Layman's Guide to NICI

Distribution	Public
Classification	Unclassified
Last Updated	January 12, 2006
Author	NICI Team, part of eDirectory Core
Pages	12
Version	2.0

Table of Contents

1 Introduction

NICI is Novell International Cryptography Infrastructure first shipped on NetWare 5.0. This document is prepared in order to help resolve NICI issues typically in the field or during testing of various Novell or third party products. A particular product may use NICI directly or indirectly via another module (NLM, DLL, so, etc.).

All actions described here don't have any warning or an "Are you sure?" question. They may and will cause unrecoverable data loss, and must be executed with the full knowledge of such an action. Most NICI problems as well as solutions have implications in other products. It may not be easy to predict the effects of taking a NICI action: NICI provides one of the most critical services in the system and its inoperable condition typically renders the system inoperable as well as causing a permanent and unrecoverable damage. If certain NICI keys are lost in an irrecoverable fashion, even backed-up data may be useless, as it can't be decrypted.

Contents of this document are no guarantee for a fix, but are advisory.

Table 1. NICI Configuration Directory.

Platform	Start-up Directory	NICI Directory	NICI User directory
NetWare	C:\NWSERVER	SYS:\SYSTEM\NICI	SYS:\SYSTEM\NICI
Microsoft Windows	%SystemRoot%\System32	%SystemRoot%\System32\Novell\NICI (See Section 3)	%SYSTEMROOT%\System32\Novell\NICI (See Section 3)
UNIX	/usr/lib	/var/novell/nici (See Section 3)	/var/novell/nici/ (See Section 3)

2 NICI Modules

NICI is shipped or will be shipped on multiple platforms: NetWare, Microsoft Windows 95/98/Me/NT/2000, Linux (ix86), Solaris (SPARC), and AIX. Except on the NetWare platform, it is a shared library (DLL, so, etc), whereas it is comprised of multiple signed NLMs, called XLMs, on the NetWare platform. On non-NetWare platforms, NICI has another piece running in the Dhost environment in server mode distributed as part of an eDirectory release.

2.1 NetWare

NICI on NetWare has multiple signed NLMs, called XLMs. The "modules" command displays the names of NLMs, and not the XLMs. The start-up directory is typically C:\NWSERVER.

2.1.1 CCS.XLM (ccs.nlm)

This file is located in the Start-up directory, and is the only XLIB module that exports APIs used by other NLMs.

2.1.2 XMGR.XLM (xmgr.nlm)

This module is located in the Start-up directory. It has no useable APIs by other NLMs.

2.1.3 EXPXENG.XLM (xengnul.nlm ,xengexp.nlm,xngaexp.nlm)

This module is located in the Start-up directory. The presence of these NLMs identifies the availability of weak/exportable cryptography.

2.1.4 DOMXENG.XLM (xengnul.nlm,xengexp.nlm,xengusc.nlm,xngausc.nlm)

This module is located in the Start-up directory. The presence of these NLMs identifies the availability of strong/domestic cryptography. As of NICI 2.x, Novell ships strong cryptography worldwide.

2.1.5 XSUP.XLM (xsup.nlm)

This module is located in the Start-up directory.

2.1.6 NICISDI.XLM (nicisdi.nlm)

This module is present in the SYS:\SYSTEM directory and loaded by AUTOEXEC.NCF file. This is the Security Domain Infrastructure management module. If this module is not loaded, then security domain keys (e.g., tree key) are not loaded into NICI and they are not available. It is a typical symptom to get a 1460 error when this module is not loaded.

2.1.7 SASDFM.XLM (sasdfm.nlm)

This module is present in the SYS:\SYSTEM directory and loaded by AUTOEXEC.NCF file. This is the SAS Data Flow Manager file and is responsible for handling NCP communications for session key set-up as well handling client NICI initialization requests. The lack of this module disables session key support in NICI. A typical symptom is not being able to export user certificates in ConsoleOne, or not being able to use NMAS to login to eDirectory.

2.2 Microsoft Windows

2.2.1 NICICCS.SYS and NICICCS.VXD

NICI versions before NICI 2.0 are kernel drivers. On Windows NT/2000 systems, it was called NICICCS.SYS, and was located in the *drivers* directory under *System32* directory. On Windows 95/98 systems, it was called NICICCS.VXD. Kernel versions of NICI are not maintained anymore.

2.2.2 CCSW32.DLL

NICI versions newer than 2.x have a DLL named CCSW32.DLL. These are the FIPS 140 level 1 and level 2 certified modules. Refer to the security policy document for more on the FIPS 140 evaluations. Note that simply copying the DLL into a directory does not make NICI operational as it requires Windows registry and configuration file setup. Additionally, NICI module self verifies itself, hence most components are coupled with the distributed DLL, and usually are not distributable alone. NICI does not depend on directory services to be installed.

2.2.3 NICIEXT.DLM

In the DHost environment, NICI has a DLM called NICIEXT.DLM managing NCP connections and other eDirectory services on behalf of NICI. The DLM is shipped with eDirectory distributions.

2.3 UNIX

2.3.1 libccs2.so

The first version supported on all UNIX platforms is 2.3.0. NICI is a shared object (.so), and named *libccs2.so*. Typically, it is a symbolic link to the actual file named per platform and version. NICI does not depend on directory services to be installed.

2.3.2 libniciext.so

In the DHost environment, NICI has a shared object called *libniciext.so* loaded by DHost to carry out communications and other directory services of behalf of NICI. The shared object is shipped with eDirectory distributions.

3 NICI Setup

We strongly encourage using the NICI install program provided on each platform to install and configure NICI. NICI installed by other means may cause irreparable damage, and support may ask a customer to remove NICI, and perhaps other items such as certificates that a customer may have paid for, and reinstall NICI properly.

3.1 NICI Configuration Files

NICI configuration files are located in the NICI directory listed on Table 1. The NICI configuration files listed on the following table are present on all platforms. Platform-specific files and other configuration details are explained in following sections.

File	Created by	Description
NICIFK	eDirectory install	NICI license material for server-mode of operation.
Xmgrcfg.wks	NICI install	NICI license material for client-mode of operation. Not used if NICIFK is present.
Xmgrcfg.nif	First use of NICI or by install by a privileged user	NICI per-box, unique keying material generated locally.
Xarchive.000	First use of NICI by a privileged user	NICI master archive.

The NICI configuration files are signed and partially encrypted. An invalid license file (NICIFK) or a client license file (xmgrcfg.wks) renders NICI dysfunctional.

3.2 NICI User Configuration Files

A NICI user directory is created by NICI upon first use of NICI by a user, if the directory does not already exist. NetWare does not have user directories, as the system has only one user: the server itself. Likewise, user directories are not created on single-user systems like Microsoft Windows 95/98/Me, if multi-user capability is not configured. NICI sets the rights on each user directory, when it creates the directory, so that only the user has access to it. In order for the system administrator (e.g., Administrator on Windows, root on UNIX) must typically take the ownership of a user directory, and then change its permissions accordingly. Refer to the operating system's file management utilities for more details.

File	Created by	Description
Xmgrcfg.ks2	First use	User-specific key materials and other configuration materials.
Xmgrcfg.ks3	First use or update	User-specific state data, updated occasionally.
Xarchive.001	First use or update	NICI user archive.

3.3 NetWare Configuration

The *SYS:/SYSTEM/NICI/NICISDI.CFG* file is used to configure the NICISDI module's operation parameters. By default, this file does not exist. At present, the only configurable parameter is the synchronization period the *NICISDI.XLM* module checks for new security domain keys. A typical file would contain the following:

```
# This is a sample NICISDI.CFG file for NetWare systems.
# There is only one configuration parameter; all others are ignored.
# The pound sign in the first column marks the
# entire line as a comment, and the line is ignored.

# The time in minutes NICISDI.XLM module polls.
NICISDI Sync Period = 60
```

The *NICISDI.CFG* file is read when the *NICISDI.XLM* module is loaded (as part of *AUTOEXEC.NCF* processing). If the file does not exist, does not contain the period, or if the period is zero, NICISDI does not attempt to read it again. If the file exists and contains a non-zero period, the file is read once in a period before synchronization. One can disable the background synchronization process by deleting the file, setting the period to zero, or commenting out the period line.

The *SYS:\SYSTEM\NICI\NICISDI.KEY* file contains encrypted security domain keys as discussed in Section 4.

3.4 Microsoft Windows Configuration

NICI install creates and populates a key in the Windows registry. The location of the key is HKEY_LOCAL_MACHINE\SOFTWARE\Novell\NICI. The table below describes each value.

Key	**Type**	**Description**
ConfigDirectory	String	Location of NICI configuration files (Section 3.1)
DAC	Binary	NICI module's digital authentication code.
SharedLibrary	String	The name of the library, e.g., ccsw32.dll
Strength	String	U0 for strong, W1 for import restricted (not anymore supported)
UserDirectoryRoot	String	(Optional). Name of a directory where user directories are created. Defaults to ConfigDirectory.
Version	DWORD	NICI version, such as 0x00002400 for 2.4.
NICISDI Sync Period	DWORD	NICISDI synchronization period in minutes, represented in hexadecimal.
EnableUserProfileDirectory	DWORD	NICI user files are created in *"Application Data\Novell\NICI"* directory in the user's profile directory.

Users' directories are created, by default, in *%SystemRoot%\System32\novell\nici* directory by the user's name, e.g., *c:\WINNT\System32\novell\nici\Administrator*. In order to change the root directory in which all user directories are created by name, modify or create the string type registry entry *UserDirectoryRoot* in the NICI registry key, and set it to the desired root directory, for instance *c:\Documents and Settings* to create the NICI user configuration files in each user's local profile path on a Windows 2000 system.

The user name is the name of the user owning the process that started NICI. If it is a local user, NICI uses the user name. If it is a remote or a domain user, NICI forms the user name as the combination of user name and domain separated by a dot, i.e., *userName.domainName*.

EnableUserProfileDirectory is not created by the NICI install, hence disabled. If set, existing NICI user files may need to be copied or moved to the new location. If user profile directory is enabled, NICI does not set the ACLs on this directory, it relies on existing security properties (ACLs, inheritance, and ownership) of the user's profile directory: use this option very carefully as you may disclose all users' NICI keys. NICI creates the *"Application\Novell\NICI"* directory if not present, and stores all NICI user files in this directory. This option is provided to enable the dynamic user creation/deletion feature in the Novell ZEN Works product. It must be set manually or by another application's install, such as ZEN Works.

The *NICIEXT.DLM* module reads the *NICISDI Sync Period* value when DHost loads it. If the value does not exist, or if the period is zero, NICIEXT does not attempt to read it again. If the value exists and contains a non-zero period, the value is read once in a period before synchronization. One can disable the background synchronization process by deleting the value, or setting the period to zero.

The *%SystemRoot%\System32\Novell\NICI\NICISDI.KEY* file contains encrypted security domain keys as discussed in Section 4.

All users have *read*, *execute,* and *create* rights to the files in the NICI configuration directory (*%SystemRoot%\Novell\NICI*). NICI dynamically creates user directories upon first use of NICI by that user, and give full rights only to the user creating the directory.

3.5 UNIX Configuration (/etc/nici.cfg)

The */etc/nici.cfg* file emulates the Microsoft Windows registry in an editable text file. Most of the entries are setup by NICI install. A typical */etc/nici.cfg* file contents are given below.

```
ConfigDirectory:s:16:/var/novell/nici
SharedLibrary:s:19:/usr/lib/libccs2.so
DAC:b:8:1a:aa:6d:49:48:a8:83:98
MkUserDir:s:24:/var/novell/nici/nicimud
NiciVersion:s:5:2.4.0
BuildVersion:s:11:4001101.23
BuildDate:s:6:020123
NiciStrength:s:2:u0
NICISDI Sync Period:b:1:3c
```

Each line may have multiple entries all separated by a column ":". The first entry in a line is the name, followed by its type. The second is the length in decimal, followed by the actual value. There are two types: string (s), and binary (b). For instance, the name of the first line in the sample above is "ConfigDirectory", of type string (s) of 16 characters. The value is "/var/novell/nici". The name of the last line is "NICISDI Sync Period", of type binary (b) of 1 hexadecimal digit; its value is 0x3c, or 60 in decimal, which represents minutes for this particular parameter.

Each line is described on the table below, if not covered in the Microsoft Windows registry section.

Key	Description
MkUserDir	This executable executed to create user directories. */var/novell/nici/nicimud* is supplied by NICI install.
NICIVersion	NICI version string.
BuildVersion	NICI build version string.
BuildDate	NICI module's build date; year, month, and day, each in two decimal digits.
NiciStrength	u0 for strong, w1 for import restricted (not anymore supported).
NICISDI Sync Period	NICISDI synchronization period in minutes, represented in hexadecimal.

The *libniciext.so* module reads the *NICISDI Sync* Period value when DHost loads it. If the value does not exist, or if the period is zero, NICIEXT does not attempt to read it again. If the value exists and contains a non-zero period, the value is read once in a period before synchronization. One can disable the background synchronization process by deleting the value, or setting the period to zero.

The *SYS:\SYSTEM\NIC\<UID>I\NICISDI.KEY* file contains the encrypted security domain keys as discussed in Section 4. The *<UID>* is the numeric user ID defined by the UNIX system. For instance, it is typically 0 for root. Having a *NICISDI.KEY* file per user enables multiple instances of eDirectory running with different user Ids to host multiple trees on the same physical box.

All users have *read* and *execute* (where applicable) rights to the files in the NICI configuration directory (*/var/novel/nici*). Only the installing user has full rights in the configuration directory. User directories are created by a *setuid* executable (*nicimud*, for NICI Make User Directory) provided by NICI install by user Ids. The *nicimud* creates a user directory upon the first use of NICI by that user, and give full rights only to the user creating the directory (0700).

4 NICISDI: Security Domain Infrastructure

NICISDI stands for NICI Security Domain Infrastructure. This module is responsible for managing domain keys, where a domain is typically defined as the whole tree. In future, a directory partition or custom domains can be defined.

Up to NICI version 1.5.x, inclusive, NICI supports one single partition key, the partition being the whole tree. Starting with NICI version 2.0.1, NICI can manage multiple partition keys of varying strengths and algorithms. Such keys are called Security Domain keys.

The *NICISDI.XLM* module on NetWare, *NICIEXT.*DLM on Microsoft Windows, and *libniciext.so* on UNIX platforms manage security domain keys in coordination with NICI. Various other services rely on the availability on security domain keys, including but not limited to, SecretStore/Single-Sign-On, PKI (CertServer), and NMAS.

NICISDI module has nothing to do with the SASDFM module. SASDFM manages session keys between two boxes, typically between a client and a server. The NICISDI.XLM and SASDFM.XLM modules are both loaded during AUTOEXEC.NCF processing on NetWare. Multiple loading of these modules is controlled and should not cause problems given that NICI 1.5.5 or newer is installed on the system.

Security domain servers manage security domain keys. Any server can be configured as a security domain server. There can be multiple security domain servers in a tree. Security domain keys are not intended for clients, either.

One tree key is installed by an eDirectory installation. The tree key is created or retrieved from the security domain key server during the server installation.

4.1 Tree Merge and Split

Merging two or more trees with NICI versions before NICI 2.0.1 caused problems in various components including PKI, NMAS, SecretStore. With NICI 2.0.1, multiple security domain key support and automatic key synchronization is added, reducing such problems short of rebooting a server and adding a server name to a directory attribute. See the directory object section below for more details.

Tree splits do not cause major problems as tree merges do. Nevertheless, it is strongly recommended that existing security domain keys are revoked, and new ones created after a tree split. Revoking old security domain keys still enable access to encrypted data protected by such keys. However, new data must be encrypted with one of the new security domain keys to facilitate cryptographic tree separation. A tool development is under progress for administration of security domain keys.

4.2 Directory Objects

In the directory, the "*Security.KAP.W0*" container off the root has a list of attributes to aid in security domain key management. These attributes are described below:

4.2.1 NDSPKI:SD Key Server DN

This multi-valued attribute contains the list of SD key servers in the tree. There must be at least one server in this list. NICI 2.0.1 and newer versions, which is distributed with 6Pack, make use of this attribute. NICISDI or NICIEXT reads this attribute on each loading (typically server boot). Then, NICISDI or NICIEXT connects to each server in this list, and requests any new security domain keys from each server in this list. Existing security keys are also checked for revocation. However, deletion of a security domain key is not automatically done. Only new key retrieval (not creation) and key revocation is automatically done on every loading of NICISDI or NICIEXT, or periodically as configure by the NICISDI Sync Period (see Section 3.1).

In the case of a tree merge, add the name of the new SD key server's name to this list after trees are merged, and reboot all the servers in the tree unless periodic synchronization is enabled. The final list must contain the names of SD key servers in all trees. It is strongly recommended that NICI version 2.0.1 or newer is installed on servers.

4.2.2 NDSPKI:SD Key List

This attribute is reserved for future use to hold the list of security domain key identifiers. A future version of NICI will use this attribute.

4.3 Key Synchronization

NICISDI or NICIEXT can be configured to periodically synchronize its keys with each SD key server. This feature is disabled by default. See Section 3 to setup.

The sync period value can be updated while the server is up, and the server does not have to reboot for the change to take effect. The new period value takes effect in the next scheduled synchronization time. Setting this value to zero or removing it entirely causes the termination of the background thread at the next scheduled execution. Thus, further changes of this value to a nonzero value would have no effect unless the server reboots.

Starting with NICI 2.4.0, NICI creates a domain key automatically on a server with WRITE rights to the domain's object in the *Security.KAP* container. It is designed to support multiple domains created in the *Security.KAP* container. At present, there is only one domain represented by *W0* in the *Security.KAP* container.

4.4 INITSDI.NLM

This obsolete NLM was provided with Cobra release to create or to retrieve a tree key (the only security domain key at the time) during installation. This NLM can be used to create and retrieve a tree key as a stand-alone utility.

To create a new tree key on the local box, run

```
INITSDI –new logFile errorFile serverName
```

To retrieve the tree key from a server in the same tree, run

```
INITSDI –get logFile errorFile serverName treeName
```

For instance, load

```
INITSDI –get sys:\sdi.log sys:\sdi.err server.novell tree
```

to receive a key from server *server.novell* in *tree*.

In order to create or retrieve a tree key, the security domain key file *NICISDI.KEY* must be deleted. The NICISDI.KEY file, regardless of the platform/OS, is server unique, and should not be copied from one machine to another. Copying it would not make the key available. A manual new key creation typically causes more problems by introducing a new key on the server it is run different from the actual tree key other servers have. It is strongly recommended not to use this NLM, and let NICI 2.4+ manage such keys.

The INITSDI.NLM is obsolete with NICI 2.4 as it provides auto-sync and auto-create capabilities. It may not work if the target server has NICI versions 2.0.1 and newer.

5 Error Resolution

5.1 Error -1460: NICI_E_NOT_FOUND

If returned when trying to initialize NICI on a Windows platform, that typically means that NICI is not installed, or the NICI device (in device driver versions, 1.x) is not running. If NICI device is not running, you can try to run it by typing "net start niciccs" on a Windows NT/2000 console. If it fails, reboot the system. Otherwise, reinstall NICI.

This error is returned when a security domain key (e.g., a tree key) key is not found on the system. The API is *CCS_GetPartitionKey*. See the NICISDI section for more.

5.2 Error -1470: NICI_E_FIPS140CNRG_ERR

This is an error in the NICI's internal random number generator as defined by FIPS 140. NICI will try to recover from this error, and returns this error if it can't. Solution would be to retry, reload, or restart the application. We don't anticipate this error to occur.

5.3 Error -1471: NICI_E_SELF_VERIFICATION

This error condition is introduced with the FIPS 140 certified NICI, and is present regardless of the certification level of NICI on non-NetWare platforms. Upon loading or being instantiated by a process, NICI runs a set of tests for module integrity as well as cryptographic process integrity. In the failure of one of these tests, NICI puts itself in an inoperable state and returns this error. Typical cause of this problem is module verification failure. Solution is to reinstall NICI, or perhaps remove and then install NICI.

5.4 Error -1472: NICI_E_CRYPTO_DOWNGRADE

This error is introduced in NICI version 2.0.1. The most likely cause is installation of weak NICI version on a strong NICI installed base. Solution is to install strong NICI.

Novell started shipping the strong NICI worldwide, and stopped shipping the import restricted version with limited key sizes. We don't anticipate seeing this error anymore.

5.5 Error -1494: NICI_E_NOT_INITIALIZED

Similar to error -1497, this is typically caused by the lack of NICI license materials or configuration files. Reinstalling NICI typically solves the problem. If it does not, first try removing the NICI registry key on Microsoft Windows, deleting the UNIX configuration file */etc/nici.cfg*, and then installing NICI. Reinstalling NICI does not remove existing keys. If this doesn't solve the problem and you don't lose data by deleting the NICI configuration files and keys, then delete the NICI configuration directory together with the registry on Microsoft Windows or UNIX configuration file, and reinstall NICI.

5.6 Error -1497: CCS_E_AUTHENTICATION_FAILURE

Typical causes:

1. Lack of NICI licensing materials (.NFK file copied to NICIFK file). NICI on servers (NetWare, Dhost or equivalent environment on other platforms) must have a NICI foundation key file in order to initialize key materials. NICI license materials are part of an eDirectory license. Earlier NetWare installs had the option of installing directory without licenses that basically disabled NICI. With the new directory services introduced with Tao for the first time, DS is using NICI for a variety of cryptographic functionality. So, a simple upgrade from an earlier version of DS to a newer version would render DS unusable due to NICI. This is not a bug, per se, but an installation deficiency. NICI will not operate without a NICI licensing materials, or a proper configuration file. Solution is to install a license (can be the installation of the same license), or copy the .NFK file from the license diskette to NICIFK file (Section 3.1), and then reboot the server or restart the DHost process.
2. Lack of or corrupted NICI configuration files. Especially on NetWare servers. A corrupted NICI configuration file is not fixable; it is thrown away. An effort was made to minimize this problem starting with NICI version 1.3.x. It is less likely for this to occur especially with NICI 2.x+.
3. Cryptography module downgrade. See Section 5.4.

5.7 NICI Module Corruption (NetWare): Abend

All NICI modules are signed NLMs on NetWare, and they have the XLM extension. These modules are loaded by XIM.XLM, which is in turn loaded by XLDR.XLM as part of SERVER.EXE execution. The XIM module verifies multiple digital signatures during XLM loading. NetWare abends if any of the signatures is invalid. This is intentional, and not a problem or a bug. It makes sure that the cryptographic and key management modules are not tampered with, and that the module integrity is in place. We have seen corrupted XLMs due to CD burner and other copying problems.

Note that the NICI license materials file (NICIFK) is also signed. An invalid license file renders NICI dysfunctional.

5.8 Error –670 Error creating/fetching Security Domain key

This error is not unique to Dove but was first reported during Dove (version 8.6.0) upgrade testing, probably because servers are not rebooted during the Dove upgrade but DS is restarted. The problem is duplicated in other environments by restarting DS (without rebooting and allowing NICI to reinitialize) on servers listed in the W0 object.

Workarounds:

1. Avoid restarting DS on the servers listed in the W0 object without also initializing NICI.

2. Restart the server identified by the W0 object before requesting the security domain key. (A restart will allow NICI to reinitialize, but you still have to be careful not to restart DS.)
3. Upgrade to NICI version 2.4+.

6 Install and Upgrade

NICI has a platform dependent install program. Reinstalling NICI does not destroy existing keys. Except on NetWare, NICI 2.0+ install does not require rebooting the box in most instances. However, if the NICI module (DLL or .so) is in use and can't be overwritten by the install program, a reboot may be necessary.

6.1 Version Upgrade and Compatibility

Installing a newer version of NICI on top of an existing NICI installation upgrades NICI. Always upgrade NICI using its install program. Freely copying NICI modules often result in a chaotic system, and consequences of such an action often cause irreparable damage to the system including PKI, SecretStore/Single-Sign-On, NMAS, DS, and other products.

Applications developed for NICI 1.x are not compatible with newer NICI versions (2.x+). To provide backward compatibility, NICI 1.x on Microsoft Windows platforms can be simultaneously installed with newer NICI versions. If this is desired, always install the newer version *after* the old version of NICI; e.g., install NICI 2.4 after NICI 1.5.7.

6.2 NICI Transfer (NUWNICI)

As part of NetWare upgrade wizard utility, NICI team provided an NLM (NUWNICI.NLM) to encrypt and transfer NICI configuration files from one physical box to another. The encrypted files are written to a floppy diskette, and the floppy is physically transported to the target box. The NUWNICI.NLM can also be used as a stand-alone NICI transfer utility. It has multiple phases. The first phase (Phase 1) is executed on the target (new) server, phase 2 is executed on the source (old) server, phase 3 is executed on the target (new) server. After phase 3 is completed, the target (new) server must be rebooted for the transfer to take effect.

There also is a phase 4 executed on the target (new) server. Phase 4 basically is a handy tool to check if NICI is working properly. This is useful to check the state of NICI on the new server after the reboot.

A help screen is displayed by using the –h command line option.

On non-NetWare platforms, copying the NICI configuration files to the new box transfers NICI keys and keying materials to the new server.

In such an event, it is highly recommended to delete the NICI configuration on the old server.

6.3 Microsoft Windows NT/2000: chkdsk

NICI versions 1.x are implemented as a kernel driver on Microsoft Windows systems. Due to an improper registry configuration, NICICCS.SYS kernel driver on Windows NT/2000 systems may prevent a check disk (chkdsk) running during system reboot (initial blue screen). Alternatively, the system may try to run chkdsk on every system reboot. NICI version 1.5.5 install fixed this problem. However, you can also check the Windows NT/200 registry to make sure that your system does not have this problem. Check "*HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NICICCS*" key's "*Start*" DWORD value. It must be set to 2 to prevent "chkdsk" volume access errors.

6.4 NetWare 5.x and 6Pack Install Issues

The NICISDI.XLM module shipped as part of NICI 2.x+ does a better job of authenticating and checking rights among other enhancements in conjunction with eDirectory. Together with the directory services' rights management changes at install time, some changes were inevitable, and a backward compatibility is broken. This issue comes up when installing a new server with an earlier version of NICI (earlier than 2.x), such as NetWare 5.x server, into a tree with a Security Domain Server (the server listed in the *Security.KAP.W0* container) running NICI 2.x or later. The new server being installed (added) into the existing tree, say a 6Pack tree, fails when trying to connect and get a copy of the tree key. This error occurs during the final file copy and shows up as part of the certificate server installation.

It is important to realize that there will not be a fix as the fix would reduce the overall security.

Workaround: Assume that a NetWare 5.x server is installed into a 6Pack tree.

1. Install the 5.1 server standalone (its own tree)
2. Update to NICI 2.0.1 or later (The one shipped with 6 pack) after the installation is completed. Availability of NICI 2.0.1 on the web is TBD.
3. Uninstall the directory on the NW 5.x server.
4. Delete file sys:system\nici\nicisdi.key
5. Install the NW 5.x server into the 6pack directory tree
6. Create the server certificates via the PKI management console for this server.
7. Configure up LDAP/etc.

6.5 NICI Backup and Restore

A backup and restore of the NICI configuration directory, its subdirectories, and files within is sufficient for backing up NICI configuration on all platforms.

On older versions of NICI before 2.x on NetWare platforms, some of the NICI configuration files are stored in the SYS:_NETWARE directory. A backup utility may or may not include this directory. We recommend an upgrade to NICI 2.0.1 for a more streamlined NICI configuration backup. NICI versions 2.x and newer store NICI configuration files in SYS:\SYSTEM\NICI directory.

6.6 NICI Upgrade to version 2.x on UNIX* Systems

A hybrid version (mixed features of NICI 1.2 and NICI 1.5) of NICI was shipped with the Tao version of eDirectory. In order to migrate the NICI configuration files from the hybrid version to 2.x, an upgrade utility (*runf2dc*) is provided.

If a UNIX (Solaris, Linux, or AIX) server is hosting more than one eDirectory, each eDirectory instance typically has its own NICI directory setup. Both instances of NICI configuration files must be migrated with the provided tool. For instance, let two eDirectory instances run on a single Solaris host in */var/nds1* and */var/nds2* directories, respectively. The *runf2dc* tool must be run on */var/nds1/nici* and */var/nds2/nici* directories to migrate each instance separately.

Materials in the NICI configuration files don't depend on the contents of eDirectory files. On the contrary, encrypted data in eDirectory depend on keys stored in NICI configuration files. Such encrypted data (such as user private keys, certificates, secret store data, and NMAS store data) will not be available if NICI files are not migrated properly.

We strongly recommend running each instance of eDirectory on the same host with different user Ids to separate their cryptographic materials using the host system's security mechanisms. Note that, NICI does not require a special user to run, except for install: a privileged user who can install *setuid* programs must install NICI (a one-time operation).