NeoExec Professional Administrators' Guide

Version 1.0

Liability Notice

Information in this manual may change without notice and does not represent a commitment on the part of NeoValens.

The software described in this manual is provided by NeoValens under a license agreement. The software may only be used in accordance with the terms of the agreement.

NeoValens S.A.
66, Rue de Luxembourg
L-4221 Esch-sur-Alzette
Luxembourg

Email: support@neovalens.com
Web: www.neovalens.com

Published on: January 2004

Contents

Introduction

This document describes how to administer NeoExec Professional by means of the NeoExec Administrative Console (NAC). The NAC is a Windows application that allows you to create and edit NeoExec configuration files. A configuration file contains the list of applications that you wish your users to run with elevated privileges. Such applications are known as *privileged applications*. The configuration file needs to be deployed to each workstation where the NeoExec kernel driver will parse it and apply the policies contained therein

When an end user launches a privileged application NeoExec modifies the process token on the fly adding the local 'Administrators' group, thereby allowing the user to run the application as if he/she was a member of such group. The only difference between a NeoExec privileged application and applications run in the context of a regular administrator (a user member of the local Administrators group) account is represented by the Operating System privileges available. NeoExec privileged applications run with the same privileges of the logged on user. Therefore some applications that require additional privileges, such as, for example the Time & Date applet, may not run. Such privileges (*Change the System time* in this example) can however be enabled on a per-user or per-group basis by using the Group Policy MMC snap-in. Please refer to Appendix B for further details.

Privileged Applications Vs Privileged Accounts

Members of the local Administrators group have privileges that allow them to perform any action on a computer. Users are often made members of the Administrators group because some applications require elevated privileges to run. The problem is twofold: users often abuse of those privileges to install new applications and/or to modify the configuration of their computer and, possibly even more important, users with elevated privileges are more vulnerable to viruses and trojans. Most malware requires elevated privileges to be installed and to replicate, and members of the local Administrators group are the primary target.

The principle of *least privilege* states that users should be granted the most restrictive set of privileges needed for the performance of authorized tasks. Application of this principle limits the damage that can result from accident, error, or unauthorized use of an information system (IS).

NeoExec helps in applying the least privilege principle by restricting elevated privileges to selected applications.

Getting Started

The NAC is accessible via a shortcut from the Start menu under *Programs -> NeoExec Professional -> Administrative Console*. Upon launching the NAC you will be presented with the NAC main screen as shown below.

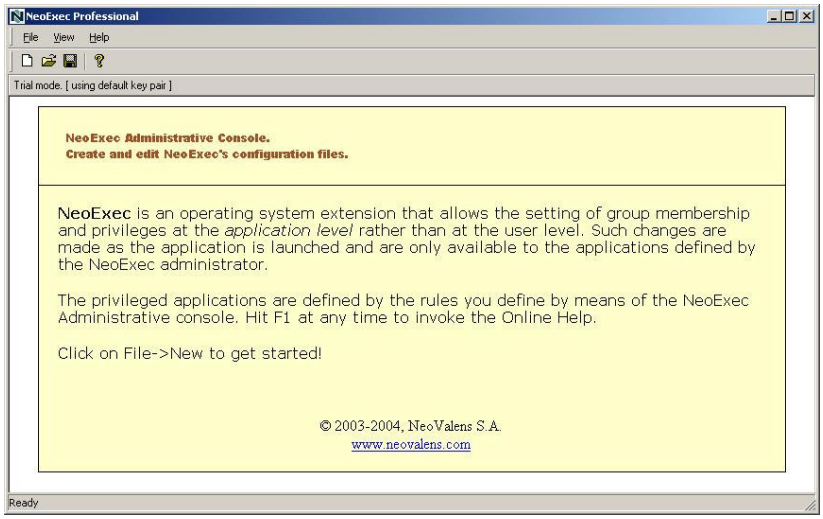

Figure 1: NeoExec Administrative Console

To create a new configuration file click on *File -> New* or *File -> Open* to open an existing one. Clicking on File -> New will create a new configuration file for which you need to specify which are the privileged applications. To add privileged applications to the configuration file one needs only to click on the *Add Privileged Application* button to invoke the Rule editor dialog as shown below in Figure 2.

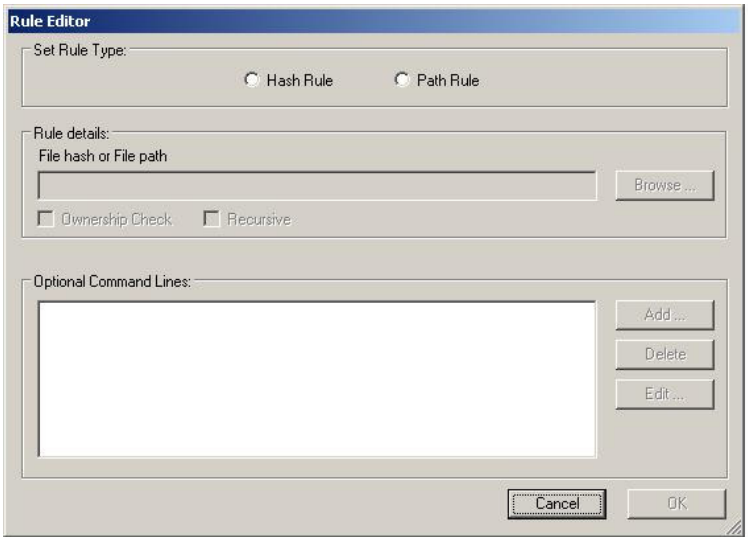

Figure 2: Creating a new rule

NeoExec Professional supports two kinds of rules: Path Rules and Hash Rules. Path Rules rely upon file name and path while Hash Rules rely on the fingerprint of the executable file. NeoExec uses the SHA-1 message digest algorithm to create the executable file fingerprint.

Creating a Path Rule

A Path rule identifies an executable by its name or one or more executables by means of a wild char specifier. The following are example of path rules:

Rule example	Rule target
C:\Windows\regedit.exe	for regedit.exe
C:\Windows\System32\mmc.exe	for mmc.exe
C:\Program Files*.exe	for all executables found in the C:\Program Files directory.
%SystemRoot%\System32\net.exe	The rule targets is the net.exe application. This rule does not depend on the system root name (WINNT, WINDOWS..)
%SystemDrive%mydir\myApp.exe	This rule targets the application named myapp.exe located under the mydir directory on the system drive. The system drive is the drive where Windows is installed.
%ProgramFiles%\Internet Explorer\iexplore.exe	This (sample!) rule allows you to execute Internet Explorer. This rule uses %ProgramFiles% and it will work on all versions of Windows 2000 and Windows XP.

To create a Path Rule you must first click on the Path Rule option button. Clicking on the Path Rule option will enable the File Path edit box and the Browse button.

If you are targeting a single file, and the file location on both your computer and the target computer is the same, it is recommended to use the *Browse* button. In all other cases it is better to enter the File Path by hand.

> **Note**: when targeting multiple files in a directory you must append the appropriate wild char to the directory specifier. For example, C:\test*.exe is a valid path while c:\test is not.

Keywords

> **%SystemDrive%** Use it in place of the system drive (usually C:).
>
> **%SystemRoot%** Use it in place of a hard-coded system root such as c:\winnt or c:\windows.
>
> **%Program Files%** Use it when targeting the Program Files directory.

The keywords are case-sensitive.

Network Drives

When targeting Network drives you need to specify the file location by using *server**sharename**directory* rather than using mapped network drives.

Ownership Check

The Ownership Check option, when selected, instructs the NeoExec kernel driver to check who is the owner of the executable file. Only files owned by the local Administrators group will be trusted. The Ownership Check option is especially important when end users have *write access* to directories where privileged applications reside.

Please note that the Ownership Check can only be used on NTFS formatted local drives. The Ownership Check cannot be used on network shares.

> Beware that files created by the user Administrator will be owned by the Administrator (the user) and will fail the ownership check. The files must be owned by the Administrators group. The same applies to any other member of the local Administrators group.

Recursive

The Recursive option, when checked, instructs the NeoExec Driver to apply the path rule recursively.

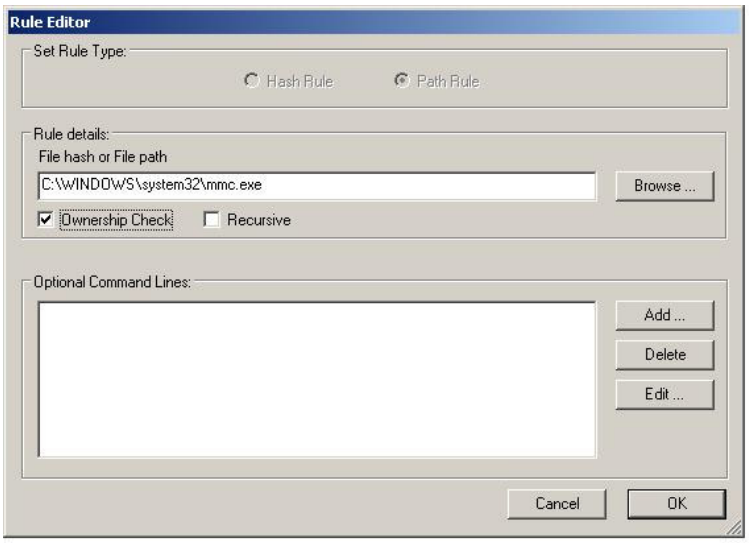

Figure 3: Path rule example for mmc.exe

Creating a Hash Rule

Creating a Hash Rule is a simple two step process. You need first to select the Hash Rule option and then click on the Browse button to select an executable file. Figure 4 shows the end result.

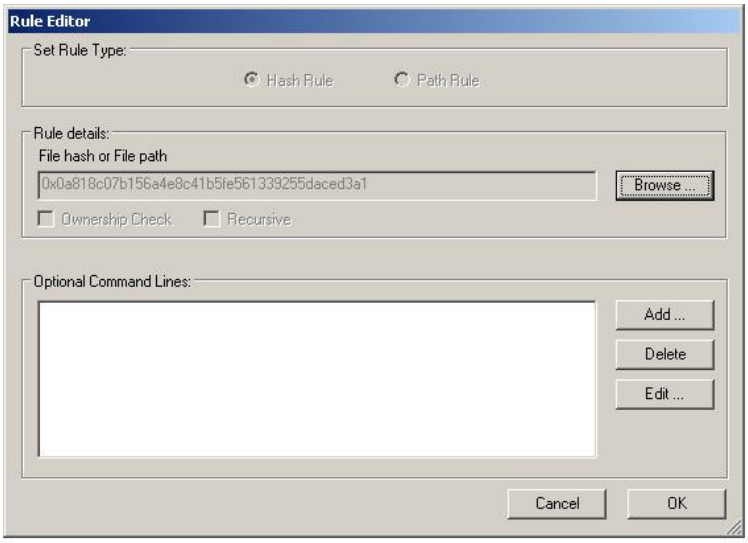

Figure 4: Hash Rule

Command Lines

You may want to restrict the execution of certain privileged applications to so they cam only launched with a specific command line argument. The most obvious examples are container applications that allow you to load and/or launch other modules. For example, all Control Panel applets are either .cpl extensions run by means of rundll32 or shortcuts to administrative MMC snap-ins run by means of mmc.exe. It would be unsafe to grant end users unrestricted access to those applications. A much safer approach is to limit the execution of such applications to run only when invoked with particular command line arguments.

The following two tables(*) list the Control Panel applets and MMC snap-ins found on Windows 2000, Windows XP and Windows 2003.

Administrative action	Control Panel applet
Accessibility Options	access.cpl
802.11 Monitor	apgui.cpl
Add/Remove Programs	appwiz.cpl
Console	console.cpl [1]
Display	DESK.cpl
SCSI, PCMCIA, and Tape Devices	DEVAPPS.cpl [1]
Add New Hardware Wizard	hdwwiz.cpl
Internet	inetcpl.cpl
Regional Settings	INTL.cpl
Game Controllers	joy.cpl
Mouse, Font, Keyboard, Printers	main.cpl
Multimedia and Sounds	MMSYS.cpl
Modems	MODEM.cpl [1]
Network	ncpa.cpl
Logon Management for XP	nusrmgr.cpl (XP)
ODBC	odbccp32.cpl
Power Options	powercfg.cpl
Ports	PORTS.cpl [1]
Devices, Services, Server	srvmgr.cpl [1]
System	SYSDM.cpl
Telephony	telephon.cpl
Date/Time	TIMEDATE.cpl
UPS	ups.cpl [1]

[1] Uses the Windows Server 2003 family of operating systems

Administrative action	Microsoft Management Console file
Current user certificates	certmgr.msc
Certificate authority	certsrv.msc [1]
Certificate templates	certtmpl.msc [1]
Indexing service	ciadv.msc
Computer management	compmgmt.msc
Group policy object editor	dcpol.msc [1]
Device manager	devmgmt.msc
Disk defragmenter	dfrg.msc
Distributed file system	dfsgui.msc [1]
Disk management	diskmgmt.msc
Active directory domains and trust	domain.msc [1]
Default domain security settings	dompol.msc [1]
Active directory users and computers	dsa.msc [1]
Active directory sites and services	dssite.msc [1]
Event viewer	eventvwr.msc
File server	filesvr.msc [1]
Shared folders	fsmgmt.msc
Group policy object editor	gpedit.msc
Internet authentication service	ias.msc [1]
Local users and groups	lusrmgr.msc
Removable storage	ntmsmgr.msc
Removable storage operator requests	ntmsoprq.msc
Performance	perfmon.msc
Routing and remote access	rrasmgmt.msc [1]
Resultant set of policy	rsop.msc
Local security settings	secpol.msc
Services	services.msc
Telephony	tapimgmt.msc [1]
Terminal services configuration and connections	tscc.msc [1]
Remote desktops	tsmmc.msc [1]
Windows management infrastructure	wmimgmt.msc

[1] Uses the Windows Server 2003 family of operating systems

(*) Table source: Microsoft Developer network (MSDN)

The NAC makes it extremely easy to identify the command line required. Once the Path or Hash rule have been defined all you need to do is start the privileged application and then click on the *Add* button located at the top-right corner of the *Optional Command Lines* group box. Clicking on on *Add* will invoke the Command Lines dialog as shown in Figure 5.

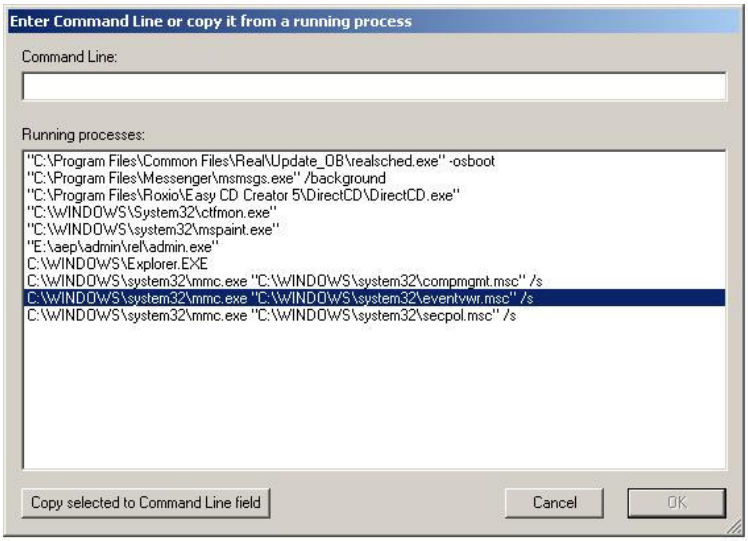

Figure 5: Command Line Example

Double-click on any command line shown in the Running Processes list box to copy it to the Command Line field. Alternatively, you can also select the command line by clicking on the *Copy selected Command Line field* button. Simply repeat the process for any command line argument you wish to use. Figure 6 shows an example of a complete rule.

Application	**C:\WINDOWS\system32\mmc.exe**
Rule type	path
Rule value	?oC:\WINDOWS\system32\mmc.exe
Command Lines	3
Command Line1	C:\WINDOWS\system32\mmc.exe "C:\WINDOWS\system32\compmgmt.msc" /s
Command Line2	C:\WINDOWS\system32\mmc.exe "C:\WINDOWS\system32\eventvwr.msc" /s
Command Line3	C:\WINDOWS\system32\mmc.exe "C:\WINDOWS\system32\secpol.msc" /s

Edit Delete

Figure 6: Path Rule example

Editing a rule

To edit a rule click on the *Edit* button located to the right-side of the rule you want to edit. Clicking on the *Edit* button will invoke the Rule Editor for the selected rule.

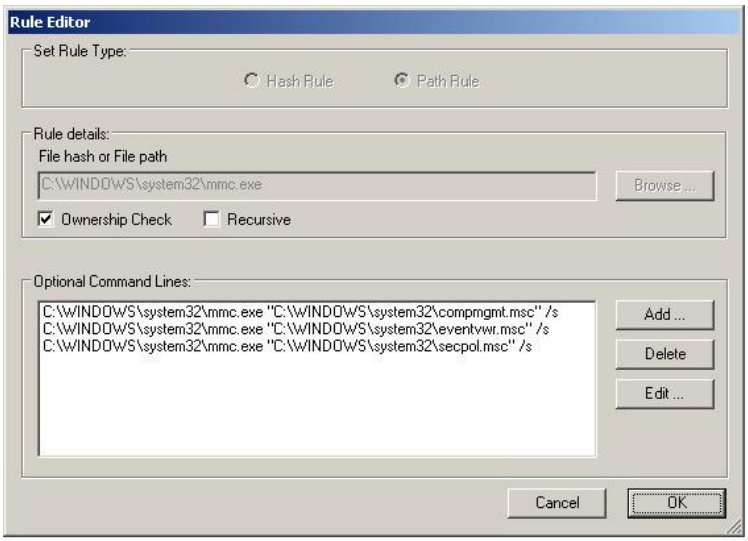

Figure 7: Editing a Path Rule

For a Path Rule, you can change all attributes but the path itself. For a Hash Rule, you can only add or remove command lines.

Deleting a rule

To delete a rule click on the Delete button located to the right side of the rule you want to delete.

Deploying the configuration file

To deploy the configuration, copy the file named **neoexec.cfg** to the *\SystemRoot\System32\Neo* directory of those computers running the NeoExec Professional kernel driver. The NeoExec kernel driver monitors the directory every thirty seconds and will automatically pick up any new configuration file.

By default, only members of the Administrators group can copy files to that location.

Licensing

NeoExec Professional can be tested for evaluation purposes for a maximum of thirty (30) days after which a license must be purchased or you must uninstall the product. When you purchase a license you will receive a license file and a private key that you will use to sign your configuration files. Your public key is embedded in the license file. By default, when operating in Trial mode, the NAC will sign the configuration file with the default private key.

License file

The License File is always named **neoexec.lic** and should be copied to the *\SystemRoot\System32\Neo* directory of all computers, even the ones you will run the NAC from. The license file contains information about the licensee and the number of licensed clients.

Private key

The private key is used by the NAC to sign the neoexec.cfg file. You should store the private key file, always named **ne-private.key**, in a safe place. The NAC will scan for such file in the following locations:

1. The NAC current directory

2. Any Floppy drive

3. Any Removable drive (such as a SUB memory stick)

4. Any CD-ROM drive

The NAC will show which key pair is being used (Trial mode Vs client one) in the NAC main window header.

Signing configuration files created during the evaluation period

To convert any configuration file created during the trial period all you need to do is to open the file(s) from the NAC and save them back. Once saved, you will need to deploy the newly signed configuration file, along with the license file, to all target computers.

Appendix A: Events logged to the System event log

The NeoExec Professional kernel driver logs a number of events to the System event log. The following table describes the events and the conditions that trigger them.

ID	Type	Message	Condition
28672	Error	Unsupported Operating System detected. NeoExec Professional supports only Windows 2000 and Windows XP.	This message is logged when NeoExec Professional detects an Operating System other than Windows 2000 or Windows XP.
28673	Informational	NeoExec Professional loaded	The driver was loaded and started successfully.
28675	Warning	NeoExec Professional is in trial mode due to the lack of a valid license. NeoExec Professional will be disabled after 30 days unless an appropriate license is obtained. In the latter case, no reboot is required	Could not find license file (neoexec.lic) or the license is not valid.
28676	Error	NeoExec Professional is in Trial Mode and has been disabled due to the lack of a valid license. To enable NeoExec, replace the current license (if any) with an appropriate license. No reboot is required.	This message is given once the trial period has elapsed. NeoExec will no longer function until a valid license file is provided.
28677	Error	NeoExec Professional has loaded but is disabled due to a license violation. To enable NeoExec, replace the current license with an appropriate license. No reboot is required.	The number of computers running NeoExec Professional is greater than the number licensed. Please contact NeoValens to obtain a new license.
28678	Informational	NeoExec Professional license check OK.	The license file was found and verified successfully.
28679	Warning	No public key file has been found: NeoExec will use the default one.	This error should never occur. Please contact NeoValens if this error is ever logged.
28680	Error	Cannot read public key file.	The public key was found but an error occurred while reading it.
28681	Error	Invalid signature detected in file neoexec.cfg.	The signature was found but is invalid. The configuration file will be ignored.
28682	Error	No signature detected in file neoexec.cfg.	The configuration file does not appear to be signed at all and will be ignored.
28683	Warning	NeoExec config file has changed.	A new configuration file has been copied to the Neo directory.
28684	Warning	No configuration file found. Save neoexec.cfg under SystemRoot\System32\Neo.	This event is typically logged when you start NeoExec Professional for the first time. This message will no longer be logged once you supply a configuration file.

Appendix B: Privileges

Operating System privileges and logon rights are referred to as *User Rights*. User rights are assigned by using the Group Policy MMC snap-in. After you have started MMC and opened the Group Policy snap-in, use the console tree pane to ocate the User Rights Assignment folder. It is located under Local Computer Policy/Computer Configuration/Windows Settings/Security Settings/Local Policies. The following list shows the privileges that you can assign to a user by setting user rights.

- Act as part of the operating system
- Add workstations to a domain
- Back up files and directories
- Bypass traverse checking
- Change the system time
- Create a token object
- Create permanent shared objects
- Create a pagefile
- Debug programs
- Enable trusted for delegation on user and computer accounts
- Force shutdown from a remote system
- Generate security audits
- Increase quotas
- Increase scheduling priority
- Load and unload device drivers
- Lock pages in memory
- Manage auditing and security log
- Modify firmware environment values
- Profile a single process
- Profile system performance
- Replace a process-level token
- Restore files and directories
- Shut down the system
- Take ownership of files or other objects
- Unlock a laptop

Source: Microsoft Developer Network (MSDN)

By default, members of the local Users group have the following privileges:

Privilege	Description	Status
SeChangeNotifyPrivilege	Bypass traverse checking	enabled
SeShutdownPrivilege	Shut down the system	disabled
SeUndockPrivilege	Remove computer from docking station	enabled

Disabled privileges can be enabled programmatically.

Members of the local Administrators group have the following privileges:

Privilege	Description	Status
SeChangeNotifyPrivilege	Bypass traverse checking	enabled
SeSecurityPrivilege	Manage auditing and security log	disabled
SeBackupPrivilege	Back up files and directories	disabled
SeRestorePrivilege	Restore files and directories	disabled
SeSystemtimePrivilege	Change the system time	disabled
SeShutdownPrivilege	Shut down the system	disabled
SeRemoteShutdownPrivilege	Force shutdown from a remote system	disabled
SeTakeOwnershipPrivilege	Take ownership of files or other objects	disabled
SeDebugPrivilege	Debug programs	disabled
SeSystemEnvironmentPrivilege	Modify firmware environment values	disabled
SeSystemProfilePrivilege	Profile system performance	disabled
SeProfileSingleProcessPrivilege	Profile single process	disabled
SeIncreaseBasePriorityPrivilege	Increase scheduling priority	disabled
SeLoadDriverPrivilege	Load and unload device drivers	enabled
SeCreatePagefilePrivilege	Create a pagefile	disabled
SeIncreaseQuotaPrivilege	Adjust memory quotas for a process	disabled
SeUndockPrivilege	Remove computer from docking station	enabled
SeManageVolumePrivilege (*)	Perform volume maintenance tasks	disabled

(*) Not available under Windows 2000 Professional

Applications that require privileges beyond those granted to the members of the Users group usually enable them on the fly and display an error message should that fail. If an error arise, grant the required privilege(s) to the user by means of the Group Policy MMC snap-in as described above.

Appendix C: Security considerations

Synthetic SID

NeoExec adds two SIDs to the token of each instance of privileged applications: the local Administrators group (S-1-5-32-544) and a synthetic one (S-1-21-101010101-21030440) used primarily to track privileged applications. In this document we will refer to S-1-21-101010101-21030440 as the NE-SID.

The NE-SID should be used to prevent users executing privileged applications from accessing certain areas of the file system or registry. Access to these areas is governed by ACLs and one needs only to add, where necessary a deny ACE or an audit ACE.

For example, the NeoExec Professional kernel driver setup adds a deny ACE for the NE-SID on the *\SystemRoot\System32\Neo* directory to prevent end users from replacing the neoexec.cfg file.

The setup also adds an audit ACE for the Everyone user to the *\SystemRoot\System32\Neo* directory as well as a deny ACE for NE-SID to the MACHINE\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Windows registry key in order to prevent the user from injecting code into Privileged Applications by means of AppInit_DLLs.

Code Injection

There still exist a number of techniques for injecting code into a running process that an end user could use to execute a portion of unauthorized code at elevated privileges. The fact that privileged applications are vulnerable to code injection does not make NeoExec less useful or less secure as, without it, end users would be running all processes with elevated privileges.

Tampering

The neoexec.cfg file is digitally signed against tampering. The NeoExec driver will verify the digital signature and will log to the System event log any abuse attempt.

NTFS Permissions

Where Path rules are used you should ensure that regular users cannot write to privileged applications. Failing to protect privileged applications could result in end users substituting them with other applications. Privileged applications can be protected by removing the write permission on the executables.

Appendix D: Glossary of Terms

access control entry
(ACE) An entry in an access control list (ACL). An ACE contains a set of access rights and a security identifier (SID) that identifies a trustee for whom the rights are allowed, denied, or audited.

access control list
(ACL) A list of security protections that applies to an object. (An object can be a file, process, event, or anything else having a security descriptor.) An entry in an access control list (ACL) is an access control entry (ACE). There are two types of access control list, discretionary and system.

access token
An access token contains the security information for a logon session. The system creates an access token when a user logs on, and every process executed on behalf of the user has a copy of the token. The token identifies the user, the user's groups, and the user's privileges. The system uses the token to control access to securable objects and to control the ability of the user to perform various system-related operations on the local computer. There are two kinds of access token, primary and impersonation.

privilege
The right of a user to perform various system-related operations, such as shutting down the system, loading device drivers, or changing the system time. A user's access token contains a list of the privileges held by either the user or the user's groups.

process
The security context under which an application runs. Typically, the security context is associated with a user, so all applications running under a given process take on the permissions and privileges of the owning user.

security context
The security attributes or rules that are currently in effect. For example, the current user logged on to the computer or the personal identification number entered by the smart card user. For SSPI, a security context is an opaque data structure that contains security data relevant to a connection, such as a session key or an indication of the duration of the session.

security descriptor
A structure and associated data that contains the security information for a securable object. A security descriptor identifies the object's owner and primary group. It can also contain a DACL that controls access to the object, and a SACL that controls the logging of attempts to access the object.

security identifier
(SID) A data structure of variable length that identifies user, group, and computer accounts. Every account on a network is issued a unique SID when the account is first created. Internal processes in Windows refer to an account's SID rather than the account's user or group name.

Source: Microsoft Developer Network (MSDN)