The character `I` indicates the beginning of the inaccuracy component associated with the time.

Although the DECdts management interface displays all times in the previous format, the interface also accepts the variations to the ISO format shown in *Figure 1.5, "Time Display Format Variants"*.

Figure 1.5. Time Display Format Variants

ZK-4068A-GE

In Figure 1.5, "Time Display Format Variants", the delineator T separates the calendar date from the time, a comma separates seconds from fractional seconds, and the plus or minus character indicates the beginning of the inaccuracy component.

DECdts offers a translation feature that changes UTC-based absolute times to your local time whenever the time is displayed. The local time displayed is derived from UTC and a TDF, which can have a positive or negative value. In the previous example, the string ([+][-] hh:mm) denotes the TDF. When installing a system, you select a time zone rule for the system, which determines the TDF and any seasonal changes to the TDF. After the initial startup, all subsequent output times reflect the local time. If an absolute time displayed by your system does not contain TDF information, it is a UTC time.

1.2.5.2. Relative Time Representation

A **relative time** is a discrete time interval that is usually added to or subtracted from another time. The TDF associated with absolute times is an example of a relative time; relative times are normally used as input for commands or system routines. The following example shows a relative time used in a DECdts command:

```
21-08:30:25.000I00.300
```

Note that relative times do not use the calendar date fields, since these fields concern absolute time. Positive relative times are not signed, while negative relative times are preceded with the minus sign.

Relative times are often subtracted from or added to other relative or absolute times. For example, if you say "I will meet you in an hour," you add a relative time of +01:00 to the present, absolute time. When you add or subtract a relative time and an absolute time, the inaccuracy of the absolute time is carried over to the resulting absolute time. For example, 2019-11-30-00:30:25.000I00.030 minus 00-00:15:25.000I00.000 equals 2019-11-30-00:15:00.000I00.030.

The relative times that DECdts uses internally are opaque binary timestamps. Through its API, DECdts offers several calls to calculate new times using relative binary timestamps.

1.3. How DECdts Works

DECdts has two major software components:

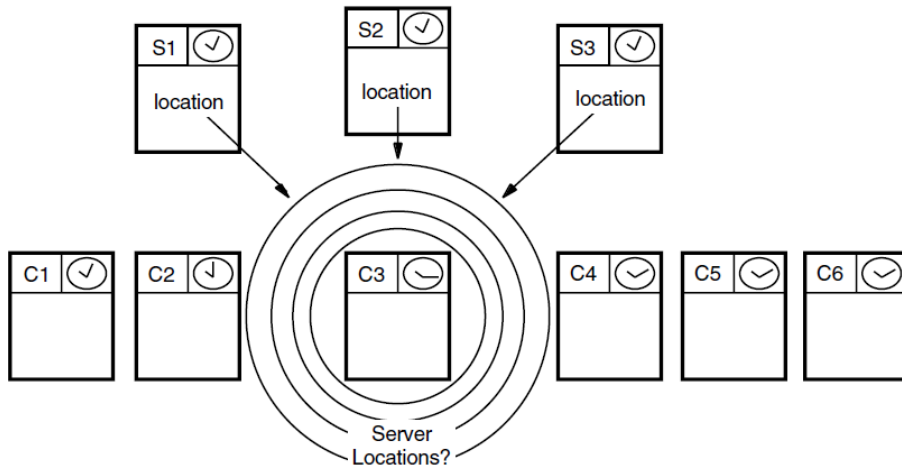
- Clerks
- Servers

The following sections describe each component and explain how they interact to provide time to client applications and to synchronize system clocks.

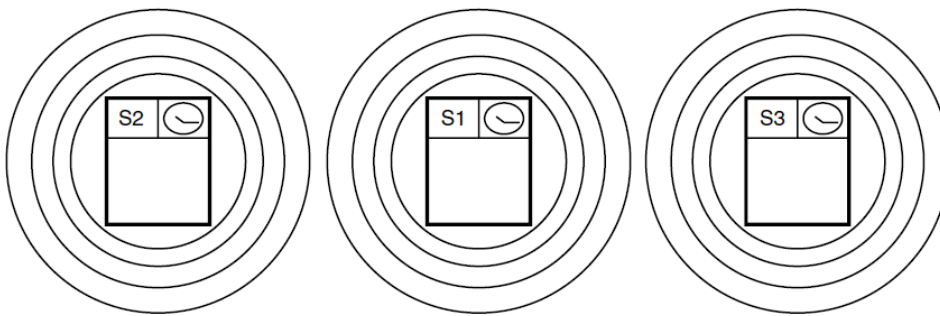
1.3.1. Clerks

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 also contact known servers for new server lists 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.)

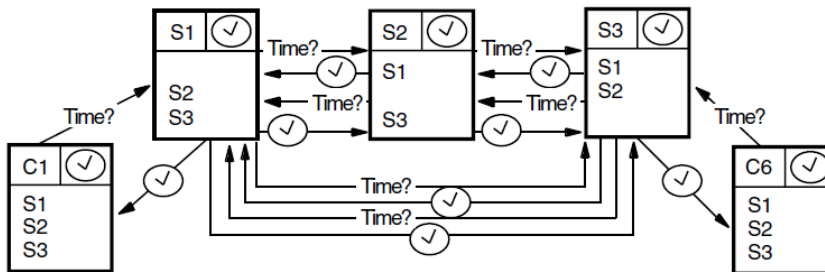
After building a server list, a clerk can directly request time values from several of the servers on the list. The clerk then receives these time values and uses them to compute a new system time for its client system. *Figure 1.6, "DECdts Operation"* illustrates how DECdts builds server lists, passes time values around the network, and synchronizes system clocks.



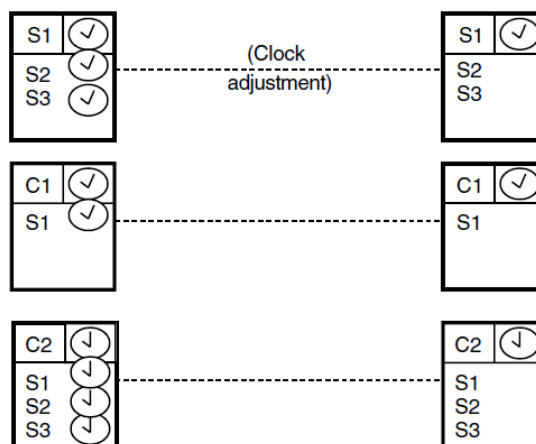
1A Each clerk solicits server locations via multicast messages; servers listen and respond.



1B Each server broadcasts its location via multicast messages; other servers listen.



2 Each clerk queries 1–3 servers for time values; each server queries all local servers for time values.



3 Each server and clerk computes a new system time from the values received and adjusts its clock.

ZK-2764A-GE

1.3.2. Servers

Servers provide many of the communications functions for DECdts.

External time-providers can be connected to servers, which then propagate the precise time intervals they obtain from the time-providers throughout the network.

Before one server can obtain time values from another, the servers must have the same **epoch** number. Epochs divide the DECdts implementation into logically separate areas; servers synchronize only with other servers that have the same epoch number. All servers have the same epoch number when they are created. Infrequently, you may wish to change a server's epoch number (using the management interface) to isolate it from the network to correct a problem.

The rest of this section describes the three types of DECdts servers: local servers, global servers, and couriers.

1.3.2.1. 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 (Ethernet) protocol to communicate; because of the high throughput on this type of network, the skews between the local servers on the LAN are normally maintained at under 200 milliseconds.

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.

Note

DECdts servers listen for multicast messages containing the following (DECdts) multicast address: **09-00-2B-02-01-02**. Servers ignore any multicast messages containing addresses for other products.

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. At longer intervals, clerks request time values from the local servers. Clerks, however, need only request intervals from the number of servers determined by the `servers_required` management parameter, which is usually a subset of all the local servers.

1.3.2.2. 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.

DECdns is optional; if you have installed DECdns, it maintains a list of the DECdts global servers that are registered with the `advertise` command. 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.

If you use the namespace or the Domain Name System (DNS/BIND) rather than DECdns, DECdts systems use an alternate method of configuring and locating global servers. For information on configuring global servers using the Local namespace or DNS/BIND (referred to in DECnet-Plus as the Domain namespace), see the file `sys$common:[sysmgr]dtss$config_template.dat`. For information about the Local and Domain namespaces and about node name syntax using these namespaces, see the *DECnet-Plus for OpenVMS Installation and Basic Configuration Manual*.

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 *Section 1.3.2.3, "Couriers"* for a description of these servers.

1.3.2.3. 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 synchronizations that occur on the LAN, couriers randomly select a global server from before each synchronization. Couriers use the responses of both local and global servers when synchronizing their own clocks.

Couriers provide networkwide synchronization through the following procedure:

1. Couriers request time values from at least one global server and all of the local servers.
2. Couriers use the global server times and local server times to synchronize the clocks in their systems.
3. Couriers relay newly computed clock times to other servers and clerks on the LAN during future synchronizations.

For a network containing multiple LANs or point-to-point links, one server on each LAN or segment should be configured 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.

1.4. Interoperation with DECdns

The DECdns (DECdns) provides a networkwide registry called a **namespace**, which stores the names and attributes of many types of network resources. DECdts global server names are maintained as objects in the DECdns namespace in the default directory `ns : .DTSS_GlobalTimeServers`, where `ns` is the name of the default namespace. For further information on DECdns and namespace objects, see the *VSI DECnet-Plus for OpenVMS DECdns Management Guide*.

Chapter 2. Managing DECdts

This chapter describes how to manage the VSI DECnet-Plus Distributed Time Service (DECdts) using the Network Command Language (NCL) and the NCL management interface. The chapter begins with a brief introduction to basic DECdts command syntax and procedures. The remainder of the chapter describes how the commands are used to set the software variables and monitor the DECdts implementation.

2.1. The DECdts Management Interface

DECdts uses the Network Command Language (NCL) for all configuration and management functions. NCL provides consistency to the commands for VSI's distributed system and networking products, enabling users to manage different layered and bundled software with the same commands and syntax. This section explains how to access and use the NCL command interface for DECdts. For further information on NCL, refer to the *VSI DECnet-Plus for OpenVMS Network Control Language Reference Guide*.

2.1.1. Interface Components and Command Syntax

DECdts conforms to VSI's Enterprise Management Architecture (EMA). Each product that uses the architecture is assigned to a **module**. Each module is made up of one or more **entities**, which control different functional areas of each module. When you issue an NCL command, key in the directive, the keyword `node` and the node ID (optional for the local node), the entity name, and an attribute/value that is the object of the command. For some commands, you can also provide an argument or prepositional phrase that modifies the command or redirects its output. See *Chapter 4, "DECdts Command Dictionary"*, DECdts Command Dictionary, for detailed information on the NCL commands and syntax you can use to manage DECdts. A sample command is shown in *Figure 2.1, "Command Syntax"*.

Figure 2.1. Command Syntax

```
ncl> set node my_node dtss servers required 3
```

Command Node Node-ID Entity [Attribute] [Argument] Value

ZK-2763A-GE

Note

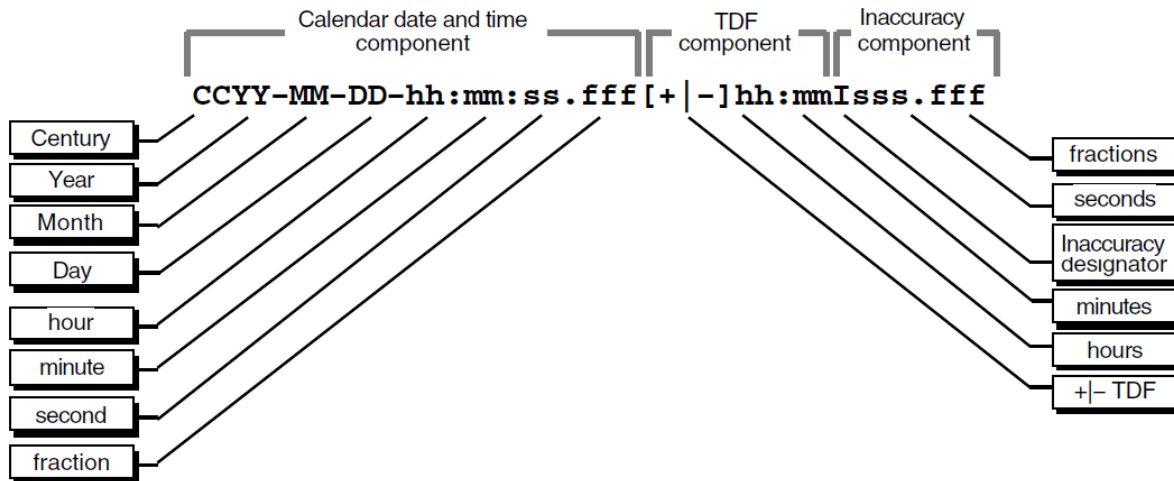
The DTSS module manages the DECdts software; you must enter the appropriate DTSS entity name in all DECdts management commands.

You can enter DECdts commands in uppercase, lowercase, or any combination of uppercase and lowercase, and you can shorten any command or keyword to its minimum unique abbreviation of three characters. Usage examples are provided in the reference pages for each command. You can spawn a subprocess from a command procedure or program to issue the commands.

2.1.2. DECdts Timestamp Syntax

Once you begin using the command interface to manage DECdts, you will notice that all responses to commands contain a timestamp that conforms to the input and output format shown in *Figure 2.2, "DECdts Timestamp Format"*.

Figure 2.2. DECdts Timestamp Format



A typical DECdts time display looks like this:

```
2019-03-16-14:29:47.520-05:00I000.003
```

The timestamp uses the DECdts format that is explained in *Chapter 1, "Introduction to the VSI DECnet-Plus Distributed Time Service"*. In the previous example, the year is 2019, the day is March 16, and the local time is 14 hours, 29 minutes, and 47.52 seconds. A negative time differential factor (TDF) of 5 hours and an inaccuracy of 3 milliseconds are included in the timestamp.

2.1.3. The DECnet Startup Script

The `net$configure` configuration procedure does not generate the DECdts NCL command file; it is installed when the DECnet-Plus software is installed.

The DECdts startup procedure runs as a detached process. The procedure follows:

1. The DECnet startup process in `sys$startup:net$startup.com` calls the DECdts startup process in `sys$startup:dtss$startup.com`.
2. The `dtss$startup` process activates the detached process `sys$system:dtss $service.exe` and invokes either `sys$common:[sys$startup]net $dtss_clerk_startup.ncl` or `sys$specific:[sys$startup]net $dtss_server_startup.ncl`, depending on whether you have installed the server files at installation.
3. At system startup, the `clerk_startup` or `server_startup` process automatically executes the create and enable commands through the NCL management interface.

To further customize the system's DECdts configuration at startup, you can edit the default commands supplied in the startup files, add some of the commonly used commands described in *Section 2.2.2, "Enabling DECdts"*, or add other commands described later in this chapter.

2.1.4. Accessing and Exiting the Management Interface

You can use either of two methods to start the NCL command interface and access the NCL prompt. The first method follows:

```
$ run sys$system:ncl
ncl>
```

You can also access the NCL command interface using a second method:

```
$ mcr ncl
ncl>
```

After you start the management interface and the NCL prompt is displayed, you can begin to enter commands. (See *Section 2.2, "Creating and Enabling DECdts"* for instructions on creating and enabling DECdts.) Three NCL entities manage the DECdts software:

- The `dtss` entity
- The `dtss decnet global server` entity
- The `dtss decnet local server` entity

Of the three entities, only the `dtss` entity has settable characteristics. Only the `show` command is available for the `dtss decnet global server` and `dtss decnet local server` entities.

When you finish a management session, exit from the command interface and access the normal system prompt by entering the following command:

```
ncl> exit
%
```

2.2. Creating and Enabling DECdts

Once the operating system is functioning on your system and you have accessed the NCL prompt, you can enter DECdts management commands. If you have just completed the DECdts installation and startup procedures for the system, the DECnet startup script executes the `create` and `enable` commands for you. If you are restarting DECdts or have altered the startup script, you may need to create and enable DECdts. This section describes these steps; *Section 2.2.2, "Enabling DECdts"* also describes some common commands you may want to execute immediately after enabling the service.

2.2.1. Creating DECdts

You cannot activate the DECdts process by entering the `create` command interactively. To configure a system as a clerk or server, you must install the desired clerk or server startup file and execute it. See *Section 2.1.3, "The DECnet Startup Script"* for further details.

If your network has extended LAN or WAN links, establish where to install couriers during the planning process. After you create a system as a server, you can designate it as a courier by entering the following command. The available roles are `courier`, `noncourier`, and `backup courier`.

```
ncl> set node splsys dtss courier role courier
```

Once you create all the servers, clerks, and couriers in your network, proceed to enable them for the first synchronization.

2.2.2. Enabling DECdts

Use the `enable` command to turn on the DECdts software. When you enter the command, the status attribute is set to `on`. The `set clock` argument is available to abruptly set the clock to the computed time following the first synchronization. If this is the initial system startup, set the argument's value to `true` to permit faster synchronization with the other systems.

Note

If you are enabling the first server on a LAN and clerks are already operating on the LAN, ensure that the system clock is within 5 minutes of UTC before entering the `enable dtss set clock true` command. If the clock has more than 5 minutes of inaccuracy, clerks may synchronize to the incorrect time and interfere with DECdns operation.

If the system's clock is close to UTC, you can enter the `set clock false` argument (the default setting). This setting dictates a gradual clock adjustment to the computed time. Gradual adjustment prevents the clock from jumping forward or backward. The adjustment rate is 100 to 1 (it takes 100 units of time to correct a 1-unit error), so adjusting the clock may be inappropriate for correcting an error of hours or days.

The following example shows how to enter the `enable` command:

```
ncl> enable node splsys dtss set clock true
```

The DECdts systems in your network locate and synchronize with each other regardless of the order in which you enable them. Enabling servers first, however, allows the systems to synchronize faster.

Note that the initial synchronization may take longer than subsequent synchronizations, because it depends on the presence of other systems. The system on which you are enabling DECdts uses its hardware clock value until the initial synchronization is completed.

2.2.3. Controlling LAN Devices Used by DECdts

The static device tables formerly used to determine the devices used by DECdts have been removed. Now, DECdts uses the `$DEVSCAN` and `$GETDVI` system services to build a list of devices that have the following characteristics:

- a device class of `DC$SCOM` (synchronous communication device)
- a device characteristic of `DEV$V_NET`
- a device status of `UCB$V_ONLINE` and `UCB$V_TEMPLATE`
- a device name in the form `_ddcn:`

You can use the logical name `DTSS$ETHERNET_DEVICE` to provide a list of devices that DECdts should NOT use. All devices must be in the form `_ddcn:`, where `dd` is the OpenVMS physical device name, `c` is the controller letter, and `n` is the port number. The string is limited to 255 characters and can contain spaces and other text which is ignored by DTS. For example, the following two commands tell DECdts not to use the `_EIA0:` and `_FWA0:` devices.

```
$ DEFINE/SYSTEM DTSS$ETHERNET_DEVICE "_EIA0:_FWA0:"
```

Note

Unterminated network adapters can cause the dynamic device recognition process to hang. Either terminate all network adapters or include any unterminated devices in the DTSS\$ETHERNET_DEVICE logical definition.

2.2.4. Designating Global Servers and Couriers

If your network has WAN links or is an extended LAN, you may need to use global servers and couriers to synchronize the systems in separate network segments. Once you create and enable all the servers and clerks in the network, you can assign global server and courier roles to selected local servers. Refer to the *DECnet-Plus for OpenVMS Installation and Basic Configuration Manual*, *VSI DECnet-Plus for OpenVMS Introduction and User's Guide*, and *VSI DECnet-Plus Planning Guide* for advice on planning the location of global servers and couriers.

2.2.4.1. Advertising Global Servers to DECdns

To assign a server to the global set of servers, you can advertise the server to the DECdns namespace. The DECdns namespace contains a directory that is the repository (DECdns) for the name and address information that DECdts uses to locate systems in extended LANs or WANs. If you have installed DECdns, DECdts can call it at any time to obtain the locations of systems that it cannot reach on the LAN.

The following example shows how to advertise a server to register it with DECdns as a global server. Enter the system's DECdns simple name as the argument to `server name`. DECdns translates the simple name to a full name when it adds the server to the time server directory.

```
ncl> advertise dtss server name acctg
```

See the `advertise dtss` command in *Chapter 4, "DECdts Command Dictionary"* for information on removing a server name from the global time server directory.

2.2.4.2. Designating Global Servers Using the BIND Database

DECdts supports getting information on the global servers from the BIND database. If you want to use this feature of DECdts, you must use the conventions discussed in this section.

You must set up canonical name (CNAME) records for each of the global servers you intend to use. These records provide an alias for looking up the DECdts global servers. These names are of the general form:

```
gbl_dtssn IN CNAME name
```

In this syntax, the `n` value is an integer. These integer values must start at 1 and can go to some arbitrarily large number. The integer values must be contiguous (no gaps). The reason is that DECdts stops scanning for the next higher integer if it fails to find a record. The name value must be the *name* of an address (A) record appearing in the BIND database.

A small sample BIND data base for the `crosspoint.com` domain is shown below. There are two global servers defined in it: `wzbang` and `case`.

```
$ type CROSSPOINT_COM.DB
```

```
$ORIGIN crosspoint.com.
@ IN SOA wzbang.crosspoint.com. system.crosspoint.com. (
1 ; Serial
10800 ; Refresh
3600 ; Retry
604800 ; Expire
86400 ) ; Minimum
;
IN NS wzbang.crosspoint.com.
;
case IN A 161.114.94.62
wzbang IN A 161.114.94.69
gbl_dtss1 IN CNAME wzbang
gbl_dtss2 IN CNAME case
```

When configuring the BIND resolver, you must pay attention to the domain that the BIND server is in (especially if it is different from the local domain). You should set the system domain to be the same domain of the BIND server that you want to access. To change the system domain, use the `set config` command. The `set config` command makes the change in the permanent database so that the data survives reboots. For example, to set the domain to the `crosspoint.com` domain, use the following command:

```
$ tcpip
TCPIP> set config name_service/domain=crosspoint.com
```

You can verify the resolver setting using the following command:

```
TCPIP> show name_service
BIND Resolver Parameters
Local domain: DOM1.ASHFIELD.NET
System
State: Started, Enabled
Transport: UDP
Domain: crosspoint.com
Retry: 4
Timeout: 4
Servers: wzbang.crosspoint.com
Path: No values defined
Process
State: Enabled
Transport:
Domain:
Retry:
Timeout:
Servers:
Path:
TCPIP>
```

Prior to doing a synchronization, DECdts extracts the information about global DECdts servers from the BIND server and stores it in the temporary file `sys$manager:dtss$bind_data.tmp`. The data is subsequently loaded into DECdts in the in-memory global set. The data in this temporary file is in configuration file format. DECdts generates only one of these files at a time and deletes the old file prior to accessing the BIND server to get new data.

You can check to see if DECdts synchronized properly with a global server by issuing the following command:

```
NCL>show dtss decnet global servers * all
```

```
Node 0 DTSS Decnet Global Servers wzbang.crosspoint.com
at 2004-03-11-04:10:40.130+10:00I0.410
Identifiers
Name = wzbang.crosspoint.com
Status
Last Time Polled = 2004-03-11-04:10:27.417+10:00I0.533
Last Observed Time = 2004-03-11-04:10:27.430+10:00I0.379
Last Observed Skew = +0-00:00:00.013I0.911
Used In Last Synchronization = True
Characteristics
Transport = DECnet
Node 0 DTSS Decnet Global Servers case.crosspoint.com
at 2004-03-11-04:10:40.130+10:00I0.410
Identifiers
Name = case.crosspoint.com
Status
Last Time Polled = 2004-03-11-04:10:27.417+10:00I0.533
Last Observed Time = 2004-03-11-04:10:27.392+10:00I0.468
Last Observed Skew = -0-00:00:00.025I0.1000
Used In Last Synchronization = True
Characteristics
Transport = DECnet
NCL>
```

The previous display confirms that DECdts synchronized with both case and wzbang.

Troubleshooting

If DECdts does not synchronize with the global servers, verify that the `sys$manager:dtss$bind_data.tmp` file is present. If the file exists, type it out to see if the data looks valid. The following is an example of a valid file:

```
Global DECNET DOMAIN:wzbang.crosspoint.com
Global DECNET DOMAIN:case.crosspoint.com
```

If the file has no data or missing data you may have set up the canonical name (CNAME) records for the global DECdts servers incorrectly.

If the temporary file data looks correct, you can turn on DECdts tracing in `sys$startup:dtss$startup` by uncommenting the following line:

```
$ ! To turn on DTSS tracing (output to DTSS$ERROR.LOG),
$ ! uncomment the following line(s):
$ ! Define/system DTSS$TRACE 65535
```

After modifying the `dtss$startup` file, restart the DECdts server. The trace log in the `dtss$error.log` file should have some lines referring to the BIND server. This should give you an indication what is going wrong.

2.2.4.3. Designating Global Servers with the Site-Specific Template

If you have not installed DECdns in your network, you cannot use the `advertise` command to designate a system as a global server. As an alternate means of designating global servers, a site-specific template is supplied with the DECdts kit. You can edit the template at each site to add a list of global server entries and then rename the template file. Once you rename the file, the DECdts process on the related system uses the information in the file to locate global servers rather than attempting to contact the DECdns clerk software.

The site-specific global server template is `sys$common:[sysmgr]dtss$config_template.dat`. The comments in the file describe how to add global server entries and rename the file.

2.2.4.4. Assigning the Courier Role to Servers

Couriers serve an important role in maintaining synchronization between the systems in separate parts of your network. A courier server functions like any other server in the local set, but a courier also requests a time value from at least one global server at every synchronization. This procedure enables a courier server to propagate times from remote systems to a LAN, thereby keeping the LAN in synchronization with all the other parts of the network.

There are three courier roles that you can assign to a server:

- `backup courier` (the default setting)
- `courier`
- `noncourier`

The default role for the `courier role` characteristic is `backup courier`. If a server system is connected to a time-provider, it configures itself with the `noncourier` role after the first synchronization, overriding any setting you supply. A server connected to a time-provider cannot have the courier role, because the server process on that system requests only time values from the time-provider.

Use the `courier` setting to designate a server as the primary link to other portions of your network. Use the `backup courier` setting to designate a server as a secondary link to other areas of the network. A backup courier is effective only if no other courier is available on the LAN.

Because there are no significant processing or overhead penalties associated with the backup courier role, `backup courier` is the default setting. You can designate one of the servers on a LAN as a courier and designate all the other servers on the LAN as backup couriers. If you configure several servers as backup couriers and the courier is unavailable, the backup courier with the lowest numbered Ethernet address becomes the effective courier.

The following example shows how to assign the courier role to a server:

```
ncl> set node splsys dtss courier role courier
```

2.2.5. Changing the DECdts Entity Type

Once your network is running DECdts, you may decide you have too many or not enough servers on a given LAN. Rather than adding unnecessary servers or clerks to the LAN, you can change a clerk to a server or a server to a clerk.

The procedure to change the DECdts entity type varies according to whether the system is a standalone node or part of a cluster. The following procedure turns a standalone clerk to a server or changes all of the clerk satellites in a cluster to servers; for the clustered environment, run the procedure on any one of the clerks.

1. Enter the following commands for the clerk system:

```
ncl> disable dtss  
ncl> delete dtss
```

2. Rename the clerk startup file to the server startup file as follows:

```
$ rename net$dtss_clerk_startup.ncl net$dtss_server_startup.ncl
```

3. Edit the file `sys$common:[sysmgr]net$dtss_server_startup.ncl`; change the type setting in the create command from `clerk` to `server`.
4. If you plan to run a time-provider, you can install the time-provider examples in the system's `sys$examples` area by entering the following DCL command:

```
$ backup/log dnvosi010.n/save sys$examples:
```

5. Start the server by entering the following DCL command:

```
$ @sys$startup:dtss$startup
```

If you want to change a cluster satellite that is a clerk to a server, without changing the other members of the cluster, enter the following command in place of Step 3:

```
$ rename net$dtss_clerk_startup.ncl-
_$ sys$specific:[sysmgr]net$dtss_server_startup.ncl
```

You can reverse the previous procedure to change servers to clerks; edit the file `net$dtss_server_startup.ncl` (found in `sys$common:[sysmgr]`) to change the create argument from `server` to `clerk`. Rename the file to `sys$common:[sysmgr]net$dtss_clerk_startup.ncl`.

2.2.6. Option to Disable DECdts at System Boot

You can disable DECdts when the system boots. You should use this option when another time provider controls the system clock.

To disable DECdts, do the following:

1. Add the following line to the `sylogicals.com` file:

```
$ define/system net$disable_dtss 1
```

2. Reboot the system.

The `NET$DISABLE_DTSS` logical name causes DECdts to leave the time information set by the operating system unchanged.

2.3. Summary of Management Commands and Characteristics

There are several commands and characteristics you can use to manage DECdts. This section contains a summary of the management commands and characteristics that apply to DECdts servers and clerks. *Table 2.1, "DECdts Commands"* contains a list of these commands.

Table 2.1. DECdts Commands

Servers	Clerks	Arguments
DTSS Entity		
advertise		server name
change		epoch, time

Servers	Clerks	Arguments
create	create	type
delete	delete	
disable	disable	
enable	enable	set clock
set	set	
show	show	
synchronize	synchronize	set clock
update		time
DTSS DECnet Global Server Entity		
show		
DTSS DECnet Local Server Entity		
show		

In the DTSS portion of *Table 2.1, "DECdts Commands"*, note that the `set` command does not have arguments, but it does have characteristics that accompany the command to modify its function. See *Table 2.2, "Settable DTSS Characteristics"* for the characteristics you can use with the `set` command. The characteristics are fully described in this chapter and in *Chapter 4, "DECdts Command Dictionary"*, the DECdts Command Dictionary.

The `show` command does not have arguments, but you can specify one or more attribute groups with the command; that is, characteristics, counters, identifiers, or status. See the `show` command in *Chapter 4, "DECdts Command Dictionary"* for the attributes and attribute groups you can append to the command.

Table 2.2. Settable DTSS Characteristics

Servers	Clerks
advertisement interval	
automatic tdf change	automatic tdf change
check interval	
courier role	
error tolerance	error tolerance
global directory	global directory
lan timeout	lan timeout
maximum inaccuracy	maximum inaccuracy
query attempts	query attempts
servers required	servers required
synchronization hold down	synchronization hold down
wan timeout	wan timeout

Note that several commands and characteristics apply only to servers. If you enter a server-related command or characteristic for a clerk, DECdts displays an error message similar to the following:

```
Node 0 DTSS
```

```
AT 2019-09-11-17:00:35.541-04:00I1.847
```

```
FAILED IN DIRECTIVE: CHANGE  
DUE TO: Directive NOT Supported
```

2.4. Routine Management Tasks

Once you create and enable DECdts on all your network systems and the systems are synchronized, routine management tasks include performance enhancements, reconfiguration of the network, and changes to local time. This section describes how to perform these tasks.

2.4.1. Critical Commands and Characteristics

The DECdts entity has several commands and characteristics that modify and improve the performance of servers and clerks. Use the `set` command to change the values for many of the characteristics; use the `show` command to display the values of characteristics at any time.

This section explains the commands and settable characteristics you can use to manage clerks and servers after they have been installed and enabled. This section also offers suggestions for various command and characteristic settings depending on your network configuration.

2.4.1.1. The Servers Required Characteristic

The `servers required` characteristic specifies how many servers must supply time values to the DECdts entity before it can synchronize the system clock.

For servers, the default and minimum recommended value for the `servers required` characteristic is 3; servers require values from three servers to compute a reliable new time. For clerks, the default value is 1 to permit synchronization in a network with only one or two servers. Even though the default setting is 1 for clerks, VSI recommends you use a value of at least 3 if sufficient servers are available in the network. The clerk cannot detect a faulty server unless it gets time values from at least 3 servers.

Depending on whether it is a server or clerk, the entity has different requirements of the servers in the network. The following requirements reflect the default `servers required` settings.

- Clerks require a value from one server.
- Servers require values from at least two other servers. Each server uses its own clock value when computing a new time.

To change the `servers required` value, enter the `set` command with the `servers required` characteristic set to the desired value. The command accepts values from 1 to 32. For example, the following command shows how to increase the required number of servers to 4:

```
ncl> set node splsys dtss servers required 4
```

Once the DECdts nodes on your LAN have been running for more than 10 minutes, you can check the number of local servers in your system's synchronization list by issuing the `show` command for the `dtss local server` entity:

```
ncl> show dtss decnet local server *  
Node 0:. DTSS Decnet Local Server 00-00-04-00-01-fc (63.1)  
AT 2019-02-21-11:22:18.008-05:00I0.103  
Identifiers
```

```
ADDRESS                                = 00-00-04-00-01-fc (63.1)
Node 0:. DTSS Decnet Local Server 01-00-04-00-02-fc (63.2)
AT 1991-02-21-11:22:18.032-05:00I0.103
Identifiers
ADDRESS                                = 01-00-04-00-02-fc (63.2)
Node 0:. DTSS Decnet Local Server 02-00-04-00-03-fc (63.3)
AT 1991-02-21-11:22:18.036-05:00I0.103
Identifiers
ADDRESS                                = 02-00-04-00-03-fc (63.3)
Node 0:. DTSS Decnet Local Server 03-00-04-00-04-fc (63.4)
AT 1991-02-21-11:22:18.040-05:00I0.103
Identifiers
ADDRESS                                = 03-00-04-00-04-fc (63.4)
```

In this example, the output reflects the `servers required` setting of 4. This setting provides redundancy; when no global servers are in the network, the system synchronizes even if a local server becomes unavailable.

Whenever the system cannot contact the number of servers determined by the `servers required` setting, the system logs the event Too Few Servers Detected. The log includes the number of servers currently available and the number required. If you see this event displayed, see whether any of the servers are unavailable, test the communications links to ensure that the system has not been isolated from the servers, or add servers to the network.

You can use the `servers required` characteristic in other ways depending on your network configuration. See the following cases.

- If you have few systems in your network and you want to synchronize the nodes regardless of server drift, lower the `servers required` value to 1 or 2. Although the resulting synchronized time is a less reliable measure of UTC, you increase the likelihood that the systems will synchronize. If the setting is less than 3, however, the system will not be able to identify non intersecting (potentially faulty) servers.
- To increase fault tolerance and ensure that the systems compute reliable times, set the `servers required` value to 3 (the default setting for servers) or higher. The systems can then identify potentially faulty servers and compute the narrowest overlapping interval for the time values they receive. Keep in mind, however, that your system will not synchronize until there are at least three servers available.

The number of nodes in your network and the types of applications you use determine whether guaranteed synchronization or reliable times and fault tolerance are more important.

2.4.1.2. Use with Global Servers

If your network consists of more than a single LAN, it should have a set of global servers. You can create global servers by advertising local servers to the DECdns (DECdns) or by configuring the site-specific template. See *Section 2.2.4.1, "Advertising Global Servers to DECdns"* for further information on advertising a server as a global server.

The presence of global servers in your network can influence the value you choose for the `servers required` characteristic. If the number of local servers available to a system is less than the `servers required` setting, the system automatically queries DECdns for a server name from the DECdns list

of global time servers. The system then adds the global server to its server list and requests time values from the global and local servers.

You can check to see which global servers are known to a system by entering the `show` command for the entity `dtss global server`. DECdts then lists all known global servers and specifies which global servers were used in the last synchronization. The following example shows how to display all of the global servers (the asterisk is a wildcard character).

```
ncl> show dtss decnet global server * all
Node 0:. DTSS Decnet Global Server ABC:.DTSS_GlobalTimeServers.sys
AT 1991-02-21-17:34:27.358-05:00I0.092
Identifiers
    Name = ABC:.DTSS_GlobalTimeServers.sys
Characteristics
    Transport = DECnet
```

2.4.1.3. Use with Systems on Point-to-Point Lines

If you are using DECdts on a system that connects to a LAN through a point-to-point WAN link, the solitary system never has more than one local server available. If the system is a server, the default and recommended `servers required` setting is 3; the server must query two global servers to synchronize. If the system is configured as a clerk, it does not have any local servers, and must query the number of global servers determined by the `servers required` attribute to synchronize.

2.4.1.4. The Maximum Inaccuracy Characteristic

The `maximum inaccuracy` characteristic specifies the greatest allowable bound on your system's inaccuracy before DECdts causes the system to synchronize. When the system exceeds the bound determined by the `maximum inaccuracy` setting, DECdts forces the system to synchronize until the inaccuracy is reduced to a level at or below the setting. The `maximum inaccuracy` setting is a trigger for synchronization. You can vary the setting to vary the tolerance of inter-system synchronizations, but be aware that as the setting becomes lower, network overhead rises. The default setting is 0.20 second (200 milliseconds).

The effects of the `maximum inaccuracy` setting on the system's synchronization behavior is summarized in the following steps:

1. The system's clock value accumulates more inaccuracy than the `maximum inaccuracy` value and the DECdts entity initiates a synchronization.
2. The DECdts entity computes a new time value.
3. The DECdts entity adjusts the system clock, using the new time value.
4. If the new clock setting still exceeds the `maximum inaccuracy` value, or if clock drift later causes the inaccuracy to reach the value, the cycle is repeated.

Note that if synchronization repeatedly fails to achieve an inaccuracy that is less than the `maximum inaccuracy` value, the system could be continuously synchronizing. See *Section 2.4.1.5, "The Synchronization Hold Down Characteristic"* for information on how the `synchronization hold down` characteristic prevents this loop.

The default `maximum inaccuracy` value is designed to keep the system accurate enough for most applications without being intrusive to network communications or system processing. If your network includes one or more time-providers that ensure extremely low inaccuracy, you can lower the `maximum inaccuracy` value. Raise the value in the following cases:

- When a time-provider is not used in the network
- When a system is part of a WAN-only network configuration
- When the applications that call DECdts do not require the level of accuracy achieved by the default setting

The following example shows how to change the maximum `inaccuracy` value to 0.2 second:

```
ncl> set node splsys dtss maximum inaccuracy 00.200
```

2.4.1.5. The Synchronization Hold Down Characteristic

The `synchronization hold down` characteristic prevents your system from synchronizing more often than the specified interval and prevents the `maximum inaccuracy` characteristic from causing continuous synchronizations. As mentioned in *Section 2.4.1.4, "The Maximum Inaccuracy Characteristic"*, the `maximum inaccuracy` characteristic causes system synchronization as long as the system's inaccuracy is above a specified value. The `synchronization hold down` characteristic prevents synchronization from occurring more frequently than the hold down value. (The value is randomized to prevent several systems from synchronizing simultaneously and is an average value.)

The `maximum inaccuracy` and `synchronization hold down` characteristics are interdependent; system synchronization occurs automatically when *both* of the following conditions are met:

- The inaccuracy of its clock equals or exceeds the `maximum inaccuracy` value.
- The time since the last synchronization equals or exceeds the `synchronization hold down` value (slightly randomized).

Note that if the system reaches the `synchronization hold down` setting, but has not yet reached the `maximum inaccuracy` setting, the system will not synchronize.

The default `synchronization hold down` value is 5 minutes for servers and 10 minutes for clerks. If you are trying to minimize the skew between systems, you can lower the `synchronization hold down` value. For example, to make a clerk synchronize every 5 minutes if its inaccuracy reaches 100 milliseconds, enter the following command:

```
ncl> set node splsys dtss sync hold down 05:00.0000
```

The `synchronization hold down` characteristic does not prevent the `synchronize` command from working. You can synchronize a system at any time by entering this command. The `hold down` characteristic affects only automatic synchronizations triggered by the `maximum inaccuracy` characteristic. For more information on the `synchronize` command, see *Chapter 4, "DECdts Command Dictionary"*.

2.4.1.6. The Error Tolerance Characteristic

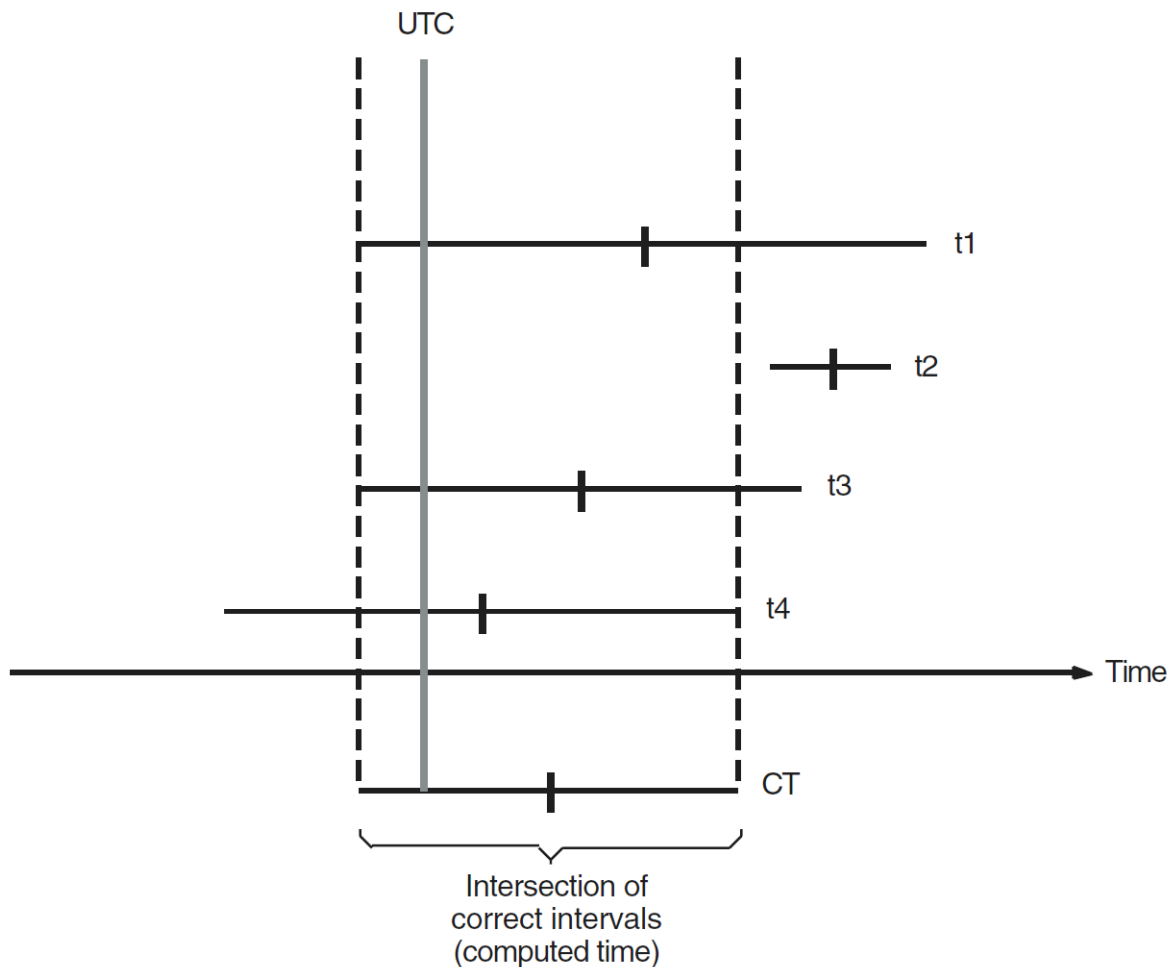
The `error tolerance` characteristic determines how DECdts reacts if the system's clock is faulty. A faulty clock is a rare condition, but some causes of faulty clocks include the following:

- Defects in the clock hardware, including clock drift greater than the manufacturer's specifications
- Malfunctioning time-providers

- Hardware clock ticks lost by the operating system
- The system memory containing the clock value is corrupted
- The presence of an application running for an extended period of time at a high IPL (thereby preventing clock updates)

During the synchronization process, DECdts detects that a system's clock is faulty if the clock value and its inaccuracy fail to intersect with those of the servers used for synchronization. This process is shown in Figure 2.3, "Nonintersecting Time Interval", where value t_2 is faulty.

Figure 2.3. Nonintersecting Time Interval



ZK-3855A-GE

If DECdts detects a non intersecting (potentially faulty) time interval during synchronization, the divergence of the non intersecting time and the system's `error tolerance` setting determine how DECdts reacts. When the non intersecting time is detected, DECdts performs one of the following operations:

- If the non intersecting interval supplied by the clock is within the bounds of the error tolerance, DECdts increases the inaccuracy of the value supplied by the clock and adjusts the clock gradually.
- If the non intersecting time interval supplied by the clock is outside the bounds of the error tolerance, DECdts immediately sets the clock to the new computed time.

Before you change the default `error tolerance` setting, determine the requirements of the applications that use the system time. Some distributed applications, such as DECdns server software, require that systems have no more than 5 minutes of inaccuracy. Larger error tolerances can prevent such applications from properly sequencing DECdns namespace entries. For these applications, set the `error tolerance` value to 5 minutes or less.

Some applications may require DECdts to adjust the system clock gradually and monotonically (forward). You can increase the `error tolerance` setting for these applications to ensure that the clock is abruptly set only in the event of a catastrophic error. If you could set the `error tolerance` characteristic to infinity, you could guarantee that the clock is never set abruptly. This setting is not available, but you can enter any setting less than 1334399-21:21:00.685 (approximately 3650 years).

The default value for `error tolerance` is 5 minutes (00-00:05:00.0000). The following example shows how to change the `error tolerance` setting to 3 minutes.

```
ncl> set node splsys dtss error tolerance 00-00:03:00.0000
```

2.4.1.7. The LAN Timeout, WAN Timeout, and Query Attempts Characteristics

When a system queries a server, it waits for a response for the period specified by the `lan timeout` or `wan timeout` characteristic. The `lan timeout` setting applies when the system is attempting to contact a local server; the `wan timeout` setting applies when the system is attempting to contact a global server.

The `query attempts` characteristic determines how many times DECdts resets the timeout timer before the system quits trying to contact a local server. Once the timeout setting has elapsed the number of times determined by the `query attempts` characteristic, the system quits querying the local server and does not post an event. Global servers, however, are only queried once. If a global server does not respond when queried, the DECdts process generates a Server Not Responding event report and removes the server from the system's list of global servers. If a response from the local or global server is required in order to meet the `servers required` setting, DECdts generates a Too Few Servers Detected event report, and the system does not synchronize.

The default setting for the `query attempts` characteristic is 3. The following example shows how to set the `query attempts` value.

```
ncl> set node splsys dtss query attempts 4
```

The default setting for the `lan timeout` characteristic is 2 seconds, and the default setting for the `wan timeout` characteristic is 15 seconds. The WAN setting is larger to account for the communications delay associated with WAN links. It is unlikely that you will have to change the `lan timeout` setting. The WAN setting, however, may need to be changed because of variations in WAN topologies and transmission quality. In the following example, the `wan timeout` setting is changed to 20 seconds:

```
ncl> set node splsys dtss wan timeout 20.00
```

If you continually receive Server Not Responding event reports for a global server, increase the `wan timeout` setting. If you increase the setting and the event reports continue, there may be a problem with the communications link to the server, the server may be offline, or the server may not be enabled. The following example shows how to display the status of a global server (enter the node name displayed in the event message):

```
ncl> show node farsys dtss all status
```

2.4.2. Server Management

Some commands and characteristics are unique to server systems. Managing a DECdts server involves some tasks in addition to normal system management. These tasks include the following:

- Setting a server's epoch
- Assigning the courier role to a server
- Advertising a server to DECdns or configuring the site-specific template to add the server to the list of DECdts global servers
- Setting and using the `check interval` and `advertisement interval` characteristics

Most of the tasks in the preceding list are performed at the initial system startup. Setting up a system as a global server and assigning the courier role are described in *Section 2.2.4, "Designating Global Servers and Couriers"*. This section describes the remaining tasks.

2.4.2.1. Matching the Epoch

At startup, a server's epoch number must match those of the other servers with which it synchronizes. When synchronizing, a server disregards clock values from servers whose epoch numbers do not match its own.

When they are enabled initially, all DECdts servers have the epoch number 0 (zero); you need not change the epoch numbers at initial installation. Later, if you add a server to an existing network, or change a clerk to a server, ensure that the new server and the preexisting servers have matching epoch numbers. Enter the following command to find out the epoch number of a preexisting server; you must enter the server's system ID.

```
ncl> show node splsys dtss epoch number
```

If the epoch of a server you have just created matches those of the other servers, the new server can synchronize immediately. If the epochs do not match, however, and you do not change the epoch of the new server, the new server ignores the pre-existing servers. The following example shows how to change a server's epoch number after you have enabled the server:

```
ncl> change node splsys dtss epoch 0
```

Once you know that a server is starting up with the proper epoch number, do not change the epoch, unless serious system or network problems corrupt all of the server clock values. In the unlikely event that the majority of the server clocks become faulty, use the `change` command and the `epoch number` characteristic to isolate problem servers, so you can perform troubleshooting and maintenance without affecting the rest of the DECdts application. See *Chapter 3, "Problem Solving"*, for further information on setting epoch numbers to isolate, test, and troubleshoot system clocks.

2.4.2.2. The Advertisement Interval and Check Interval Characteristics

Use the `advertisement interval` characteristic to specify how often you want the servers on a LAN to broadcast their availability by advertisement messages. The default setting of 90 minutes should be adequate. You can update the lists of servers maintained in all the systems more frequently by lowering the setting, but lower settings contribute to greater network overhead. The following example shows how to set the `advertisement interval` value:

```
ncl> set node splsys dtss advertisement interval 00-00:30:00.00
```

Enter the `check interval` characteristic only for servers that are connected to time-providers. The `check interval` function allows DECdts to periodically check all the servers on a LAN to make sure they are remaining synchronized with the time-provider. When the amount of time specified by the `check interval` setting has elapsed, the server with the time-provider (the TP server) performs the following procedure:

1. The TP server requests time values from all the other servers on the LAN.
2. The TP server starts the synchronization process.
3. The TP server identifies the server time intervals that do not intersect with its own.
4. The TP server issues event messages for each nonintersecting server it detects.

In the preceding sequence, note that the TP server does not actually modify the system clock after it starts the synchronization process. The TP server runs the process only to detect nonintersecting (potentially faulty) servers. The DECdts software assumes that the time value at the TP server is the most accurate available, so the TP server does not use the values it collects from other servers to change its clock. The TP server instead functions as a reference timekeeper for the other servers.

You can set the check interval to a lower value for a more rapid notification of nonintersecting servers, but lower settings contribute to greater network overhead. The following example shows how to set the `check interval` value:

```
ncl> set node splsys dtss check interval 00-00:30:00.00
```

2.4.3. Modifying the System Time

There are four ways you can modify a system's time using DECdts commands.

1. Update the time monotonically
2. Change the time non monotonically
3. Force system synchronization
4. Set the system clock to local time

This section describes reasons for changing the system time, and shows examples of the commands you can use to modify the time and change the system clock.

2.4.3.1. Updating the Time Monotonically

If your network does not use time-providers and the network systems have been running for some time, you may want to update the time on several servers to match UTC or another external reference. When time-providers are absent from your network, the systems remain closely synchronized, but their clocks may drift away from accepted time standards such as UTC.

Use the `update` command to modify the time on a server system to make it more accurate. The DECdts synchronization process ensures that the new time you supply with the `update` command is propagated to the other network systems. To update the server clock to a new time, the interval determined by the new time and inaccuracy must be within the bounds of the current interval.

To use the `update` command effectively, you must have temporary access to a trusted time reference. Such references include the time signals that many standards organizations disseminate by radio or

telephone. You can also use a clock that you have recently verified as accurate. See *Appendix B, "Time-Providers and Time Services"* for suppliers of UTC time.

Because the `update` command is a manually entered command used to modify an absolute time, it is not useful for small inaccuracy settings. The minimum reliable inaccuracy you can achieve with the `update` command is approximately 1 second. Human error and processing delays combine to make lower settings unreliable. For example, you type the command and new time and then begin monitoring the reference. When you perceive that the reference has reached the desired time, you press the Return key to initiate the command. Your perception of the reference mark and your pressing of the Return key do not coincide exactly. Furthermore, once the command is initiated, the system load increases and DECdts takes time to interpret and execute the command.

The following example shows how to update the time on a system (and eventually propagate the adjustment throughout the network):

```
ncl> update node splsys dtss time 1990-10-07-09:30:15.0I01.0
```

If your systems require synchronization closer than 1 second to a standard such as UTC, consider purchasing one of the time-providers listed in *Appendix B, "Time-Providers and Time Services"*. All of the time-providers described compensate for transmission and processing delays, and can provide time references that are precise to the millisecond level.

2.4.3.2. Changing System Time Nonmonotonically

Use the `change` command when you want to immediately set the time for a server system. The command immediately changes the system clock setting to the specified time, rather than gradually adjusting the clock as with the `update` command. Use caution when changing the time on a server; the command can set the clock backwards and can affect other distributed applications. Use the `change` command at system startup or when the system clock is faulty and you have identified and corrected the problem. The `epoch` argument is required for this command, and the epoch number you specify must be different from the current epoch number before the command will execute and change the time. If you change the epoch, then change the server's time to a setting that falls outside the time intervals of the previously known servers, and then change the epoch number back to the original setting, DECdts will declare the local system faulty at the next synchronization.

Because the `change` command is normally used to correct gross clock errors, it is likely that the time you specify for a given system will not intersect with those of the system's known servers if the system and servers have the same epoch number. You can prevent the systems whose times you are changing from being declared faulty by using the `time` argument to set the new time for a system and the `epoch` argument to isolate it from the other systems. You can then change the epoch for the other systems until all the systems once again share the same epoch. This process is useful in the extremely rare case when the majority of servers in the network have become faulty.

To use the `change` command effectively, you must have temporary access to a trusted time reference. Such references can include the time signals that many standards organizations disseminate by radio or telephone. You can also use a clock that you have recently verified as accurate. See *Appendix B, "Time-Providers and Time Services"* for a list of time reference sources.

Like the `update` command, the `change` command is a manually entered command used to modify an absolute time. It is not useful for small inaccuracy settings; the minimum reliable inaccuracy you can achieve with the `change` command is approximately 1 second.

The following example shows how to change both the time and epoch for a system. The epoch argument is required when setting the time.

```
ncl> change dtss epoch 1 time 1990-10-07-09:30:15.0000I01.0000
```

2.4.3.3. Forcing System Synchronization

Once you have created and enabled DECdts on all the systems in your network, they will synchronize without any further intervention. There are situations, however, when you want to force a system to initiate a synchronization rather than waiting for the amount of time specified by the `synchronization hold down` and `maximum inaccuracy` characteristics. For example, you can synchronize a system with a TP server that you have just added to the network. If you enter the `synchronize` command, the narrow time interval contributed by the time-provider is quickly propagated throughout the network.

The following example shows how to force a system to synchronize:

```
ncl> synchronize node splsys dtss set clock true
```

You can adjust the system clock gradually or set it immediately. By entering the `set clock` argument and the value `true`, the system clock is set; use this argument if you want faster results and are not running applications that depend on monotonic time. If the system clock has a relatively small inaccuracy, and it is not important for the clock to be set immediately, enter the `set clock` argument and the value `false`; however, the clock adjustment takes longer to complete.

2.4.3.4. Setting the System Clock to Local Time

DECdts displays the local time by default. DECdts uses UTC as the standard by which it monitors and adjusts time. All times that are displayed by DECdts reflect the time zone rule that is selected at system installation. Normally, the time displayed is a **local time** that is offset from UTC by a specified number of hours and minutes. The offset used to determine the local time is called the time differential factor (TDF). The time zone rules for each geographic location also dictate seasonal changes to the TDF and the resulting local time display. For further details on changing the TDF, see *DECnet-Plus for OpenVMS Installation and Basic Configuration Manual*.

Some time zones specify periodic seasonal adjustments of 1 hour or less. Depending on the time zone you select at the initial operating system or DECnet-Plus configuration, your system may require periodic adjustment of the local TDF.

If the time zone rule for your system specifies a periodic TDF change, you can use either of the following methods to adjust the local TDF:

- Allow the DECdts entity to change the local TDF (and displayed local time) automatically by leaving the `automatic tdf change` characteristic set at its default setting of `true`.
- Adjust the local TDF manually by entering the `set dtss auto tdf change false` command and then entering the `set dtss auto tdf change true` command at the appropriate time of year. After the local time has changed, you can disable automatic TDF changes again.

If you are running time-sensitive applications (applications that read the system clock and require monotonically increased time) on a DECnet-Plus for OpenVMS system, you should use the preceding manual procedure with additional steps as follows:

1. Enter the NCL command `set dtss automatic tdf change false` after you enable the DECdts software. You may want to add this command to the DECnet-Plus startup script as described in Section 2.1.3, "The DECnet Startup Script".

2. When a seasonal time change is required, disable the time-sensitive application.
3. Enter the NCL command `set dtss automatic tdf change true`.
4. Monitor the local time with the NCL command `show dtss current time` to ensure that it has changed. If the seasonal time change for the system's locale has already passed, the local time is changed immediately.
5. Restart the time-sensitive application.

Chapter 3. Problem Solving

This chapter describes the problems most commonly encountered when implementing DECdts and recommends actions to solve the problems.

3.1. A Clerk System's Clock is Incorrect

If you have just installed the clerk, the time may be incorrect unless you entered the DTSS management command `enable dtss set clock true`. If you do not set the clock when enabling the clerk, the time is updated gradually and may take a long time to complete if the clock is grossly inaccurate.

If the clerk is not newly installed, complete the following steps:

1. Use NCL to check the values of the `synchronization hold down` and `maximum inaccuracy` attributes:

```
ncl> show dtss synchronization hold down
ncl> show dtss maxi inac
```

The default values for these attributes are 10 minutes and 0.2 second, respectively. If the current settings for either of these attributes are higher, go to step 2. Otherwise, go to step 3.

2. Set the `synchronization hold down` and `maximum inaccuracy` attributes to their default settings or less.

```
ncl> set dtss sync hold down
ncl> set dtss maxi inac
```

3. Enter the following command to force a synchronization and set the clock (note that setting the clock can interfere with DECdns and other distributed applications that depend on monotonically adjusted time):

```
ncl> synch dtss set clock true
```

After you enter the `synchronize` command, recheck the current time. If the time is still incorrect, go to the next step.

4. Enable all DTSS events as follows:

```
ncl> pass event disp outb stream * glob filt ((dtss), all)
```

5. After you have enabled all DTSS events, enter the following NCL command:

```
ncl> synchronize dtss set clock true
```

If the `Synchronization Completed` event is displayed and the time is still incorrect, one or more of the servers have incorrect times; see *Section 3.2, "A Server System's Clock is Incorrect"*. If a different event is displayed, see the following list of events and actions.

- **Insufficient Resources Detected:** Allocate more virtual memory to the system.
- **System Errors Detected:** Follow the recommended user action described in the supplemental error message.
- **Too Few Servers Detected:** Lower the `servers required` setting or add servers to the network.

- Server Not Responding: See *Section 3.3, "Local Servers are Unavailable or Unresponsive"* and *Section 3.4, "Global Servers are Unavailable or Unresponsive"*.

3.2. A Server System's Clock is Incorrect

If you have just installed the server, the time may be incorrect unless you entered the DTSS management command `enable dtss set clock true`. If you do not set the clock when enabling the server, the time is updated gradually and may take a long time to complete if the clock is grossly inaccurate.

If the server is not newly installed, see the problem descriptions and solutions in the following sections.

3.2.1. Large Inaccuracy On One Server

If the inaccuracy value is infinite or very large on the subject server, but relatively small on other servers, follow these steps:

1. Enter the following commands to determine if the server has synchronized recently:

```
ncl> show dtss current time
ncl> show dtss last sync
```

If there is a large difference between the two times, go to Step 4. If the server has synchronized recently, go to the next step.

2. Enter the following NCL commands to check the values of the `synchronization hold down` and `maximum inaccuracy` attributes:

```
ncl> show dtss synchronization hold down
ncl> show dtss maxi inac
```

If the `synchronization hold down` setting is higher than 5 minutes, and/or the `maximum inaccuracy` setting is higher than 0.2 second, go to step 3. Otherwise, go to step 4.

3. Set the `synchronization hold down` and `maximum inaccuracy` attributes to their default settings or less.

```
ncl> set dtss synchronization hold down
ncl> set dtss maxi inac
```

4. Enter the following command to force a synchronization and set the clock (note that setting the clock can interfere with DECdns and other distributed applications that depend on monotonically adjusted time):

```
ncl> synch dtss set clock true
```

After you enter the `synchronize` command, recheck the current time. If the time is still incorrect, go to the next step.

5. Enable all DTSS events as follows:

```
ncl> pass event disp outb stream * glob filt ((dtss), all)
```

6. After you have enabled all DTSS events, enter the following NCL command:

```
ncl> synchronize dtss set clock true
```

7. If the Synchronization Completed event is displayed, the time should be correct. If a different event is displayed, see the following list of events and actions.

- **Different Epoch Detected:** Enter the NCL command `show dtss epoch number`; note the epoch number of the correct servers and change the epoch number of the differing server to match. At the same time, you can specify a correct time with the `change` command. For example:

```
ncl> change dtss epoch number 1 time-  
_ncl> 1991-03-21-16:07:46.479-05:00I01.00
```

- **Insufficient Resources Detected:** Allocate more virtual memory to the system.
- **System Errors Detected:** Follow the recommended user action described in the supplemental error message.
- **Too Few Servers Detected:** Lower the `servers required` setting or add servers to the network.
- **Server Times Not Intersecting:** Correct the times of the incorrect servers by entering the `change` command for each server as shown in the previous example.

3.2.2. Large Inaccuracy On All Servers

If the inaccuracy value is infinite or very large on all servers, and you want to adjust the clocks monotonically, enter the `update` command for any of the servers. See the following example:

```
ncl> update dtss time 2019-03-21-16:07:45.000-05:00I02.00
```

To verify that the clock adjustment is occurring, enter several `show dtss current time` commands (pause between each command). The inaccuracy value should decrease and eventually stabilize on all servers.

If the inaccuracy value is infinite or very large on all servers, and you want to set the clocks non monotonically, enter the `change` command for one of the servers, specifying an epoch number (that is different from those of the other servers) and the correct time. See the following example:

```
ncl> change dtss epoch 1 time 2019-03-21-16:07:45.000-05:00I0.50
```

Enter the `show dtss current time` command to make sure that the system has the correct time and inaccuracy, and then enter the `change` command for each of the other systems in the network, specifying a new epoch number for each. See the following example:

```
ncl> change node jhmsys dtss epoch 1
```

As each server switches to the new epoch, its clock is synchronized with the servers that already have that epoch.

If you have used the `update` or `change` command to correct the clocks on all servers, and their times are not corrected, follow these steps:

1. Enable all DTSS events on each node as follows:

```
ncl> pass event disp outb stream * glob filt ((dtss), all)
```

2. After you have enabled all DTSS events, enter the following NCL command for each server:

```
ncl> synchronize dtss set clock true
```

3. If the Synchronization Completed event is displayed, the time should be correct. If a different event is displayed, see the following list of events and actions.

- Different Epoch Detected: Enter the NCL command `show dtss epoch number`; note the epoch number of the correct servers and change the epoch number of the differing server to match. At the same time, you can specify a correct time with the `change` command. For example:

```
ncl> change dtss epoch number 1 time-  
_ncl> 1991-03-21-16:07:46.479-05:00I01.00
```

- Insufficient Resources Detected: Allocate more virtual memory to the system.
- System Errors Detected: Follow the recommended user action described in the supplemental error message.
- Too Few Servers Detected: Lower the `servers required` setting or add servers to the network.

3.3. Local Servers are Unavailable or Unresponsive

If the NCL command `show dtss decnet local server * all` does not display servers, or if some servers were not used for the last synchronization, there are several possible reasons.

- Servers not installed/created: Verify that the other servers have been installed and the `create dtss type server` command has been entered for each server; if not, install and create the servers.
- Servers not enabled: Verify that all servers are enabled by entering the `show dtss state` command for each server. The `state` value should be `on`. If the `state` value is `off`, enter the `enable dtss` command for each server.
- Server epochs differ: If a server cannot find other servers, it may have a different epoch from the others. In this case, if a server attempts to synchronize with a server in another epoch, the event Different Epochs Detected is generated at the initiating server and the synchronization fails. Use the `show dtss epoch number` command for several servers to learn which epoch they are using, and then enter the `change` command with a new epoch number for the nonmatching server.
- Servers time out: Check the value of the `lan timeout` characteristic by entering the following command:

```
ncl> show dtss lan timeout
```

If the value is less than 2 seconds, or you suspect that the LAN may have high traffic volume, increase the value by entering the command `set dtss lan timeout n`, where `n` is a higher value than previously shown. It should not be necessary to set the value higher than 15 seconds.

If increasing the `lan timeout` setting does not cause the servers to respond, enter the following commands (disabling and enabling the entity forces it to repopulate the server list):

```
ncl> disable dtss
```

```
ncl> enable dtss
```

- Servers isolated from the LAN: If your network has an extended LAN topology, and bridges connect the LAN segments, the bridges may be preventing communications between servers. Program all bridges to pass the DECdts multicast messages unfiltered. The DECdts multicast address (ID) is 09-00-2B-02-01-02.

3.4. Global Servers are Unavailable or Unresponsive

If the NCL command `show dtss decnet global server * all` does not display servers, or if some servers were not used at the last synchronization, there are several possible reasons. See *Section 3.3, "Local Servers are Unavailable or Unresponsive"* for a preliminary list of reasons and suggested actions; see the following list for WAN-related reasons and actions.

- Servers do not respond in time: Check the value of the `wan timeout` characteristic by entering the `show dtss wan timeout` command. If the value is less than 15 seconds, or you suspect that the WAN may have high traffic volume or routing delays, increase the value by entering the `set dtss wan timeout n` command, where `n` is a higher value than previously shown.
- Global directory not set or empty: To see if a global directory has been set, enter the `show dtss global directory` command. If a full name is not displayed, specify a global directory by entering the `set dtss global directory name` command, where `name` is the full name of the global directory. To show the objects in the global directory, use the `DECdns directory` command or the `DECdns browser` (for further details, see the *VSI DECnet-Plus for OpenVMS DECdns Management Guide*). If the directory is empty, add global server names to the directory by entering the NCL command `advertise dtss server name name`, where `name` is the simple name of a global server system.
- DECdns clerk not functioning: Enter the NCL command `show dns clerk status` and verify that the state is `on`. If the state is not `on`, see the *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for recommended actions.
- Global servers do not respond: Enter the NCL command `show dtss servers not responding`, wait for the amount of time specified by the `synchronization hold down` attribute, and enter the command again. If the counter value has increased, the WAN link may not be working or the timeout value may be too low. Verify that the WAN link is functional.

3.5. The Advertise Command Fails

There are several reasons why the `advertise` command can fail. This section contains possible causes and suggested actions.

3.5.1. Global Directory Not Configured

If the `advertise` command fails, an incorrect or missing global server directory is the most common cause. To check the accuracy of the global server directory, follow these steps:

1. Display the global server directory by entering the following command:

```
ncl> show dtss global directory
```

2. Verify the directory name by entering a `DECdns` command as in the following example:

```
dns> show directory WAK:.DTSS_GlobalTimeServers
```

If the DECdns command does not return the directory name and attributes, the name is incorrect. See the next step.

3. Verify the global directory name by checking the directory name from other systems or by checking the name recorded on the planning worksheets. Set the correct global server directory by entering an NCL command as in the following example:

```
ncl> set dtss global directory MAK:.DTSS_GlobalTimeServers
```

3.5.2. Access Violation Messages

When you enter the `advertise` command, an access violation (no privilege) or unknown entry message may be returned. If an access violation message is displayed, follow these steps:

1. Enter the following NCL command to display the global directory for the node you are using:

```
ncl> show dtss global directory
```

2. Exit NCL, access the DECdns control program, and display the access rights for the global server directory, as in the following example:

```
dns> show directory WAK:.DTSS_GlobalTimeServers
```

3. When the directory attributes are displayed, ensure that the DNS\$ACS set with the suffix `system` has read, write, delete, test, and control privileges. If the root or system accounts do not have these privileges, add them as in the following examples:

```
dns> add directory WAK:.DTSS_GlobalTimeServers-  
_> access .mynode.system for r,w,d,t,c
```

4. Reaccess NCL and enter the `advertise` command again.

If you follow the preceding steps and the `advertise` command still returns an access violation message, the problem may be due to session control access to the DECdns directory. Follow these steps to change the access rights:

1. Access the DECdns control program and display the access rights for the global server directory, as in the following example:

```
dns> show directory WAK:.DTSS_GlobalTimeServers
```

2. When the directory attributes are displayed, ensure that the DNS\$ACS set with the suffix "DNA \$SessCtrl" has read and write privileges. If the set does not have these privileges, add them as in the following example:

```
DNS> add directory WAK:.DTSS_GlobalTimeServers -  
_DNS> access wak:nodename.DNA$SessCtrl for r,w,t,c  
DNS> add directory WAK:.DTSS_GlobalTimeServers -  
_DNS> access wak:nodename.system for r,w,t,c
```

In these examples, `nodename` is the full name of your local system.

3. Complete the `advertise` command by entering a session control command as in the following example:

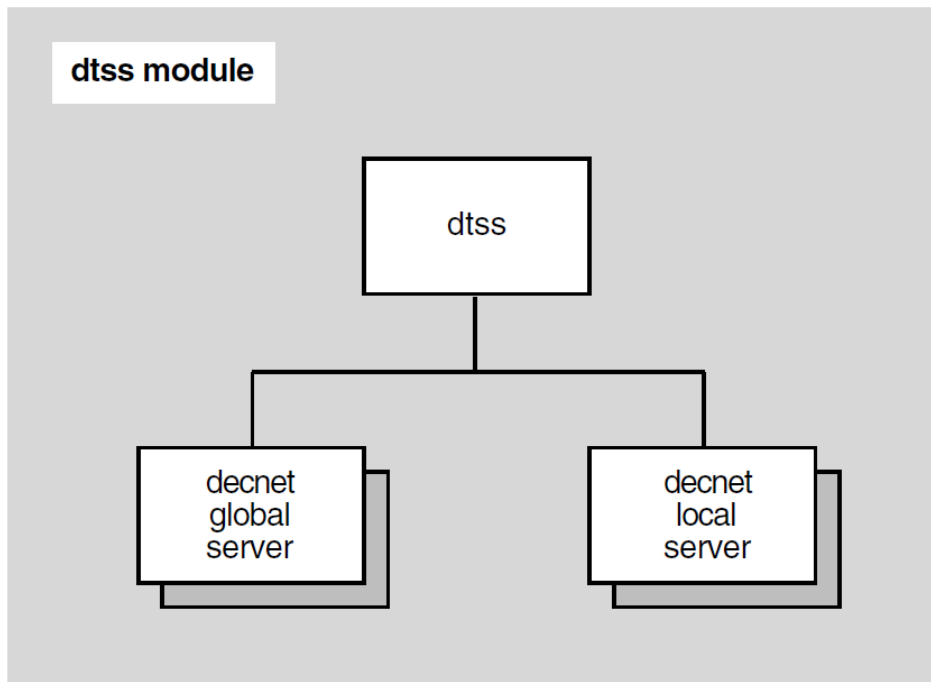
```
ncl> update session control tower maintenance-  
_ncl> WAK:.DTSS_GlobalTimeServers
```


Chapter 4. DECdts Command Dictionary

The Network Architecture (NA) contains many modules that each relate to a functional area or product. All DECdts management functions are contained in the DTSS module. The DTSS module has several entities that enable you to synchronize, adjust, and maintain the system clocks in a distributed network. This chapter describes accessing the command interface and the command format, and also describes the commands for the entities that constitute the DTSS module. For a detailed explanation of system clock synchronization and management, see *Chapter 2, "Managing DECdts"*. For further information on the Network Architecture and its Network Command Language (NCL), see *VSI DECnet-Plus for OpenVMS Network Control Language Reference Guide*.

Figure 4.1, "Hierarchy of DTSS Module Entities" shows the hierarchical relationship of the entities that constitute the DTSS module.

Figure 4.1. Hierarchy of DTSS Module Entities



ZK-2629A-GE

4.1. The DTSS Entities

Each entity in the DTSS module is listed as follows, along with the commands used to manage it. Note that the main entity in the DTSS module is named `dtss`.

```
dtss advertise
  change
  create
  delete
  disable
```

```

enable
set
show
synchronize
update
dtss decnet global servers
show
dtss decnet local servers
show

```

The DTSS entities have the following functions:

- The DTSS entity provides access to most of the management functions for the DTSS module and the DECdts product, including creating and deleting the software, enabling and disabling the software, setting attributes, adjusting the clock, and forcing synchronizations.
- The `dtss decnet global server` entity provides information about a node's synchronization with one or more servers in the global set.
- The `dtss decnet local server` entity provides information about a node's synchronization with one or more servers in the local set.

The commands for each entity are described in reference pages that appear in alphabetical order in this chapter.

4.2. Command Format

All of the entities in the DTSS module use the same NCL format for commands. To enter a command, enter the name of the command and the name of the entity to which the command applies. If you are trying to manage the entity at a remote system, you must also enter `node` followed by the node ID. Some commands have required arguments, and some commands have optional arguments, attributes, or prepositional phrases. Arguments modify the way the command is executed; attributes further define a general command such as `set` or `show`; prepositional phrases filter or redirect the command output. A value is associated with each argument and set table attribute, and most prepositional phrases. A command example is shown in *Figure 4.2, "Command Syntax"*.

Figure 4.2. Command Syntax

```

ncl> set node my_node dtss servers required 3
      ^   ^   ^   ^   ^   ^   ^   ^   ^
      |   |   |   |   |   |   |   |
      Command Node Node-ID Entity [Attribute] Value
                                   [Argument]

```

ZK-2763A-GE

4.2.1. Attribute Groups

Each entity in the DTSS module has several types of attributes—pieces or sets of data associated with the entity. Attributes describe the operational properties of an entity and specify other information that regulates or monitors the entity's behavior. The DTSS module attributes are categorized as follows:

Characteristics	Display or affect the entity's operation
-----------------	--

Counters	Record the number of occurrences of an event since the entity was last enabled
Identifiers	Distinguish the entity from other entities
Status	Describe the current operational state of the entity

Each entity has a `show` command that you can use to display attribute values; you can specify attribute groups or individual attributes with these commands. The `dtss` entity also has a `set` command that you can use to modify the values of some attributes.

4.2.2. Prepositional Phrases

You can use prepositional phrases to affect the destination or content of command output. The phrases are most useful with the `show` command. If you are managing a remote system or managing the local system from a nonprivileged account, you can use the `by` prepositional phrase to specify access control information with a command. See *Section 4.2.3, "NCL Access Control"* for further information on how to use the `by` phrase.

You can combine several prepositional phrases in one command; precede each phrase with a comma and a space. The following table summarizes the `to` and `with` prepositional phrases and their uses:

<code>, to file=<i>filename</i></code>	Redirects the output to <i>filename</i> . If the file exists, a new version is created. If the file does not exist, it is created.
<code>, to extend file=<i>filename</i></code>	Appends the output to an existing file <i>filename</i> . If the file does not exist, it is created.
<code>, to terminal</code>	Directs the output to the terminal. This is the default prepositional phrase.
<code>, with <i>attribute</i>[<i>relop</i>] <i>value</i></code>	Limits a directive only to those entities whose attributes have certain values. If you do not specify a <i>relop</i> (relational operator), the default is an equal sign. Other valid relational operators are greater than (<code>></code>), less than (<code><</code>), greater than or equal to (<code>>=</code>), less than or equal to (<code><=</code>), and not equal (<code><></code>). If you specify multiple <code>with</code> phrases in a command, the entity must meet all the conditions you specify before the command executes. This phrase works only with commands that can operate on multiple attributes (such as <code>show</code> commands).

4.2.3. NCL Access Control

NCL access control allows you to manage entities on remote systems or through various accounts on the local system (node 0). To supply access control information in a command, use the `by` prepositional phrase or specify an access control string in the node id. This section describes these methods.

Use the `by` prepositional phrase at the end of a command to execute the command through an account or proxy account other than the one you are currently using. The account and user must have the required permissions to execute the command. The following example shows the `by` phrase format:

```
by user=username, password=password, account=account, proxy=[true/false]
```

The following example shows a command that uses access control:

```
ncl> set node 0 dtss serv req 4, by user=kosta, password=abc4th5zbn
```

You can also specify access control information as part of a node id you enter with a command. The combined node-id/access control string contains a node name, a user/account name, and an account password. In the following example, `systempasswd` is the password of the system account on node `farnod`.

```
ncl> set node farnod"system systempasswd" dtss servers required 3
```

For more information about access control in NCL, see the *VSI DECnet-Plus for OpenVMS Network Control Language Reference Guide*.

4.2.4. Wildcards

You can use wildcard characters in the last simple name that you specify in show commands. Wildcards are especially useful with the `show dtss decnet local server` and `show dtss decnet global server` commands for displaying all known servers or a subset of the known servers. The following table describes the valid wildcard characters you can use.

Table 4.1. Wildcard Characters

Symbol	Meaning
*	Matches zero (0) or more characters in the name that you specify in a simple name
?	Matches exactly one character in the name that you specify
...	Searches any part of the directory hierarchy at or below this level for a match

See the show command reference pages for examples of wildcard use.

4.3. Editing the Commands

You can cancel commands, edit command lines, continue a command beyond one line, or recall commands within NCL.

Canceling a Command

Press `Ctrl/C` to cancel command processing during command entry or while the command is being processed; you are returned to the system prompt.

Continuing a Command Line

To continue a long command line onto the next line, type a hyphen at the end of the first line:

```
ncl> show node smrt_node dtss decnet local server * all status, -  
_ncl> with used in last synchronization=true
```

Recalling a Command

You can recall previously typed commands to avoid retyping them. The recall buffer holds up to 20 commands entered during the current NCL session. Once a command is redisplayed, you can reexecute or edit it.

To display the commands stored in the recall buffer, press `Ctrl/B` or the up arrow and down arrow keys. Pressing `Ctrl/B` recalls the previous command. The up and down arrow keys allow you move up and down through the stack of commands in the buffer. When the desired command is displayed, press the Return key to enter it.

4.4. Accessing and Exiting the Management Interface

You can use either of two methods to start the NCL command interface and access the NCL prompt. The first method follows:

```
$ run sys$system:ncl
ncl>
```

You can also access the NCL command interface using a second method:

```
$ mcr ncl
ncl>
```

To exit the NCL command interface and return to the system prompt, enter the `exit` command as follows:

```
ncl> exit
```

Alternatively, you can type `Ctrl/Z` to exit the interface.

4.5. Commands

This section describes the commands for managing the DTSS module entities.

advertise dtss

`advertise dtss` — Modifies the DECdns directory of DECdts global servers and adds the DECdts server to the global set.

Format

advertise [**node** *node-id*] **dtss** *argument*

Arguments

server name *simple-name*

Specifies the name of the server to be added to the DECdns list of global time servers.

Description

The `advertise` command, including the `server name` argument, causes the DECdts server entity to register the name of the server with DECdns, which adds the name to the directory of global time servers. To delete a global server name from the DECdns directory `.DTSS_GlobalTimeServers` and remove it from the global set, use the following DECdns command, substituting the full name of the node for `object-name`:

```
dnscp> delete object object-name
```

See *VSI DECnet-Plus for OpenVMS DECdns Management Guide* for information on how to access and use the DECdns control program.

Note

This command is valid only for servers.

If your network uses the Local namespace instead of DECdns, the advertise command fails and the error `DNS_NOCOMMUNICATION` is displayed. See the site-specific configuration `sys$common:[sysmgr]dtss$config_template.dat` for information on configuring global servers using the Local namespace.

Exceptions

System Error

The command failed because of an operating system error.

Wrong State

The entity is not in the on state.

Examples

The following example shows how to advertise a server.

```
ncl> advertise dtss server name yamsca
```

change dtss

`change dtss` — Alters the epoch number and time on the local node.

Format

change [**node** *node-id*] **dtss** *epoch integer* [, *time absolute-time*]

Arguments

epoch integer

Specifies the new epoch number; the new epoch number must be different from the current epoch number and in the range 0–255. This argument is required.

time absolute-time

Specifies a new clock setting for the new epoch. If the argument is not supplied, the current clock time is used for the new epoch. This argument is optional.

Description

Changes the epoch and (optionally) sets the time of the DECdts server on which it is entered. Use this command to isolate a server from the rest of the servers in the network and to change (set) the time.

Note

This command is valid only for servers.

Exceptions

Same Epoch

The new epoch must differ from the current epoch.

System Error

The command failed because of an operating system error.

Wrong State

The entity is not in the on state.

Examples

The following examples show uses of the `change` command.

1. `ncl> change dtss epoch 1`

This example shows how to change the epoch number.

2. `ncl> change dtss epoch 1, time 2019-11-30-10:58:00.000-05:00I02.000`

This example shows how to change the epoch number and time.

create dtss

`create dtss` — Creates the DECdts clerk or server process on the specified node.

Format

create [**node** *node-id*] **dtss** argument

Arguments

type *type*

Specifies the type of DECdts process to be created on the specified node. This argument is required.

<code>clerk</code>	The DECdts process is created as a clerk. The default setting is <code>clerk</code> .
<code>server</code>	The DECdts process is created as a server.

Description

This command creates a DECdts server or DECdts clerk process on the specified system.

Note

You cannot activate a DECdts process by interactively entering the `create` command on DECnet-Plus for OpenVMS systems. The entity type is determined by the NCL startup file that is created during the DECnet-Plus software installation (as described in *Section 2.1.3, "The DECnet Startup Script"*). To activate and create the clerk or server process without running the entire `net$startup` process, enter the following command at the system prompt:

```
$ @sys$startup:net$startup "network" "dtss" "startup"
```

As an alternate method, you can directly execute the DECdts startup file `sys$startup:dtss$startup.com` to create the clerk or server process.

Exceptions

Already Exists

The command failed because the clerk or server is already created.

System Error

The command failed due to an operating system error.

Examples

The following example shows how to create a server:

```
ncl> create node sys4u2 dtss type server
```

delete dtss

`delete dtss` — Deletes the DECdts process.

Format

delete [*node node-id*] **dtss**

Description

This command removes the server or clerk process from the specified system.

Note

The `delete` command will not execute unless you have previously entered the `disable` command, which causes the status attribute `state` to be set to `off`.

Exceptions

Wrong State

The DTSS entity is not in the `off` state.

Examples

The following example shows how to enter the `delete` command:

```
ncl> delete node abnrml dtss
```

Related Information

`disable` command

disable dtss

disable dtss — Stops the DECdts process on the specified node.

Format

disable [**node** *node-id*] **dtss**

Description

This command turns off the DECdts process on the specified system. When the command is executed, the status attribute `state` is set to `off`.

Note

The `disable` command will not execute unless you have previously entered the `enable` command, which causes the status attribute `state` to be set to `on`.

Exceptions

System Error

The command failed due to an operating system error.

Wrong State

The `dtss` entity is already in the `off` state.

Examples

The following example shows how to enter the `disable` command:

```
ncl> disable node slsmgr dtss
```

Related Information

`enable` command

enable dtss

enable dtss — Starts the DECdts process on the specified node.

Format

enable [**node** *node-id*] **dtss** [*argument*]

Arguments

set clock *boolean*

Specifies whether the clock is abruptly set or gradually adjusted to the computed time at the end of the first synchronization. This argument is optional.

<code>false</code>	The clock is gradually adjusted. This is the default condition.
--------------------	---

true	The clock is abruptly set.
------	----------------------------

Description

This command activates the DECdts process entity on the specified system. When the command is executed, the status attribute `state` is set to `on`.

Note

The `enable` command will not execute unless you have previously entered the `create` command.

Exceptions

System Error

The command failed because of an operating system error.

Wrong State

The `dtss` entity is not in the `off` state.

Examples

The following examples show uses of the `enable` command.

1. `ncl> enable node mysyst dtss`

This example shows how to enable the entity and adjust the clock gradually to the computed time following the first synchronization.

2. `ncl> enable dtss set clock true`

This example shows how to enable the entity and immediately set the clock to the computed time following the first synchronization.

Related Information

`disable` command

set dtss

`set dtss` — Modifies characteristic attributes for the `dtss` entity.

Format

set [**node** *node-id*] **dtss** *characteristic* [...]

Arguments

characteristic value

Specifies the name and value of the characteristic to be modified.

Specify one or more of the following for `characteristic`.

```

advertisement interval [relative-time]
automatic tdf change [Boolean]
check interval [relative-time]
courier role [role]
error tolerance [relative-time]
global directory [full-name]
lan timeout [relative-time]
maximum inaccuracy [relative-time]
query attempts [integer]
servers required [integer]
synchronization hold down [relative-time]
wan timeout [relative-time]

```

Description

This section describes each of the values for the `characteristic` argument.

advertisement interval

Specifies the amount of time between a server's advertisement broadcasts.

Default: 0-01:30:00.000	Value: 0-00:00:30.000–99-99:99:99.999
--------------------------------	--

automatic tdf change

Specifies whether the entity performs an unattended change to the local time differential factor, according to the time zone rule selected at the initial configuration.

Default: true	Value: true/false
----------------------	--------------------------

check interval

Specifies the amount of time between checks for faulty servers. Applicable only for servers that have external time-providers.

Default: 0-01:30:00.000	Value: 0-00:00:30.000–99-99:99:99.999
--------------------------------	--

courier role

Specifies a server's interaction with the set of global servers.

Default: backup courier	Value: See following description.
--------------------------------	--

backup courier	The local server may become a courier if it is not using a time-provider and no couriers are available on the LAN.
courier	The local server attempts to synchronize with at least one global server.
noncourier	The local server does not routinely synchronize with the global set of servers.

error tolerance

Specifies the maximum separation allowed between the local clock and the computed time before synchronizations become immediate rather than gradual (monotonic).

Default: 0-00:10:00.000	Value: 0-00:00:00.500–99-99:99:99.999
--------------------------------	--

global directory

Specifies the full name of the DECdns directory that contains the list of DECdts servers that belong to the global set. The default example uses `ns` to denote the default namespace.

Default: <code>ns::DTSS_GlobalTimeServers</code>	Value: See previous Description.
---	---

lan timeout

Specifies the amount of time the node waits for a response to a synchronization request before sending another request or declaring that a server is unavailable. The node sends the number of requests determined by the `query attempts` characteristic before declaring that a nonresponding server is unavailable.

Default: 0-00:00:02.000	Value: 0-00:00:00.000–0-00:05:00.000
--------------------------------	---

maximum inaccuracy

Specifies the inaccuracy limit for the node. When the node exceeds the maximum inaccuracy setting, it attempts to synchronize.

Default: 0-00:00:00.200	Value: 0-00:00:00.000–99-99:99:99.999
--------------------------------	--

query attempts

Specifies the number of attempts that a node makes to contact a server before the node considers the server unavailable.

Default: 3	Value: 1–10
-------------------	--------------------

servers required

Specifies the minimum number of servers the node must successfully query for time values before it can synchronize. If more than the specified number of servers are available, clerks query up to three servers and servers query all other local servers.

Default: 3 (servers) 1 (clerks)	Value: 1–32
---	--------------------

synchronization hold down

Specifies the interval a node must wait to synchronize. Also specifies synchronization frequency when a node reaches the value specified by the `maximum inaccuracy` characteristic.

Default: 0-00:10:00.000 (clerks) 0-00:05:00.000 (servers)	Value: -00:00:30.000–01-00:00:00.000
---	---

wan timeout

Specifies the amount of time the node waits for a response to a WAN synchronization request before declaring that a global server is unavailable.

Default: 0-00:00:15.000	Value: 0-00:00:00.000–0-00:10:00.000
--------------------------------	---

Examples

The following examples show some uses of the `set` command.

1. `ncl> set dtss advertisement interval 00-01:20:00.000`

This example shows how to set the advertisement interval.

2. `ncl> set dtss global directory co:.mktg.mont.bac`

This example shows how to set the global directory name.

3. `ncl> set dtss servers required 4`

This example shows how to set the number of servers required before the entity can synchronize.

4. `ncl> set dtss courier role backup courier`

This example shows how to set the courier role for a server.

Related Information

`show` command

show dtss

`show dtss` — Displays current information about the `dtss` entity.

Format

show [**node** *node-id*] **dtss** *attribute-specifier* [...]

Arguments

attribute-specifier

Specifies the name of an attribute group or individual attribute to be displayed.

Description

The `show` command displays the names and values of the attributes or attribute groups named in *attribute-specifier*. You can use any combination of attribute specifiers in any order in a single command; use a comma to separate specifiers.

If you do not supply an attribute specifier, the command displays all characteristics and their values. The following are valid attribute specifiers:

- `all [characteristics]`
- `all characteristics`
- `all counters`

- `all status`
- `attribute-name`

As shown in this list, `attribute-name` is the name of a specific attribute from one of the listed groups. The following sections list names of attributes, categorized by group (characteristics, counters, and status).

Characteristics

acting courier role

Specifies whether a backup courier is temporarily functioning as a courier. If the role is `courier`, the node is attempting to contact at least one global server at every synchronization. If the role is `non courier`, then the node is not attempting to synchronize with global servers.

automatic tdf change

Default: <code>true</code>	Value: <code>true/false</code>
-----------------------------------	---------------------------------------

Specifies whether the entity performs an unattended change to the local time differential factor, according to the time zone rule selected at the initial configuration.

advertisement interval

Default: <code>0-01:30:00.000</code>	Value: <code>0-00:00:30.000–99-99:99:99.999</code>
---	---

Specifies the amount of time between a server's advertisement broadcasts.

check interval

Default: <code>0-01:30:00.00</code>	Value: <code>0-00:00:30.000–99-99:99:99.999</code>
--	---

Specifies the amount of time between checks for faulty servers. Applicable only to servers that have external time-providers.

clock adjustment rate

Specifies the rate at which the DECdts server or clerk entity adjusts the node's clock during a synchronization.

clock resolution

Specifies the amount of time, in nanoseconds, between system clock ticks. The value is determined by the operating system.

courier role

Default: <code>backup courier</code>	Value: See following description.
---	--

Specifies a server's interaction with the set of global servers.

<code>backup courier</code>	The local server may become a courier if none are available on the LAN.
<code>courier</code>	The local server attempts to synchronize with at least one global server.

<code>noncourier</code>	The local server does not routinely synchronize with the global set of servers.
-------------------------	---

NA Version

Specifies the Network Architecture (NA) version to which the DECdts software conforms.

DTSS Version

Specifies the DTSS module/DECdts software version installed on the node.

epoch number

Default: 0	Value: 0–255
-------------------	---------------------

Specifies the server's epoch number. The `change` command modifies this characteristic.

error tolerance

Default: 0-00:10:00.000	Value: 0-00:00:00.500–99-99:99:99.999
--------------------------------	--

Specifies the maximum separation allowed between the local clock and the computed time before synchronizations become immediate rather than gradual (monotonic).

global directory

Default: <code>ns::DTSS_GlobalTimeServers</code>	Value: See following Description
---	---

Specifies the full name of the DECdns directory that contains the list of DECdts servers that belong to the global set. The default example uses `ns` to denote the default namespace.

lan timeout

Default: 0-00:00:02.000	Value: 0-00:00:00.000–0-00:05:00.000
--------------------------------	---

Specifies the amount of time the node waits for a response to a synchronization request before sending another request or declaring that a server is unavailable. The node sends the number of requests determined by the `query attempts` characteristic before declaring that a nonresponding server is unavailable.

local time differential factor

Specifies the time differential factor (TDF) for the node, which is the difference between the node's displayed local time and UTC (Coordinated Universal Time).

maximum clock drift rate

Specifies the worst case drift rate of the node's clock, in nanoseconds per second, as determined by the manufacturer's specifications.

maximum inaccuracy

Default: 0-00:00:00.200	Value: 0-00:00:00.0–99 99:99:99.99
--------------------------------	---

Specifies the inaccuracy limit for the node. When the node exceeds the maximum inaccuracy setting, it attempts to synchronize.

next tdf change

Specifies the calendar date and time of the next scheduled adjustment of the local time. The value is determined by the time zone rule selected during operating system or DECnet configuration. The value of the characteristic `auto tdf change` must be `true` for the change to become effective.

query attempts

Default: 3	Value: 1–10
-------------------	--------------------

Specifies the number of attempts that a node makes to contact a server before the node considers the server unavailable.

servers required

Default: 3 (servers) 1 (clerks)	Value: 1–32
---	--------------------

Specifies the minimum number of servers the specified node must successfully query for time values before it can synchronize.

synchronization hold down

Default: 0-00:10:00.00 (clerks) 0-00:05.00.00 (servers)	Value: 0-00:00:30.0–01 00:00:00.00
---	---

Specifies the interval a node must wait to synchronize. Also specifies synchronization frequency when a node reaches the value specified by the `maximum inaccuracy` characteristic.

time provider present

Specifies whether or not the entity used an external time-provider at the last successful synchronization. This attribute applies to servers only.

time representation version

Specifies the time stamp version (format) used by the installed DECdts software.

type

Specifies whether the node is a DECdts server or clerk. The `create` command modifies this characteristic.

wan timeout

Default: 0-00:00:15.000	Value: 0-00:00:00.000–0-00:10:00.000
--------------------------------	---

Specifies the amount of time the node waits for a response to a WAN synchronization request before declaring that a global server is unavailable. If you set the value to 0-00:00:00.000, global servers are not queried.

Counters**clock settings**

Specifies the number of times the node clock has been set nonmonotonically (abruptly).

creation time

Specifies the time at which the DECdts entity was created and the counters were initialized.

different epochs detected

Specifies the number of times the server received time response messages from servers that had epoch numbers different from its own.

disable directives completed

Specifies the number of times DECdts has been disabled.

enable directives completed

Specifies the number of times DECdts has been enabled.

epoch changes completed

Specifies the number of times the server's epoch has changed.

global responses sent

Specifies the number of times the node has responded to a synchronization request from a node outside the LAN.

insufficient resources detected

Specifies the number of times the node has been unable to allocate virtual memory.

invalid messages detected

Specifies the number of times the local node failed to process a received message that had an unknown message type or incorrect length.

local responses sent

Specifies the number of times the node has responded to a synchronization request from another node in the LAN.

local times not intersecting

Specifies the number of times the node's time interval failed to intersect with the computed interval of the servers.

no global servers detected

Specifies the number of times the server could not contact any global servers.

protocol mismatches detected

Specifies the number of times the local node failed to process a received message containing an incompatible protocol version.

server times not intersecting

Specifies the number of times a server has detected faulty servers (other than itself).

servers not responding

Specifies the number of times the courier server could not contact a specific global server.

synchronizations completed

Specifies the number of times the node has successfully synchronized.

system errors detected

Specifies the number of times a DECdts operation has detected a system error.

time provider failures detected

Specifies the number of times the external time-provider signaled a failure or the node was unable to access the time-provider.

time representation mismatches detected

Specifies the number of times the local node failed to process a received message containing an incompatible time stamp format.

too few servers detected

Specifies the number of times a node failed to synchronize because it could not contact the required minimum number of servers.

too many servers detected

Specifies the number of times the responses to a synchronization request overflowed the assigned system buffer.

updates initiated

Specifies the number of times a server has attempted to update its clock.

Status**current time**

Specifies the current time on the node.

last synchronization

Specifies the computed time at the last synchronization.

state

Specifies the state of the DECdts software.

Off	The DECdts process is disabled.
On	The DECdts process is enabled.
Synchronizing	The DECdts process is synchronizing.

Updating	The DECdts process is updating the time.
----------	--

uid

Specifies the entity's unique identifier, which is generated when the entity is created.

Examples

The examples in this section list some uses of the `show` command.

1. The following command shows how to display the current time:

```
ncl> show dtss current time
Node 0:. DTSSAT 2019-02-22-11:56:40.153-05:00I0.110Status
    Current Time                = 2019-02-22-11:56:40.153-05:00I0.110
```

2. The following command shows how to display all of the entity's characteristic attribute settings:

```
ncl> show dtss all char
Node 0 DTSS
AT 2019-02-05-16:44:17.312-05:00I0.107
Characteristics
    Advertisement Interval      = +0-01:30:00.000Iinf
    Check Interval              = +0-01:30:00.000Iinf
    Epoch Number                = 0
    Error Tolerance              = +0-00:10:00.000Iinf
    Local Time Differential Factor = -0-05:00:00.000Iinf
    Maximum Inaccuracy           = +0-00:00:00.200Iinf
    Servers Required             = 3
    Query Attempts               = 3
    LAN Timeout                  = +0-00:00:00.000Iinf
    WAN Timeout                  = +0-00:00:15.000Iinf
    Synchronization Hold Down    = +0-00:05:00.000Iinf
    Type                         = Server
    Courier Role                  = Backup Courier
    Acting Courier Role           = NonCourier
    Clock Adjustment Rate         = 10000000    nsec/sec
    Maximum Clock Drift Rate      = 50000      nsec/sec
    DNA Version                  = V1.0.0
    DTSS Version                  = V2.0.0
    Time Representation Version    = V1.0.0
    Service Trace                 = True
    Communication Trace           = True
    Global Directory              = LOCAL:.DTSS_GlobalTimeServers
    Time Provider Present         = False
    Automatic TDF Change         = True
    Next TDF Change              =
2019-04-05-03:00:00.000-04:00I0.000
    Clock Resolution              = 3970000      nsec
```

3. The following command shows how to display the entity's status:

```
ncl> show dtss all status
Node 0:. DTSS
AT 2019-02-21-16:35:15.077-05:00I0.099
Status
    Current Time                = 2019-02-21-16:35:15.073-05:00I0.099
    Uid                         = 5F000004-D0D0-11C9-9006-08002B0DC030
```

```
Last Synchronization      = 2019-02-21-16:34:55.066-05:00I0.094
State                     = On
```

Related Information

set command

show dtss decnet global servers

show dtss decnet global servers — Displays current information about the dtss DECnet global servers entity.

Format

show [**node** *node-id*] **dtss decnet global servers** *full name attribute-specifier* [...]

Arguments

attribute-specifier

Specifies the name of an attribute group or individual attribute to be displayed.

Description

The show command displays the names and values of the attributes or attribute groups named in *attribute-specifier*. You can use any combination of attribute specifiers in any order in a single command; use a comma to separate specifiers.

If you do not supply an attribute specifier, the command displays all characteristics and their values. The following are valid attribute specifiers:

- all [characteristics]
- all characteristics
- all identifiers
- all status
- attribute-name

As shown in this list, *attribute-name* is the name of a specific attribute from one of the listed groups. The following sections list names of attributes, categorized by group.

Note that when you enter the show dtss decnet global servers command at a DTSS global server, the list does not include the server itself. To verify what DECdts role the current node is running, enter the following command:

```
ncl> show dtss type
```

Characteristics

transport

Specifies the transport protocol used to communicate with the global server.

Identifier

name

Specifies the full name of the global server, as registered in the namespace.

Status

last observed skew

Specifies the skew between the node's clock and the time received from the specified global server after the most recent synchronization.

last observed time

Specifies the time the node last received from the specified global server.

last time polled

Specifies the time the node last solicited a time value from the specified global server.

used in last synchronization

Specifies whether or not a time value from the specified global server was used in the node's last successful clock synchronization.

Examples

The examples in this section list some uses of the `show` command.

1. The following command shows how to display the transport used by a global server known to the local node:

```
ncl> show dtss decnet global server .DTSS_GlobalTimeServers.eng
transport
Node 0:. DTSS Decnet Global Server ABC:.DTSS_GlobalTimeServers.eng
AT 2019-02-21-17:26:09.073-05:00I0.096
Characteristics
    Transport                      = DECnet
```

2. The following command shows how to display all attributes of the global servers known to the DECdts clerk or server on the local system:

```
ncl> show dtss decnet global server * all
Node 0:. DTSS Decnet Global Server ABC:.DTSS_GlobalTimeServers.sales
AT 2019-02-21-17:34:27.358-05:00I0.092
Identifiers
    Name                          = ABC:.DTSS_GlobalTimeServers.sales
Status
    Last Time Polled              = 2019-02-21-16:54:45.114-05:00I0.101
    Last Observed Time            = 2019-02-21-16:54:45.105-05:00I0.107
    Last Observed Skew            = 0-00:00:00.009I0.000
    Used In Last Synchronization = True
Characteristics
    Transport                      = DECnet

Node 0:. DTSS Global Server ABC:.DTSS_GlobalTimeServers.small
```

```
AT 2019-02-21-17:34:27.365-05:00I0.092
Identifiers
  Name = ABC:.DTSS_GlobalTimeServers.small
Status
  Last Time Polled = 2019-02-21-16:54:45.114-05:00I0.101
  Last Observed Time = 2019-02-21-16:54:45.105-05:00I0.105
  Last Observed Skew = 0-00:00:00.009I0.000
  Used In Last Synchronization = True
Characteristics
  Transport = DECnet
```

Related Information

`show dtss decnet local servers` command

show dtss decnet local servers

`show dtss decnet local servers` — Displays current information about the `dtss DECnet local servers` entity.

Format

show [**node** *node-id*] **dtss decnet local servers** *full name attribute-specifier* [...]

Arguments

attribute-specifier

Specifies the name of an attribute group or individual attribute to be displayed.

Description

The `show` command displays the names and values of the attributes or attribute groups named in *attribute-specifier*. You can use any combination of attribute specifiers in any order in a single command; use a comma to separate specifiers.

If you do not supply an attribute specifier, the command displays all characteristics and their values. The following are valid attribute specifiers:

- `all [characteristics]`
- `all characteristics`
- `all identifiers`
- `all status`
- `attribute-name`

As shown in this list, *attribute-name* is the name of a specific attribute from one of the listed groups. The following sections list names of attributes, categorized by group.

Note that when you enter the `show dtss decnet local servers` command at a DTSS local server, the list does not include the server itself. To verify what DECdts role the current node is running, enter the following command:

```
ncl> show dtss type
```

Characteristics

transport

Specifies the transport protocol used to communicate with the local server.

Identifier

address

Specifies the DECnet address of the server.

Status

last observed skew

Specifies the skew between the node's clock and the time received from the specified local server after the most recent synchronization.

last observed time

Specifies the time the node last received from the specified known server.

last time polled

Specifies the time that the node last solicited a time value from the specified known server.

used in last synchronization

Specifies whether or not a time value from the specified known server was used in the node's last successful clock synchronization.

Examples

The examples in this section describe some uses of the `show` command.

1. The following command shows how to display the transport for a local server known to the local node:

```
ncl> show dtss decnet local servers 00-00-04-00-01-fc transport
Node 0:. DTSS Local Server 00-00-04-00-01-fc (63.1)
AT 2019-02-21-17:19:50.645-05:00I0.097
Characteristics
    Transport                      = IEEE802.2
```

2. The following command shows how to display all the local servers known to the clerk or server system, and lists all their attributes:

```
ncl> show dtss decnet local servers * all
Node 0:. DTSS Decnet Local Server 10-00-04-00-02-fc (63.2)
AT 2019-02-21-16:56:05.151-05:00I0.098
Identifiers
    ADDRESS                      = 10-00-04-00-02-fc (63.2)
Status
    Last Time Polled             = 2019-02-21-16:54:45.114-05:00I0.101
```

```
Last Observed Time           = 2019-02-21-16:54:45.105-05:00I0.105
Last Observed Skew           = 0-00:00:00.009I0.000
Used In Last Synchronization = True
Characteristics
  Transport                   = IEEE802.2 _____
Node 0:. DTSS Decnet Local Server 11-00-04-00-03-fc (63.3)
AT 2019-02-21-16:56:05.155-05:00I0.098
Identifiers
  ADDRESS                     = 11-00-04-00-03-fc (63.3)
Status
  Last Time Polled            = 2019-02-21-16:54:45.114-05:00I0.101
  Last Observed Time          = 2019-02-21-16:54:45.109-05:00I0.107
  Last Observed Skew          = 0-00:00:00.005I0.000
  Used In Last Synchronization = True
Characteristics
  Transport                   = IEEE802.2
```

Related Information

show dtss decnet global servers command

synchronize dtss

synchronize dtss — Causes the DECdts server or clerk to initiate a synchronization.

Format

synchronize [**node** *node-id*] **dtss** [*argument*]

Arguments

set clock *Boolean*

Specifies whether the clock is immediately set or gradually adjusted to the computed time. This argument is optional.

false	The clock is gradually adjusted. This is the default condition.
true	The clock is immediately set.

Description

This command causes the DECdts process to initiate a synchronization on the specified system. If the synchronization is successfully completed, the system clock is adjusted or set to the new computed time. This command overrides the functions of the `synchronization hold down` characteristic.

Note

The `synchronize` command does not execute unless the entity is in the on state.

Exceptions

System Error

The command failed due to an operating system error.

Wrong State

The `dtss` entity is not in the `on` state.

Examples

The following examples show some uses of the `synchronize` command.

1. `ncl> synchronize node reback dtss`

This example shows how to initiate a synchronization for a remote node, followed by a gradual adjustment (update) to the clock on that node.

2. `ncl> synchronize dtss set clock true`

This example shows how to initiate a synchronization for the local system, followed by an immediate reset of the clock.

update dtss

`update dtss` — Gradually adjusts the clock on the specified server node to a new time, beginning at the time specified by the argument.

Format

update [**node** *node-id*] **dtss** *argument*

Arguments

time *absolute-time*

Specifies the absolute time that begins clock adjustment. This argument is required.

Description

Gradually adjusts the system clock to a new time, beginning at the time specified in the argument. The difference between the current clock value and the absolute time specified in the argument is used to adjust the clock. The `update` command is valid only for servers.

Exceptions

Invalid Time

The specified time and inaccuracy must be within the interval formed by the current time and inaccuracy.

System Error

The command failed because of an operating system error.

Wrong State

The `dtss` entity is not in the `on` state.

Examples

The following example shows how to update the time for a server; the clock is gradually adjusted to a new time, beginning at the specified time.

```
ncl> update dtss time 2019-12-30-11:24:00.000-05:00I02.000
```

Appendix A. Event Messages

This section alphabetically lists and describes all events for the `dtss` entity.

clock set

This event is generated each time a DECdts process sets its local clock.

Arguments:

new time	Specifies the corrected clock setting.
old time	Specifies the faulty time before the clock was set.

different epoch detected

This event is generated each time the local node receives a time value that has an epoch number different from its own. This event normally occurs after a node's epoch has been changed.

Arguments:

local epoch number	Specifies the local node's epoch number.
server [id]	Specifies the known servers and their epoch numbers.

disable completed

This event is generated each time a `disable` command is executed to stop DECdts.

enable completed

This event is generated each time an `enable` command is executed to start DECdts.

epoch change completed

This event is generated each time the epoch is set for the local node.

Arguments:

new epoch	Specifies the epoch number after the epoch was declared.
old epoch	Specifies the epoch number before the epoch was declared.

insufficient resources detected

This event is generated each time the node's virtual memory is full. The event is displayed on all nodes and is usually followed by a system abort.

invalid message detected

This event is generated each time the local node receives a corrupted or incorrect message. Invalid messages are normally retransmitted, but the hexadecimal contents of the invalid message are supplied for troubleshooting.

Arguments:

message	Specifies the hexadecimal contents of the invalid message.
node [id]	Specifies the full name of the node that generated the invalid message.

local time does not intersect

This event is generated each time the time interval of the local node does not intersect with the values received from other servers. If this event is generated frequently, you can decrease the setting for the characteristic `synchronization hold down` or increase the settings for the characteristics `error tolerance` or `maximum inaccuracy`.

Arguments:

clock set	Specifies whether or not the clock was automatically set to correct the time on the local system. If false , the clock was not set to correct the time on the local system.
computed time	Specifies the intersection of the majority of server times.
local time	Specifies the local time interval that does not intersect with the computed time .

no global server

This event is generated at a courier server when it cannot locate a global server.

protocol version mismatch detected

This event is generated each time the DECdts protocol version of a received message does not match the protocol in force at the local node.

Argument:

node [id]	Specifies the full name of the node that generated the message with the incompatible protocol version.
------------------	--

server not responding

This event is generated at a courier server when it attempts to synchronize with the listed global server, but the server does not respond. The event may be caused by a low characteristic setting for `wan timeout`.

Argument:

node [id]	Specifies the node name of the server not responding.
------------------	---

server time does not intersect

This event is generated each time the local node receives a faulty time value from a server. If continuing **faulty servers detected** events are generated for the same node, examine the node's settings for these characteristics: `error tolerance`, `synchronization hold down`, and `maximum inaccuracy`.

Argument:

server [id]	Specifies the list of servers used in the synchronization.
--------------------	--

synchronization completed

This event is generated each time a synchronization is completed for the local node.

Arguments:

clock adjustment	Specifies the relative time the local clock needs to be adjusted to match the computed time.
computed time	Specifies the time computed from the intersection of all server time intervals.

system error detected

This event is generated each time the operating system has a system error relevant to DECdts operation. See the system documentation for an explanation of the report text.

Argument:

error text	Specifies the system error report.
-------------------	------------------------------------

time provider failure detected

This event is generated each time an external time-provider returns an invalid time or does not respond.

time representation version mismatch detected

This event is generated each time the local node receives a response indicating that it is using an invalid time representation version.

Argument:

node [id]	Specifies the name of the node that is running the incompatible time representation version.
------------------	--

too few servers detected

This event is generated each time the number of servers specified by the `servers required` characteristic are not available for synchronization with the local node.

Arguments:

number detected	Specifies the number of servers that responded to the synchronization request.
number required	Specifies the number of servers required for synchronization.

too many servers detected

This event is generated each time server responses overflow the synchronization list buffer of the local node. The difference between the **maximum delivered** and **number accepted** arguments is the number of responses that overflowed the buffer.

Arguments:

number accepted	Specifies the number of server responses that were accepted by the node.
number detected	Specifies the number of servers responding to a solicitation.

update initiated

This event confirms an attempt to change the system time resulting from the `update` command.

Arguments:

current time	Specifies the current system time.
update time	Specifies the new, updated system time.

Appendix B. Time-Providers and Time Services

This appendix explains the criteria to use when selecting a time-provider, and describes time dissemination services, time-providers (hardware and/or software), and their interaction with DECdts. The appendix also contains a world time zone map.

B.1. Criteria for Selecting a Time Source

Before you select a time source for your network, ask the following questions:

- *How accurate is the time provided?*

Accuracy is affected by the time source itself, as well as the transmission media. As long as the inaccuracy is known, DECdts can compensate for it.

- *How reliable is the time source?*

The time source should always be available. If the time source is not available, the server connected to the time-provider uses times from other servers and compensates for any time difference when the source again becomes available.

- *What is the extent of coverage?*

The time source must be available in the geographical area where the time-provider server is located.

- *What is the level of known inaccuracy?*

If this is known, DECdts can compensate for it. Most sources have known inaccuracy levels.

- *What is the cost?*

- **Does the source conform to the operating environment?**

The available power supply and physical conditions must be compatible with the source; consult the supplier for specifications.

Table B.1, "Time-Provider Selection Criteria" summarizes the selection criteria for each type of time source.

Table B.1. Time-Provider Selection Criteria

Type	Selection Criteria	
	Coverage	Inaccuracy
Telephone		
NIST	Regional	10 ms
Radio		
MSF	Europe	10 ms
WWV	N. America	100 ms

Type	Selection Criteria	
	Coverage	Inaccuracy
Telephone		
WWVB	N. America	10 ms
	Europe	
WWVH	Eastern &	100 ms
	Central N. Pacific	
Satellite		
GOES	W. Hemisphere	1 ms corrected
GPS	Worldwide	< 100 ns
TRANSIT	Worldwide	1 ms

B.2. Sources of Coordinated Universal Time

There are several sources of UTC time including telephone, radio, and satellite, as described in the following sections.

B.2.1. Telephone Services

Telephone time-provider services require the time-provider software to dial a centralized UTC time source through a modem. Modem speeds and line delays can affect the accuracy of the time returned.

Telephone time-provider services are usually provided by standards agencies. For example, in the United States this service is offered by the National Institute of Standards and Technology (NIST) with their Automated Computer Time Service (ACTS), and by the U.S. Naval Observatory with their Automated Data Service (ADS). There is often a per-call fee for the service in addition to the cost of the modem software.

Sample time-provider programs you can use with NIST's ACTS are supplied with the DECdts kit in the following:

```
sys$common:[syshlp.examples.dtss]dtss$provider_acts.c
```

Comments in the programs explain their use.

For telephone time-provider services in other countries, contact the applicable standards organization listed in *Table B.2, "UTC Radio Stations and Managing Authorities"*.

B.2.2. Radio Broadcasts

Commercial radio receivers that monitor time and frequency broadcasts can return time values back through the time-provider interface (TPI) to a DECdts server. Standards organizations usually operate radio stations that broadcast time and frequency signals. In the United States, NIST operates the following time and frequency stations:

- WWV

Transmits at 2.5, 5, 10, 15 MHz to North America and South America.

- **WWVB**

Transmits at 60 kHz primarily to the United States, providing high-quality frequency information because atmospheric propagation effects are relatively minor.

- **WWVH**

Transmits at 2.5, 5, 10, 15 MHz to Alaska, Hawaii, Australia, New Zealand, Japan, and Southeast Asia.

In addition to the stations listed above, more than 30 radio stations worldwide provide UTC time. Use the partial list in *Table B.2, "UTC Radio Stations and Managing Authorities"* and consult the national standards organization in your country for further information.

Table B.2. UTC Radio Stations and Managing Authorities

Call Letters	Authority
ATA	National Physical Laboratory Dr. K.S. Krishnan Rd. New Delhi, - 110012, India
BPM	Shaanxi Astronomical Observatory Chinese Academy of Sciences P.O. Box 18 - Lintong Shaanxi, China
BSF	Telecommunication Laboratories Directorate General of Telecommunications Ministry of Communications P.O. Box 71 -Chung-Li 32099 Taiwan, R.O.C.
CHU	National Research Council Institute for National Measurement Standards - Time Standards Ottawa, Ontario, Canada K1A0R6
DCF77	Physikalisch-Technische Bundesanstalt, Lab. Zeiteinheit Bundesallee 100 W - 3300 Braunschweig Federal Republic of Germany
EBC	Real Instituto y Observatorio de la Armada - San Fernando Cadiz, Spain
HBG	Service horaire HBG Observatoire Cantonal CH - 2000 Neuchatel, Switzerland
HLA	Time and Frequency Laboratory Korea Standards Research Institute P.O. Box 3, Taedok Science Town Taejon 305-606, Republic of Korea
IBF	Istituto Elettrotecnico Nazionale Galileo Ferraris Strada delle Cacce, 91

Call Letters	Authority
	10135 - Torino, Italy
JJY JG2AS	Standards and Measurements Div. Communications Research Laboratory 2-1, Nukui-kitamachi 4-chome Koganei-shi, Tokyo 184 Japan
LOL	Director, Observatorio Naval Av. Espana 2099 1107 -Buenos Aires, Republica Argentina
MSF	National Physical Laboratory Division of Electrical Science Teddington, Middlesex TW11 OLW United Kingdom
PPE PPR	Departemento Servico da hora Observatorio Nacional (CNPq) Rua General Bruce, 586 20921 Rio de Janeiro - RJ,Brazil
TDF	Centre National d'Etudes des Telecommunications - PAB -STC Etalons de frequence et de temps 196 Ave. Henri Ravera 92220- Bagneux, France
WWV WWVB WWVH	Time and Frequency Division National Institute of Standards and Technology 325 Broadway Boulder, Colorado 80303, U.S.A
YVTO	Direccion de Hidrografia y Navegacion Observatorio Cagigal Apartado Postal No. 6745 Caracas, Venezuela

A sample time-provider program (`sys$common:[syshlp.examples.dtss]dtss $provider.c`) you can use with UTC radio receivers is supplied with the DECdts kit. Comments in the program explain its use. For a list of supported radio receivers, see *Table B.3, "Radio Receiver Manufacturers"*. Neither the listing nor the availability of the sample time-provider programs imply any endorsement of the receivers by VSI.

Table B.3. Radio Receiver Manufacturers

Model	Manufacturer
GC-100	Heathkit Heath Company Benton Harbor, MI 49022 U.S.A. Tel. 616-982-3200
6020	Hopf Elektronik, Gmbh Postfach 1847 D-5880 Luedenscheid

Model	Manufacturer
	Federal Republic of Germany Tel. 49-(0)2351-45038
OEM-10	PSTI (See Traconex)
Netclock/2	Spectracom Corp. 101 Despatch Drive Rochester, NY 14445 U.S.A. Tel. 716-381-4827
1020	Traconex Corp. 3510 Bassett St. Santa Clara, CA 95054 U.S.A. Tel. 408-727-0260

B.2.3. Network Time Protocol

Nodes that have Internet access can use the Network Time Protocol (NTP) as a source of UTC time for DECdts. See *Appendix C, "Interoperation with NTP"* for an explanation of how to use NTP as a time-provider.

B.2.4. Satellite

Satellites have worldwide availability; they can provide relatively precise times if their delays are known and compensated for. See the following list for satellite sources of UTC.

- *GOES*

Geostationary Operational Environmental Satellite; a satellite network operated by the U.S. National Oceanic and Atmospheric Administration (NOAA). Coverage is limited to the Western Hemisphere.

- *GPS*

Global Positioning System; a satellite system operated by the U.S. Department of Defense. GPS is available worldwide.

- *TRANSIT*

U.S. Navy satellite system; available worldwide.

B.3. World Time Zone Map

Figure B.1, "World Time Zone Map" shows a map of the world time zones, including the following:

- UTC (GMT) reference zone
- Odd- and even-numbered zones
- Half-hour zones
- Countries and areas that have not adopted the zone system or in which time differs less than 1 hour from the neighboring zone

Appendix C. Interoperation with NTP

This chapter explains how to implement both DECdts and NTP in the same DECnet-Plus network.

NTP is an Internet-recommended standard for synchronizing the time on Internet nodes. The NTP synchronization subnetwork is represented by a tree-structured graph with nodes representing time servers and edges representing the transmission paths between them. The root nodes of the tree are designated primary servers, which synchronize to a radio broadcast or calibrated atomic clock. Remaining nodes are designated secondary servers, which synchronize to other servers (primary and secondary).

The number of subnetwork hops between a particular server and a primary server determines the stratum of that server (the smaller the number of hops, the lower the stratum). A lower stratum server always has a higher accuracy than a higher stratum server. All servers have identical functionality and can operate simultaneously as clients of the next lower stratum and servers for the next higher one.

Servers, both primary and secondary, typically run NTP with several other servers at the same or lower strata. A selection algorithm attempts to select the most accurate and reliable server or set of servers from which to actually synchronize the local clock.

NTP and DECdts can both be used in large computer networks that have embedded local networks (connected by routers, gateways, and bridges) and use both broadcast and point-to-point transmission media. DECdts and NTP can run simultaneously on the same LAN. The following sections describe giving time to and getting time from local and remote NTP time sources, and preventing loops.

C.1. Getting the Time from Remote NTP Time Sources

The DECdts kit contains four NTP time-provider files (in the `sys$examples:directory`) that allow DECdts to interoperate with NTP:

<code>dtss\$provider_ntp.c</code>	DECdts NTP Provider Program
<code>dtss\$provider_ntp.exe</code>	DECdts NTP Provider Image
<code>run_ntp_tp.com</code>	Procedure to run DTSS\$PROVIDER_NTP
<code>start_detached_ntp_tp.com</code>	Procedure to run as a detached process

The `dtss$provider_ntp` program takes the time from a NTP server as it would from a radio receiver. To use either program, you must specify the name of the NTP server and the inaccuracy.

Run the DECdts server with the `dtss$provider_ntp` time-provider on a node with access to a NTP server. Specify the inaccuracy in a manner that is consistent with local NTP experience.

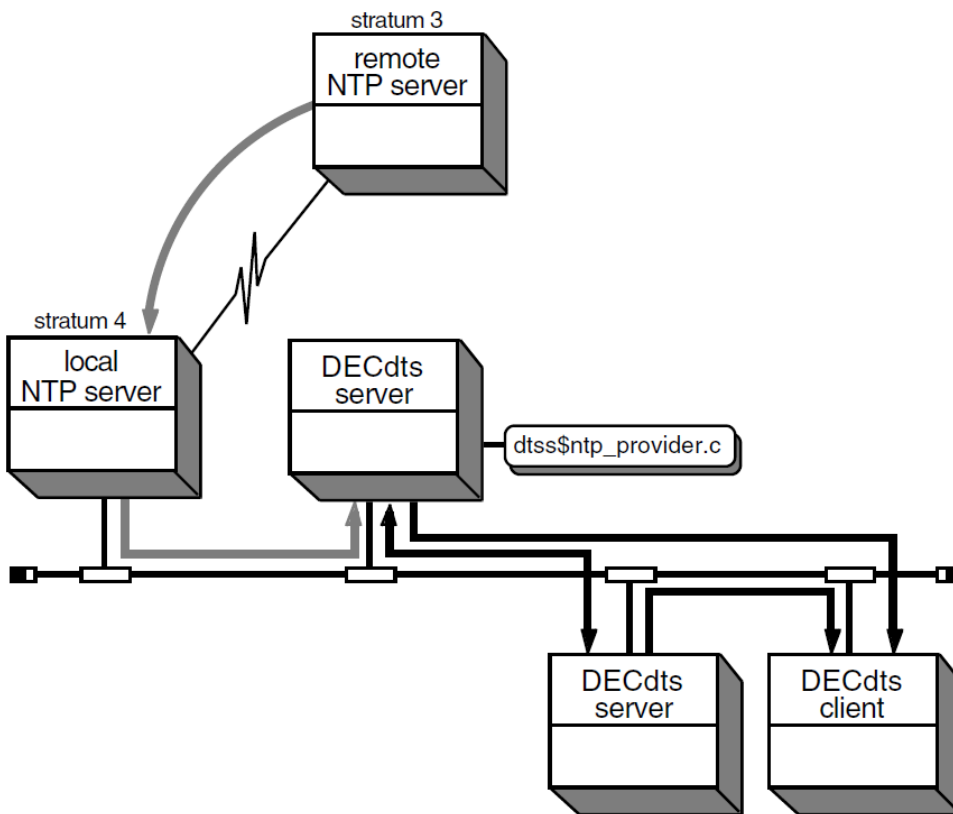
The following advisories should be observed:

- **Advisory:** If links to remote sources are distant, you may want one of the subnetwork nodes to run NTP locally.

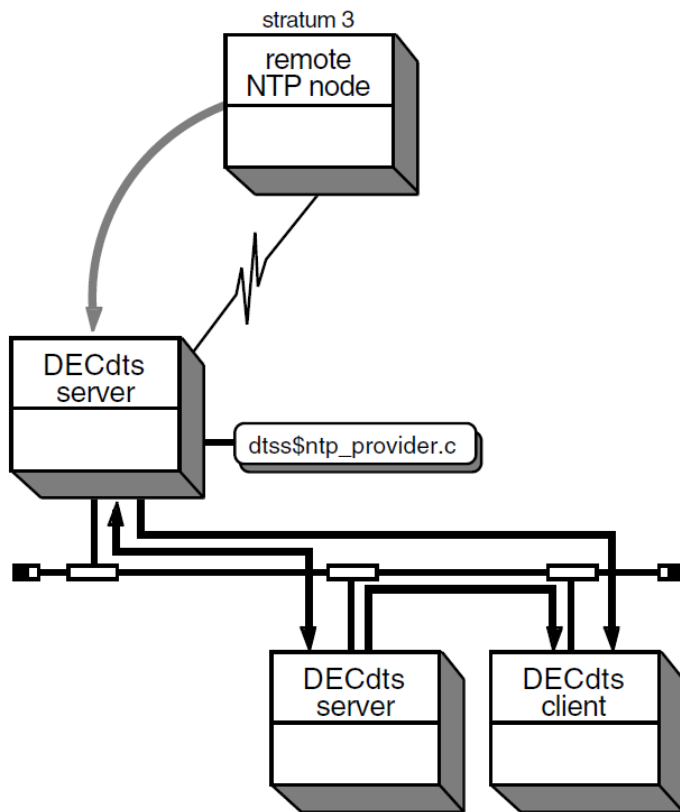
- **Advisory:** The NTP time-provider will not accept time from a NTP node at stratum 8 or higher.
- **Advisory:** The NTP node should be as close to stratum 1 as possible.

Figure C.1, "Getting the Time from a Remote NTP Time Source (Scenario 1)" and Figure C.2, "Getting the Time from a Remote NTP Time Source (Scenario 2)" both show a DECdts server getting the time from a remote NTP time source (a stratum 3 server). In the first scenario, however, all of the recommendations made in Section C.1, "Getting the Time from Remote NTP Time Sources" are followed; in the second scenario, one of the recommendations (running NTP locally on one of the subnetwork nodes if the link to a remote source is distant) is ignored.

Figure C.1. Getting the Time from a Remote NTP Time Source (Scenario 1)



ZK-2959A-GE

Figure C.2. Getting the Time from a Remote NTP Time Source (Scenario 2)

ZK-2960A-GE

C.2. Giving the Time to NTP Nodes

Any DECdts server or clerk that also runs the NTP software can be configured as a NTP server. The `tcpip$ntp.conf` configuration file controls the NTP configuration. *VSI TCP/IP Services for OpenVMS Management* for information about how to configure the NTP software as a server that gets its time from DECdts.

In this configuration, NTP *never* sets the clock. NTP can, however, give the time to other NTP clients. Note that loops between DECdts and NTP should not be allowed to form; if NTP gives the time to DECdts, then DECdts gives the time back to the *same* set of NTP servers, unexpected results can occur.

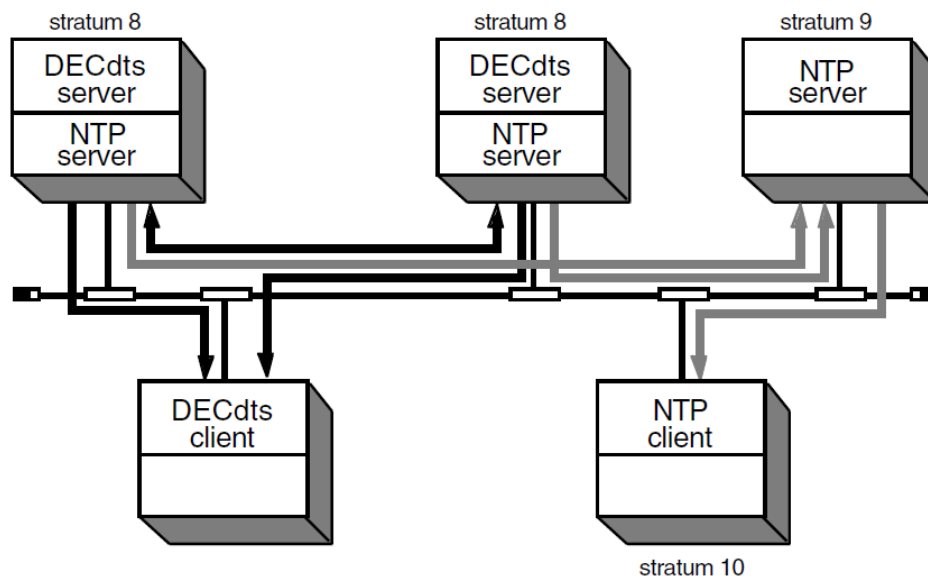
The NTP configuration file is set up to ensure that a NTP server which obtains the time from DECdts is a stratum 8 node. In addition, the `dtss$ntp_provider` is prohibited from accepting time from a node at stratum 8 or higher.

A DECdts (server) node can give time to a NTP node if the following rules and advisories are observed:

- **Rule:** The NTP server must be at stratum 8.
- **Advisory:** Multiple nodes in the set can be running the NTP software configured as an NTP server.
- **Advisory:** If any DECdts-managed system has a local time source, that system should be used as a NTP server.
- **Advisory:** Although this operation can occur on either a DECdts server or a DECdts client node, a DECdts server is preferred.

Figure C.3, "Giving the Time to NTP" shows two DECdts server nodes running NTP servers and providing time to a NTP sub network. The `tcpip$ntp.conf` file defines these servers at stratum 8.

Figure C.3. Giving the Time to NTP



ZK-2961A-GE

C.3. Preventing Loops

Regardless of the circumstances, you must never allow loops, such as the following, to form:

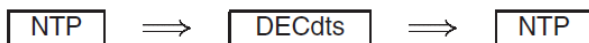
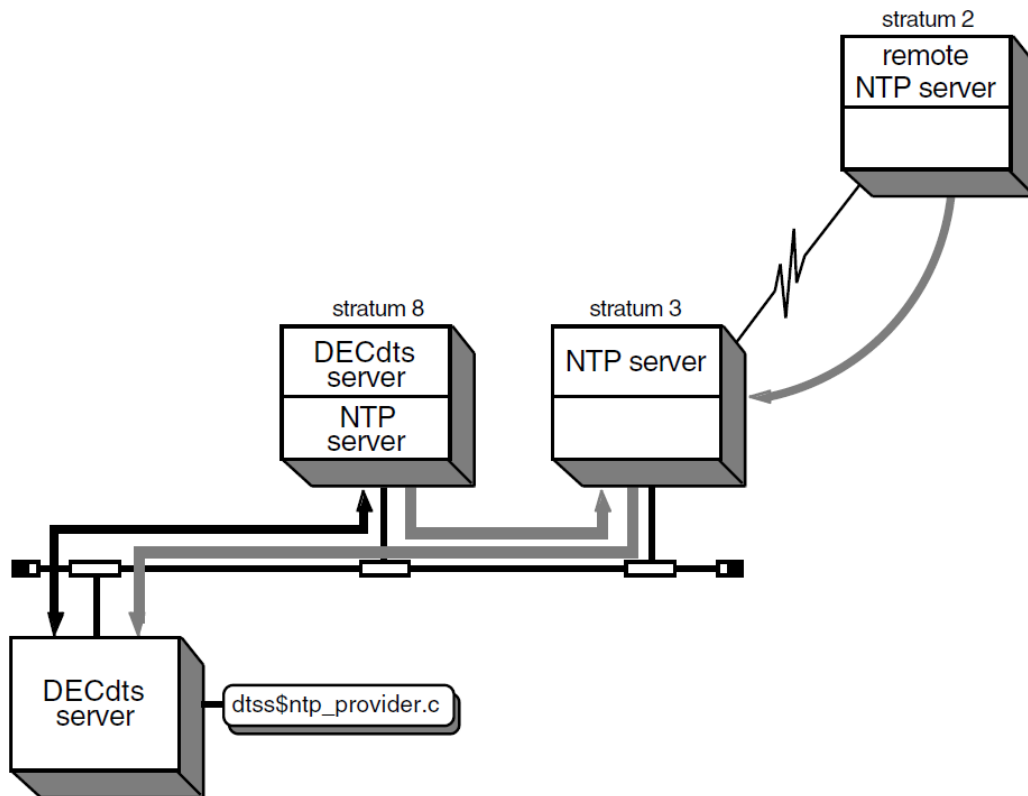
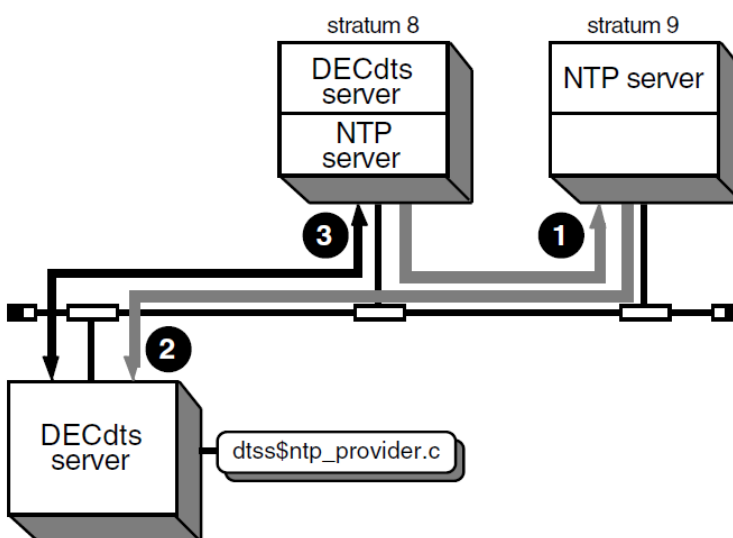


Figure C.4, "Configuration Before Stratum 2 Node Fails" shows a configuration that VSI does not recommend. This configuration works only as long as the remote NTP stratum 2 node does not fail.

Figure C.4. Configuration Before Stratum 2 Node Fails

ZK-2962A-GE

If the remote NTP stratum 2 node goes down, the stratum 3 node starts accepting time from the stratum 8 node. Once this occurs, the stratum 3 node drops to stratum 9, as shown in *Figure C.5, "Configuration After Stratum 2 Node Fails"*.

Figure C.5. Configuration After Stratum 2 Node Fails

ZK-2963A-GE

This scenario shows the creation of a loop:

1. From the node labeled stratum 8, proceed to the NTP node labeled stratum 9.
2. From the NTP node labeled stratum 9, continue to the node running `dtss$ntp_provider.c`.
3. The DECdts node running the time-provider then feeds the time back to the NTP daemon in the node labeled stratum 8, creating a loop.

If this occurs, time in the NTP and DECdts subnetwork can arbitrarily drift from UTC.