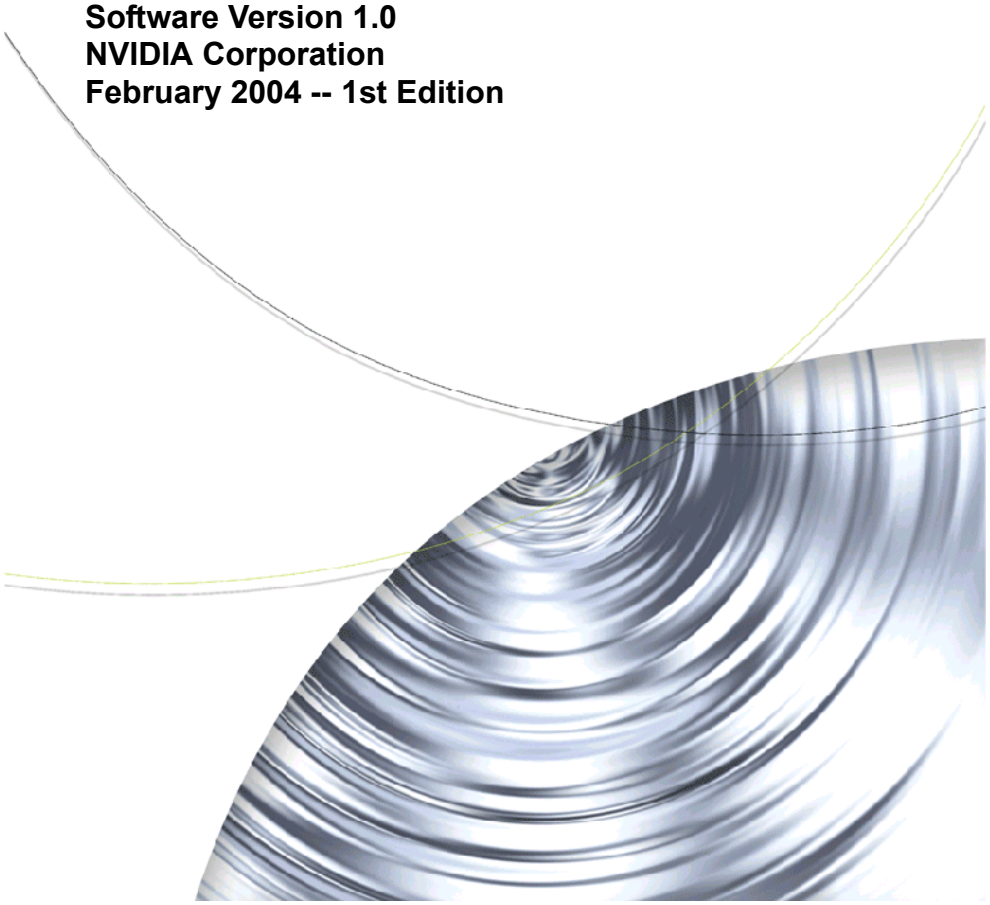
NVIDIA ForceWare Networking and Firewall Administrator's Guide

Software Version 1.0
NVIDIA Corporation
February 2004 -- 1st Edition



Published by
NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050

Table of Contents

7. Using The Command Line Interface (CLI)

A. Ethernet Parameters Reference

B. NVIDIA Firewall Parameters Reference

C. Glossary

List of Tables

List of Figures

CHAPTER 1

INTRODUCTION

This chapter contains the following major sections:

Audience

This guide is intended for the system or network Administrator of an organization as a guide to install and use the NVIDIA® ForceWare™ Network Access Manager application.

Note: This guide assumes the reader has Administrator access privileges. Exceptions are noted, where applicable.

About NVIDIA ForceWare Network Access Manager

Using the ForceWare Network Access Manager application, you can easily configure and control NVIDIA networking hardware and software, gather statistics, and monitor logs. ForceWare Network Access Manager gives you several choices in managing your networking hardware and software:

Command Line Interface (CLI)

The ForceWare Network Access Manager provides command line access through the **nCLI** program. The nCLI command can be run in either **expert** or **interactive** mode to configure and monitor NVIDIA networking components.

- **Expert mode** is suitable for deployment in an organization by running nCLI from a login script. To use nCLI in expert mode, you need to be familiar with the syntax and characteristics of configuration parameters.

  For details and examples of using the nCLI command with various Ethernet and NVIDIA Firewall parameters, see "Ethernet Parameters Reference" on page 70 and "NVIDIA Firewall Parameters Reference" on page 108.

- **Interactive mode** runs in a shell environment and is suitable for Administrators who do not have access to the syntax and characteristics of the **nCLI** configuration parameters. nCLI provides navigation feature to assist these users.

Note: Extensive nCLI usage samples in batch file format are provided in the following *subdirectories* under the *default* path of **c:\nvidia\NetworkAccessManager**, or your user-specified path:

samples\Eth (for Ethernet)

samples\Firewall (for Firewall)

You can cut and paste the appropriate command and use them in batch files or in command lines.

Also see "Using The Command Line Interface (CLI)" on page 56.

Web-based Interface

The ForceWare Network Access Manager Web-based interface offers convenient access through several features:

- **Wizards —** see "Using the Firewall Wizards Page" on page 32.
- **Profiles**
- **Status summaries**
- **Help**. Context-sensitive online Help is available on a wide range of features. From any ForceWare Network Access Manager Web page, click the **Help** tab, as shown in Figure 1.1, to access detailed Help on the parameters you are configuring.
- **Tool tips**. When your cursor hovers over a parameter name, its description is displayed in a popup text window, called a *tool tip*.

Sample Web Pages

Figure 1.1 ForceWare Network Access Manager — Home Page

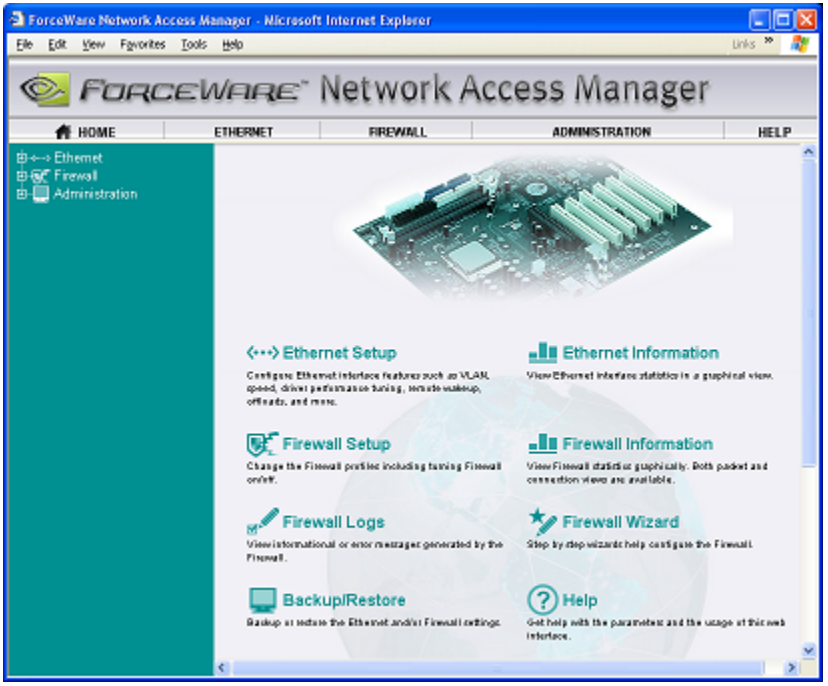


Figure 1.2 Ethernet Basic Configuration

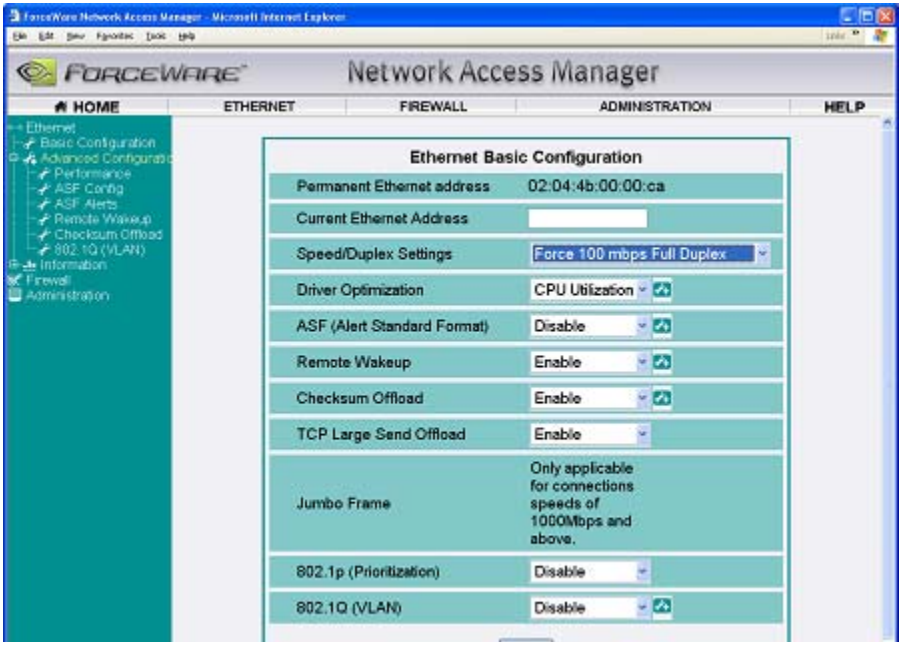
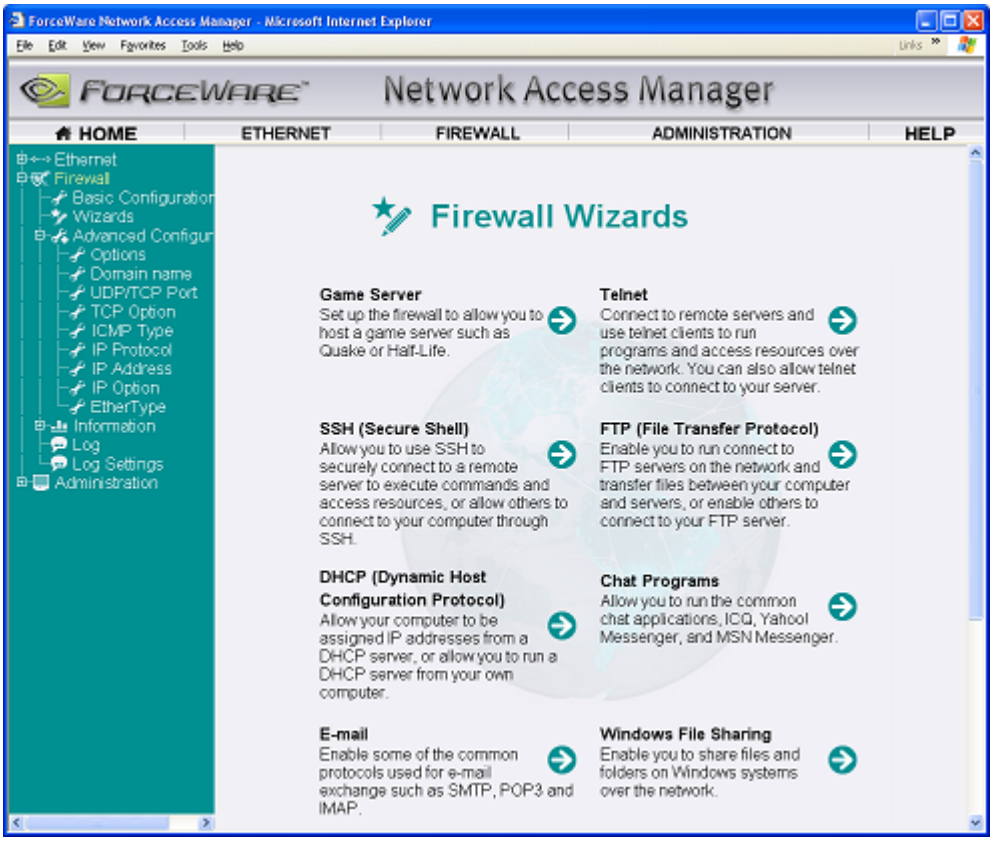


Figure 1.3 Firewall Wizards



WMI Script

You can use the Microsoft® **Windows Management Instrumentation (WMI)** script language to manage NVIDIA networking hardware and software.

Using WMI script language is recommended *only* for Administrators who are already familiar with programming in WMI script and who have become familiar with the syntax and characteristics of configuration parameters.

WMI script programming is being used by the IT staff of larger corporations to carry out day to day maintenance work. Overall benefits of using WMI scripts include:

- **Industry standard** — WMI can be implemented using languages such as VBScript and JScript.
- **Ease of use**

- **Common scripts** — allow access to ForceWare Network Access Manager data.
- **Flexibility** — The WMI script user can utilize the power of the script languages to meet almost any requirements. For example, as an Administrator, you can write a WMI script to scan for Yahoo Messenger on a computer and open the appropriate port if the computer user has sufficient rights.
- **Remote use** — means you can run the WMI script language remotely and use it as a deployment tool in an organization. See "Configuration Deployment" on page 21.

For further informations, see "Using WMI Script" on page 52.

About Security

Access control is based on the type of application, the type of user (Administrator or not), and the type of access — i.e, local or remote.

The application access control allows you to configure non-Administrator access to the applications, which includes, nCLI, the command line interface, the WMI scripting interface, and the local and remote Web interfaces.

For applications that are accessed from the local computer, the application access rights depend on the current access rights for the Windows login session.

Note: A non-Administrator user on a computer cannot access the firewall parameters and modify the access control parameters.

For further details on security and access control, see "Application Access Control Page" on page 46.

NVIDIA Firewall

NVIDIA Firewall — the only "native" firewall in the market — is optimized and integrated into the NVIDIA nForce systems that support ForceWare Network Access Manager. (See Table 1.2 for supported NVIDIA hardware and features.)

The NVIDIA Firewall is a high performance, "hardware-optimized" firewall offering enhanced reliability and protection at the end-point — i.e., desktop.

The **NVIDIA Firewall** incorporates firewall and anti-hacking technologies under **NVIDIA Personal Firewall** and **NVIDIA Professional Firewall** names. The NVIDIA Professional Firewall has all the features of NVIDIA Personal Firewall plus anti-hacking features such as anti-IP-spoofing, anti-sniffing, anti-

ARP-cache-poisoning, and anti-DHCP server, which are important security controls for corporate network environments.

For an explanation of firewall concepts and the NVIDIA Personal Firewall, see Chapter 3 — "NVIDIA Firewall: Basic Concepts" on page 23.

Key Features: NVIDIA Personal and Professional Firewall

- **Comprehensive "packet filtering" —** see "NVIDIA Firewall: Basic Concepts" on page 23.
- **"Stateful" and "stateless" packet inspections —** see "NVIDIA Firewall: Basic Concepts" on page 23.
- **Predefined security profiles —** see "Configuring the NVIDIA Firewall" on page 30**.**
  - Lockdown, High, Medium, Low, and Off
  - User-customizable profiles
  - Internet Protocol version 6 (IPv6) support
- **Advanced management features —** see "Configuring the NVIDIA Firewall" on page 30.
  - Remote administration
  - Monitoring
  - Configuration
- **User-friendly Web-based interface** includes Wizards, charts, tables, and logging statistics. See "Configuring the NVIDIA Firewall" on page 30
- **Submitted for ICSA certification**

Anti-Hacking Features: NVIDIA Professional Firewall *only*

NVIDIA Professional Firewall contains all of the above **NVIDIA Personal Firewall** features and, additionally, includes the following anti-hacking features, which provide important security controls for corporate network environments.

Note: To access and configure the NVIDIA Professional Firewall anti-hacking features, you need to be using an nForce3 250 Professional computer. See Table 1.2, "Hardware and Software Features Support" on page 16 for details.

- **Anti-IP spoofing**
- **Anti-sniffing**
- **Anti-ARP cache poisoning**
- **Anti-DHCP (Dynamic Host Configuration Protocol) server process**

For further information, see "Configuring Anti-Hacking Features — For NVIDIA Professional Firewall Users" on page 35.

Summary of NVIDIA Firewall Features

Table 1.1 summarizes the NVIDIA Firewall features.

Table 1.1 NVIDIA Firewall — Personal vs. Professional Features

NVIDIA Firewall Feature	Available in NVIDIA Personal Firewall	Available in NVIDIA Professional Firewall
Configuration Wizard	Yes	Yes
Pre-defined Security Profiles	Yes	Yes
Web-based Management	Yes	Yes
Remote Administration	Yes	Yes
Stateful Inspection	Yes	Yes
Sateless Inspection	Yes	Yes
Anti-IP Spoofing	**No**	Yes
Anti-Sniffing	**No**	Yes
Anti-ARP Cache Poisoning	**No**	Yes
Anti-DHCP server process	**No**	Yes

System Requirements

General Requirements

- **WMI (Windows Management Instrumentation) service**

  Note: WMI service is not automatically started on Windows 2000. The ForceWare Network Installer needs to change this service to run automatically on Windows startup.

- **WMI MOF compiler (MOFCOMP)** must be available on your computer.
- **NTFS file system**. It is recommended that you install the ForceWare Network Access Manager application on an NTFS file system so that

sensitive information such as Firewall or access configuration data are protected from being changed by a user without Administrator access rights.

Note: For further information on NTFS, refer to Windows online Help.

Hardware Requirements

Support of ForceWare Network Access Manager features on NVIDIA nForce series hardware (personal computer) is outlined in Table 1.2.

Note: Miscellaneous features that are not listed (e.g., checksum offload, segmentation offloads, etc.) are supported by all four nForce platforms listed in Table 1.2.

Table 1.2 Hardware and Software Features Support

NVIDIA Software Supported	NVIDIA Hardware (Personal Computer)			
	nForce2 Gigabit MCP	**nForce3 250 Gigabit**	**nForce3 Ultra**	**nForce3 250 Professional**
NVIDIA Firewall	Personal *only*	Personal *only*	Personal *only*	Professional
ForceWare Network Access Manager — Web-based Interface	Yes	Yes	Yes	Yes
ForceWare Network Access Manager — CLI and WMI Script support	No	No	No	Yes
VLAN, IEEE 802.1Q	No	No	No	Yes
Alert Standard Format (ASF)	No	No	No	Yes

Operating Systems

The ForceWare Network Access Manager application supports the following Microsoft operating systems:

- Windows XP Professional
- Windows 2000

Software, Memory, and Disk Space Requirements

Note: All figures in Table 1.3 are estimates based on default settings and a standard operating environment.

Table 1.3 Software, Memory, and Disk Space Requirements

Software	Memory	Disk space	Disk space for non-English languages
nForce Ethernet driver for Windows XP/2000 **Note:** To run the ForceWare Network Access Manager software, nForce Ethernet must be configured as a bridge device in the BIOS, which is the factory default.	1 MB	100 KB	Approximately 1.5 MB per language
NVIDIA Firewall	4 MB	200 KB	
ForceWare Network Access Manager	5 MB	25 MB	

For further information on driver installation, see "Installing and Uninstalling the NVIDIA Graphics Driver Software" on page 30.

NVIDIA Firewall and Ethernet Parameters Reference

Appendix A: "Ethernet Parameters Reference" on page 70 and Appendix B: "NVIDIA Firewall Parameters Reference" on page 108 provide detailed parameters reference and usage information.

You can also obtain context-sensitive Help when using parameters by clicking the **Help** tab from any ForceWare Network Access Manager Web page.

CHAPTER 2

INSTALLATION GUIDELINES

This chapter contains the following main topics:

Overview of NVIDIA ForceWare Network Installation

The ForceWare Network Access Manager software supports the silent installation method, which means no user interaction is needed to install the software. The silent installation process uses a response (`.iss`) file that contains information similar to what you would enter as responses to dialog boxes when running a normal setup.

Below is a summary of the silent installation steps:

1. See "Before Using the ForceWare Network Installer" on page 19.
2. Download/obtain the NVIDIA nForce Driver installer. See "Locating the ForceWare Network Installer" on page 19.
3. Run NVIDIA nForce Driver installer which will uncompress files.
4. Specify where to copy the nForce files.
5. Read the section "Locating the ForceWare Network Installer" on page 19.

6 Generate the response file, as explained in "Creating the Response File" on page 20.

7 At the target computer, install in silent mode using the three installation files and the response file. See "Running Installation in Silent Mode" on page 20

Locating the ForceWare Network Installer

The ForceWare Network Installer (**setup.exe**) and the Network Access Manager software are part of the basic NVIDIA nForce driver installation package, which can usually be obtained from the NVIDIA Web site or a partner OEM. The nForce driver installer program uncompresses and saves this software (listed below) in a user-specified directory:

- **setup.exe** (located in <*uncompressed directory_name*>\Ethernet\NRM)
- **data1.cab**
- **NVIDIA ForceWare Network Manager.msi**

Before Using the ForceWare Network Installer

Before you run the ForceWare Network Installer (**setup.exe**) software, please note the following:

- The **nForce Ethernet driver software** must already be installed and operational on your computer.
- You must have **Administrator access rights** to do the following:
  - Run the Setup installation program *and*
  - Uninstall and/or modify the ForceWare Network Access Manager software, when needed.
- If you are using the ForceWare **Network Access Manager Web-based interface**, note the following:
  - Microsoft Internet Explorer version 5 *or higher* must be running on your computer.
  - The ForceWare Network Access Manager Web-based interface uses the NVIDIA registered TCP port 3476.

  Note: In the event that some other application on the computer is using this registered port, change the port number for that application.

Running the ForceWare Network Installer

Creating the Response File

From the directory where the setup.exe program is located, run the following command and go through the installation dialog boxes as normal, selecting the options that will be used in subsequent silent installs. All choices are recorded in the response file (`nvidia_net.iss`).

```
setup.exe -a -r -f1 "c:\nvidia_net.iss"
```

Note: You can change the path and name of the response file by replacing **`c:\nvidia_net.iss`** with a drive letter and name of your choice.

Running Installation in Silent Mode

```
setup.exe -s -a -s -f1 "c:\nvidia_net.iss"
```

Launching the ForceWare Network Access Manager Web Interface

1. Using the instructions in the previous sections of this chapter, make sure you have used the **`setup.exe`** installation program to install the ForceWare Network Access Manager software.
2. To launch the ForceWare Network Access Manager Web-based interface, you can do one of the following:
   - From your Windows desktop, double-click the ForceWare **Network Access Manager** icon

     or

   - From your Windows task bar, click **Start > Programs > NVIDIA Corporation > Network Access Manager > Web-based Interface**

Configuration Deployment

Configuration deployment means configuring multiple computers to use the same configuration through an "automated" procedure. There are several ways to achieve this:

- Run the `nCLI` command to change parameters during the login script.
- You can choose to run `nCLI` to configure one parameter at a time or use the `import` command for bulk configuration.

  Note: Sample command line access scripts can be found in the `sample` directory, under the default path of **`c:\nvidia\NetworkAccess Manager`**, or your user-specified path. See "Using The Command Line Interface (CLI)" on page 56 section for more information.
- Create and run WMI scripts to change parameter during login script execution.

  Notes:

  - WMI script usage samples are provided in the following subdirectories:

    `samples\Eth`

    `samples\Firewall`

    under the default path of **`c:\nvidia\NetworkAccess Manager`**, or your user-specified path.
  - You can cut and paste the appropriate command and use them in a batch file or the command line. For further details, see "Using WMI Script" on page 52.
  - To use WMI scripting, you must be familiar with the syntax and characteristics of configuration parameters. See the "Ethernet Parameters Reference" on page 70 and "NVIDIA Firewall Parameters Reference" on page 108 for details.
  - For additional details, refer to the Microsoft documentation on WMI scripting.

Note: Many Ethernet parameters require restarting the network driver for script changes to take effect. When the network driver is restarted, network connections will terminate, which will terminate the login script. To get around the problem, you can utilize the `NV_DriverRestartFlag` to defer restarting the driver. Keep in mind that a driver restart is still required for script changes to take effect.

- If you are using nCLI, run the following command:

```
ncli set NV_DriverRestartFlag.RestartFlag DeferRestart
```

- If you are using WMI scripting, run the following command:

```
//Set NV_DriverRestartCmd ""
try
{
   var NV_DriverRestartCmd = GetObject("winmgmts:root/
nvidia/NS_Eth").Get("NV_DriverRestartCmd=@");
   NV_DriverRestartCmd.RestartCmd=1;
   NV_DriverRestartCmd.Put_();
   WScript.Echo("Success in Group Set");
}
catch(Exception)
{
   WScript.Echo("Error in Group Set: ",
Exception.description);
   WScript.Quit(1);
```

CHAPTER 3

NVIDIA FIREWALL: BASIC CONCEPTS

This chapter contains the following main topics:

Types of Firewalls

The **NVIDIA Firewall** is a type of firewall that is typically referred to as a "PC firewall" or a "desktop firewall." Another classification of firewalls is the "gateway firewall."

The main difference between the PC firewall and the gateway firewall is that while the gateway firewall monitors network traffic and controls access between two different networks or administrative domains, the PC firewall controls traffic generated or received by a single computer.

Therefore, a gateway firewall is usually a dedicated computer, or a part of a network switch or router, with multiple interfaces through which certain traffic is allowed and other traffic is blocked.

A **PC firewall** is usually software that is installed on the personal computer, or a combination of software and hardware that is integrated to the computer. In both types of firewalls, certain traffic is allowed and certain traffic is blocked according to the specific rules configured for the firewall.

Firewalls just discussed can be further classified as one of two types —

- **Application layer**
- **Packet-based** firewalls are of two main sub-types:
  - **Stateful**
  - **Stateless**

Note: The **NVIDIA Firewall** is a "**packet-based PC firewall**" with both "**stateful**" and "**stateless**" features.

Stateful vs. Stateless

Stateful and **stateless** are adjectives that describe whether a computer or computer program is designed to note and remember one or more preceding events in a given sequence of interactions with a user, another computer or program, a device, or other outside element. **Stateful** and **stateless** are derived from the usage of *state* as a set of conditions at a moment in time.

Stateful means the computer or program keeps track of the state of interaction, usually by setting values in a storage field designated for that purpose.

Stateless means there is no record of previous interactions and each interaction request has to be handled based entirely on information that comes with it.

Inbound vs. Outbound Packets

Network traffic is not inherently safe nor dangerous. In addition to the usual attributes of packets that can distinguish them from each other, such as IP addresses and TCP port numbers, one criterion that can be used to help discriminate traffic is the direction in which that traffic is flowing.

For traffic arriving from the outside the PC, it is reasonable to presume that there is a chance that an attack may be present, whereas in traffic originated by the PC, it is less likely to be dangerous. The firewall rules consider the direction of traffic as an attribute when establishing the traffic that should be allowed (i.e., such traffic is deemed to be safe or to have an acceptable level of risk) versus the traffic that should be denied (i.e., such traffic is deemed to be unsafe).

Note: The tolerance for risk will vary among users, so there is no universally accepted definition of "dangerous" packets. However, the default configuration of the NVIDIA firewall represents industry-accepted best practices, and can be used as the basis for customized configurations that more closely match the end-user's specific requirements.

By defining the direction as part of the specification of a rule, the end-user can separate traffic that he/she considers to be safe enough from traffic considered unsafe. Most protocols exchange traffic bi-directionally; therefore, the "direction" of such exchanges is defined by the connection-initiation packet. For example, in the case of TCP packet, the first packet matching a new set of IP addresses and TCP ports for which the TCP `SYN` flag is set establishes the direction of that subsequent bi-directional flow.

Other protocols, such as **User Datagram Protocol (UDP)** or **Internet Control Message Protocol (ICMP)**, may not have the equivalent of the TCP "`SYN`" flag. Therefore, for those protocols, the NVIDIA Firewall uses the direction of the first packet matching a given set of IP addresses and, for example, UDP ports, as the direction for the subsequent bi-directional flow.

About the TCP Protocol

Some network protocols, such as **TCP**, require an explicit connection initialization process. Firewall rules that apply to TCP typically depend partially on the direction of the connection establishment. When referring to protocols that involve establishment of a connection:

- **Inbound** describes a connection attempt not originated by the local computer.
- **Outbound** describes a connection attempt that was originated by the local computer.

About the UDP and ICMP Protocols

Unlike TCP, other protocols, such as **UDP** and **ICMP** do not have an explicit connection establishment process. A computer can use protocols such as UDP and ICMP to send data packets to any other computer at any time, but the receiving computer, or an intervening firewall, can reject or accept the data on a per-packet basis.

UDP

UDP is frequently used in a connection-like manner, but without the connection establishment process. In other words, UDP-based applications may rely on long-term computer-to-computer sessions. However, the meaning of the "direction of the connection" in the UDP context is broader than in the TCP context.

- The direction of a packet is **inbound** if the initial packet matching this new set of IP and UDP header field values was a *received* packet.

- Similarly, a UDP connection is considered to be **outbound** if the initial packet matching this new set of IP and UDP header field values was a *transmitted* packet.

Thus, firewall rules that apply to UDP typically also depend on the direction of the first packet of a new "connection." UDP packets, like TCP packets, can be matched against a "connection table" by performing a hash function on certain fields in the packet to determine if there is a match in a table of hash values where there is at least one connection that corresponds to each hash value.

ICMP

ICMP is an example of a protocol with neither a connection establishment process nor any connection-like functionality.

Firewall rules relevant to these types of protocols are applied to every packet, and **inbound** and **outbound** respectively refer to packets (of one of these protocols) that are received and transmitted across any of the network interfaces that the firewall is protecting.

Stateful Filtering

Stateful filtering (also known as stateful inspection or dynamic packet filtering) provides enhanced security by monitoring network packets over the period of the connection for that particular traffic. Because stateful filtering can dynamically track each connection, compare packets against the connection's expected state, and drop the packets that don't conform to the protocol, it has replaced static filtering as the industry standard firewall solution for networks.

It is also the case that stateful filtering scales much better than stateless filtering because the firewall policy table is only consulted once per connection, instead of once per packet. This means that as the number of rules grows, the stateful firewall will use a lower percentage of CPU, because in a stateless design, each packet will have to be compared against half of the firewall rules, on average, until a matching rule is found that explicitly allows or denies the packet. However, an increase in the size of the firewall policy rule table does not impact the stateful firewall to such a large degree, since the majority of packets are not connection setup packets.

A stateful firewall amortizes the CPU cycles that were used to do the firewall policy rule table lookup over the massive per-packet CPU savings due to having only a simple per-packet hash to compute, to determine if the current packet is associated with a previously allowed connection.

In contrast, a stateless firewall must examine every packet against the complete firewall policy rule table, or until it finds a matching rule, so in essence, every

packet is treated as a connection setup packet, incurring the associated processing penalty.

As a result of the differences in processing required for stateful vs. stateless firewall lookups, latency due to stateful firewall operations is very small and nearly constant on a per-packet basis, whereas latency in a stateless firewall depends on the size of the firewall policy rule table, and is of a much larger magnitude.

Once a TCP or UDP connection is established, a stateful firewall ensures that data traffic for that connection can flow in either direction — even if the rules governing the firewall limit such traffic to be only associated with remotely generated (i.e., inbound), or locally-originated (i.e., outbound) connections.

When a stateful firewall has determined that a connection is being established by decoding each packet, it checks its policy table to find out whether the connection is allowed or denied.

- In TCP, the connection establishment packet is a specially marked TCP packet that the firewall can detect.
- A UDP connection is initiated by the first packet matching a set of identifying fields in the IP and UDP headers.

If the firewall allows the new connection, the firewall saves a set of five values related to that connection's establishment into its connection-tracking table during the lifetime of that connection.

Every inbound and every outbound packet associated with a given connection contains the same five values. This allows the stateful firewall to quickly check whether or not the packet belongs to a connection that was previously granted permission and then deny or allow the packet accordingly.

Note: Only TCP packets that match the connection-tracking table are allowed. UDP packets that do not match the table may represent a new "connection" and are compared with the firewall rules in order to determine whether or not to add an entry to the connection-tracking table for this new connection.

The five "connection identifying" values saved into the connection-tracking table are:

- **IP Source Address**
- **IP Destination Address**
- **IP Protocol**
- **TCP or UDP Source Port**
- **TCP or UDP Destination Port**

For TCP, in addition to the five items in the list, the firewall tracks the state of the TCP connection (for example, the current stage of the connection establishment process) in order to enforce legal state transitions in the TCP protocol.

The firewall also tracks the current TCP "sequence and acknowledgement" numbers and the most recent TCP window in order to determine whether to drop packets that fall outside the current valid TCP window. This kind of scrutiny prevents potential attackers from sending spurious TCP "reset" packets to the local computer in that the firewall prevents these reset packets to reach the host if the TCP sequence number of the reset packet falls outside the current valid TCP receiving window.

Some TCP options can also be used by the stateful firewall in determining whether to allow or deny TCP packets because certain TCP options can only be used if their use was negotiated during the connection establishment process. If such TCP options were negotiated during the connection establishment phase, then the TCP state will reflect the successfully negotiated TCP options for that connection. The TCP policy table can still override the peers and prevent certain TCP options from being negotiated at all.

Note: Other TCP options are not pre-negotiated. Therefore, decisions about whether to allow or deny TCP packets with such options must be based on the **stateless** (see "Stateless Filtering") configuration of the firewall.

Stateless Filtering

The main difference between **stateful filtering** and **stateless filtering** is that contrary to the quick lookup-and-decide process enabled by the connection state tracking table that drives the decision making process in stateful filtering, all of the stateless filtering rules must be examined in sequence, for each packet, until a rule is found that either explicitly allows or denies that packet.

Note: For protocols such as ICMP and other non-TCP and non-UDP protocols, and for any non-IP protocols, the firewall performs stateless filtering but no stateful tracking or filtering.

In **stateless filtering**, the firewall can be configured to "allow in" or "deny in" certain kinds of traffic (from a specific protocol, with a particular option, etc.) on a given network interface. Similarly, the firewall can be configured to "allow out," "deny out," "allow in and out," or "deny in and out" on the same traffic. Note that "in" implies the receive direction and "out" implies the transmit direction.

On average, the firewall will need to search half of its rules list for any given packet in order to find an applicable rule. Therefore, in general, as the number of rules increases, the firewall consumes more time in determining the outcome

of a given packet. On the other hand, the NVIDIA Firewall has been optimized so that looking up certain commonly used parameters (for example, ICMP, TCP, and UDP in the IP protocol table) is much faster and independent of the table size.

The firewall can be configured to perform stateless filtering based on:

- EtherType values
- Specific IPv4 or IPv6 addresses or address prefixes
- Specific domain names contained within DNS name resolution queries or responses
- Specific IP options
- Specific TCP options
- Specific ICMP (Type, Code) pairs
- Other relevant parameters

In all cases, stateless filtering rules are specified in the appropriate firewall table in the ForceWare Network Access Manager Web-based interface.

For example, when filtering ICMP traffic, the filtering rule is based on both the first three items (IP Source Address, IP Destination Address, and IP Protocol) as listed in the section on "Stateful Filtering" on page 26, as well as the particular ICMP (Type, Code) field values in each ICMP packet.

In ICMP filtering, the IP Protocol is implicitly required to have a value of "0x01," which is the protocol value for ICMPv4. A similar requirement is placed on ICMPv6, with its own unique identifying number in the IPv6 headers (i.e., 0x3A).

In most situations involving stateless filtering, it is necessary to allow a given protocol to go both in and out on a given interface in order for the associated application to operate normally. However, it may also be the case that certain applications require that one type of traffic be allowed in, while another type is allowed out.

One example of the latter case is "ping" because in order for the application process to complete successfully, the firewall must be configured to allow *both* an outbound ICMP Echo packet (Type = 0x08, Code = 0x00) and an inbound ICMP Echo Reply packet (Type = 0x00, Code = 0x00). These settings will allow the local PC to "ping" remote computers but will not necessarily allow remote computers to "ping" the local computer because inbound ICMP Echo packets and outbound ICMP Echo Reply packets are not necessarily allowed.

Note: Based on the above values, note that the ICMP (Type, Code) pair values for ICMP Echo and Echo Reply are, in fact, different.

CHAPTER 4

CONFIGURING THE NVIDIA FIREWALL

This chapter contains the following main topics:

NVIDIA Firewall Parameters Reference

Appendix B: "NVIDIA Firewall Parameters Reference" on page 108 is an NVIDIA Firewall Reference guide, categorizing the firewall parameters by group and table names.

When you are using the Firewall parameters from the ForceWare Network Access Manager Web-based interlace, you can easily access online Help by clicking the **Help** tab.

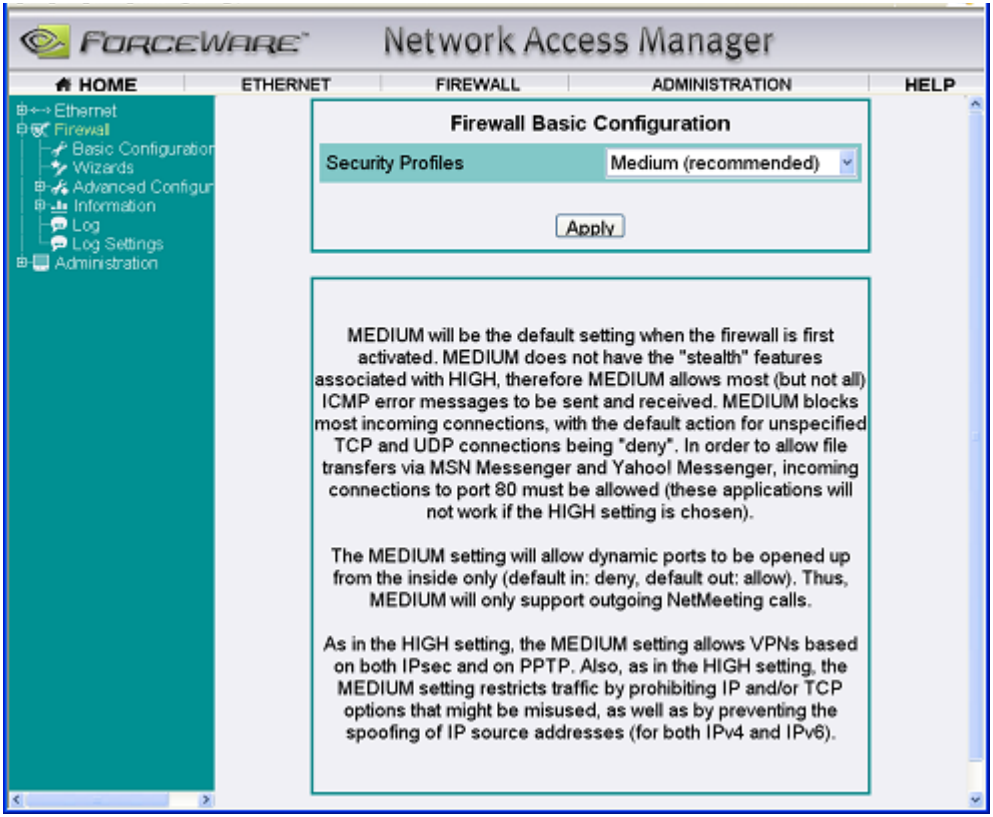
Using the Basic Configuration Page

1. Open the ForceWare Network Access Manager Web-based interface. If you need help, see "Launching the ForceWare Network Access Manager Web Interface" on page 20.
2. From the **Firewall** menu, click the **Basic Configuration** submenu to open the Firewall Basic Configuration page.
3. Click the **Security Profiles** list to view the "profiles" (predefined sets of table rules).

   The following five pre-defined profiles are *not editable*.

   - **Lockdown** – drops all traffic packets, except outbound **Alert Standard Format (ASF)** packets.
   - **High** is meant to be extremely secure, but due to its stringent filtering rules, many applications may not work as expected, or at all.

Figure 4.1 NVIDIA Firewall — Basic Configuration



- **Medium** (the *default* profile *after* installation) is intended to be a good balance between usability and security, with an emphasis on security.
- **Low** is the least secure profile, which allows the most applications to work, but is probably not very secure.
- **Off** turns off the firewall, allowing all traffic.

Note: There are also three other **Custom** profiles that allow you to define the sets of firewall rules, as explained in the "Advanced Configuration" on page 33 section.section.

4 To enable a specific profile, click the **Security profiles** list and select the profile you want.

5 Click **Apply**.

6 To view the actual rules associated with a profile, repeat step 4 above.

7 Open the appropriate table menu under the **Firewall > Advanced Configuration** menu.

This menu lets you see whether the settings are appropriate for your required applications at the desired level of protection.

Note: Unlike the custom profiles, you cannot edit the basic pre-defined profiles.

Using the Firewall Wizards Page

Another way to configure rules in your custom profile is through the **Firewall Wizards** page (Figure 4.2), which is accessible from the **Firewall > Wizards** menu.

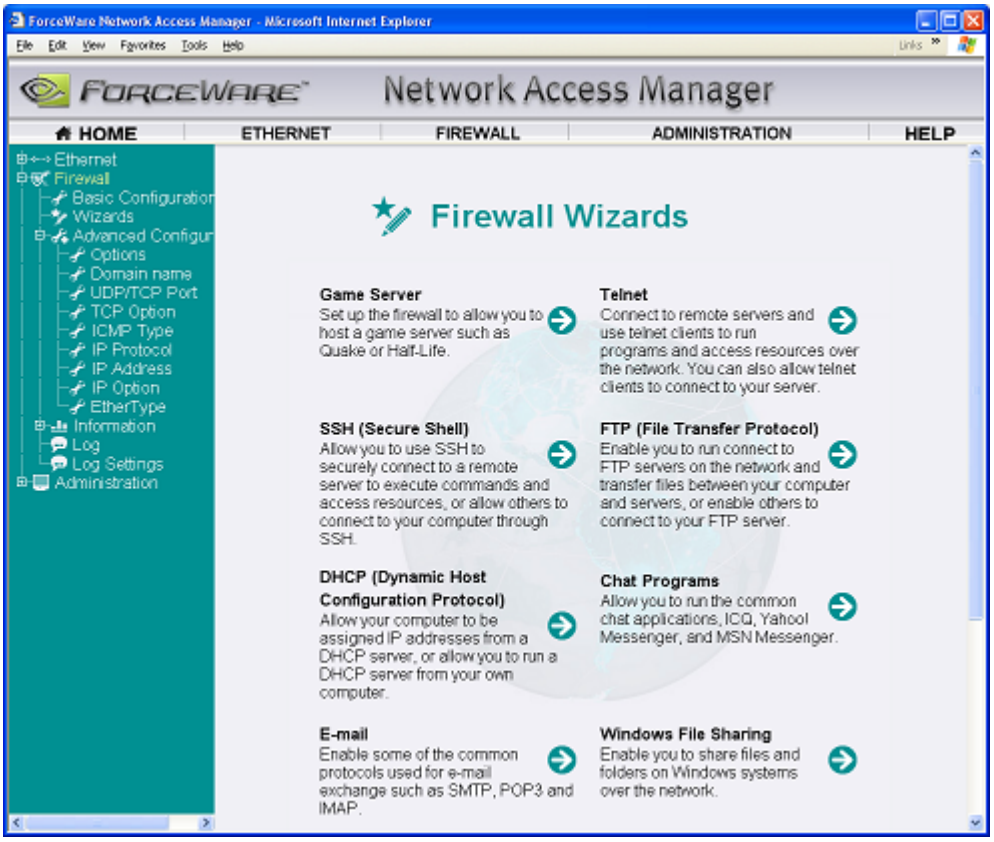
Using a questionnaire format, the wizards provide a simple, step-by-step method to configure the tables and, for convenience, are separated into different categories of commonly-used applications.

The **Firewall Wizards** page allows you to configure the firewall to allow specific applications or classes of applications to work. There are wizards for various types of applications including Telnet, FTP, SSH, game servers, and so on.

These wizards will open the required network ports that are used by these applications. If the particular application you are using needs other non-specific network ports, you can use the Generic Port wizard to add those ports for the application to work.

Note: Refer to your application documentation for information on the TCP/UDP ports that are used, if applicable.

Figure 4.2 Firewall Wizards



Advanced Configuration

If you to generate a customized configuration (profile), any of the basic **Lockdown**, **High**, **Medium**, **Low**, and **Off** profiles discussed in "Using the Basic Configuration Page" on page 31 may be used as a starting point.

Note: Up to three independent custom profiles can be defined.

To create or choose a custom profile:

1 Open the ForceWare **Network Access Manager** Web browser interface.

If you need help, see "Launching the ForceWare Network Access Manager Web Interface" on page 20

2 From the **Firewall** menu, click **Basic Configuration** to open the **Firewall Basic Configuration** page.

3 Click the **Security Profiles** list to view the "profiles" (sets of table rules).

4 Then select on of the three **Custom** profiles.

5 Specify a new name for each custom profile you select in step 4 in the **Rename...** edit box.

Note: You will probably choose to generate a custom profile based on one of the pre-defined profiles, e.g., **Lockdown**, **High**, **Low**, etc.

6 To edit the associated table rules, select the appropriate *table* sub-menu under the **Advanced Configuration** menu to perform any of the following actions:

To add a rule or purge all rules in a table, use the **Add Rule** or **Purge Table** buttons in the corresponding table's page.

To change only the "action" of an existing rule:

a Click the drop-down menu in the corresponding table row under the "action" column and choose either Allow or Deny (for all tables) or Ignore (for the UDP/TCP Port table only). For further details, see "About Working With Tables" on page 36.

b Click **Apply**.

Multiple "actions" may be modified before clicking **Apply**, which accepts all changes at once.

To edit any other parameter of an existing rule, or to delete a rule, click the icon in the corresponding row under the **Edit** column to open the **Rule editing** page.

Note: For brief descriptions of each table parameter, click the **Help** button on the upper-right corner of either the **Table** page or the **Rule editing** page. For more detailed descriptions of each table parameter, refer to "NVIDIA Firewall Parameters Reference" on page 108 in this guide.

The NVIDIA **Firewall** > **Advanced Configuration** page also allows you to toggle the firewall's more advanced security features.

Note: For detailed information on these features, click the **Help** tab on the upper right corner of the page.

Configuring Anti-Hacking Features — For NVIDIA *Professional* Firewall Users

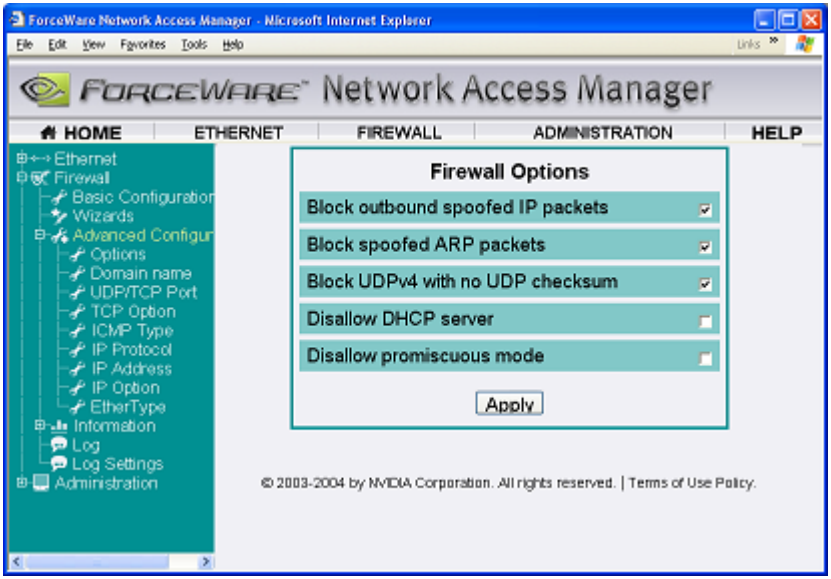
As mentioned in Chapter 1 in the section "NVIDIA Firewall" on page 13, if you are using an nForce3 250 Professional computer, you can access the anti-hacking NVIDIA "Professional" Firewall feature *in addition to* the full range of NVIDIA Personal Firewall features.

The anti-hacking features include anti-IP spoofing, anti-sniffing, anti-ARP cache poisoning, and anti-DHCP server, all of which provide important security controls for corporate network environments.

You can configure anti-hacking features from the NVIDIA **Firewall Options** page (Figure 4.3).

Note: The Firewall **Options** menu choice is available *only* if you are using an **nForce3 250 Professional** computer system.

Figure 4.3 NVIDIA Professional Firewall Options — Configuring Anti-Hacking Features



Follow these steps to access the Firewall Options page:

1. Open the ForceWare Network Access Manager Web-based interface. If you need help, see "Launching the ForceWare Network Access Manager Web Interface" on page 20.
2. From the NVIDA **Firewall** menu, click **Advanced Configuration**, and then click **Options** to open the **Firewall Options** page (Figure 4.3).

3 For detailed information about the options and how to configure them, see "Group: Configure Professional Firewall Options" on page 111 in Appendix B: "NVIDIA Firewall Parameters Reference" on page 108.

About Working With Tables

Specifying Actions

For each rule, you can specify the action that the firewall should perform if a packet or connection matches that rule.

- In the following three types of tables, where the direction of traffic is important, each rule will let you set the **inbound action** and the **outbound action** separately.
  - IP Option table
  - UDP/TCP Port table
  - ICMP table
- In all other types of tables, the direction is not important. Therefore, each rule lets you set one action for both inbound and outbound. Every rule can either **Allow** or **Deny** traffic, while each rule in the UDP/TCP Port table has an additional action called **ignore**.

The **ignore action** is useful when you want a UDP/TCP Port rule to apply in only one direction. For example, setting a rule for HTTP (Web) port 80 to deny inbound and ignore outbound will always block Web connections in the inbound direction, but will let a more generic matching rule or the "default action" to determine the action for outbound Web connections.

About Table Sorting

You can sort any table based on the contents of any column by simply clicking either the **Up** or **Down** arrows adjacent to the column name in the header at the top of each column. When you first view a table, the tables are sorted by default in the following ways:

- In the IP Address table, the Domain Name table, and the UDP/TCP Port table, the rules are normally sorted by the Rule Order column, which is both the order that the rules have been added and the order that they will be applied. Executing the rules in the order of their creation allows you to add overlapping rules that provide one action for a more generic range of IP addresses or domain names, while having a different action for a more specific IP or domain.

For example, if you first create an IP Address table rule to allow address 10.1.1.2 and mask 255.255.255.255, and then create a rule to deny address 10.1.1.0 and mask 255.255.255.0, then the IP Address table will allow traffic to 10.1.1.2 but will block other IP address beginning with 10.1.1.x. Traffic to 10.1.1.2 will not be blocked by the second rule of this table because the first rule already matches it. You can similarly set up the Domain Name table to block a generic domain suffix (e.g., example.com) but allow specific domain names (e.g., foo.example.com).

- In all other tables, the rules are normally sorted by the most significant column. For example, EtherType rules are sorted by the EtherType value, the ICMP rules are sorted by the ICMP Type and then ICMP Code, etc
- An exception to this behavior is that right after adding a rule to any table, the new rule appears at the bottom of the table so that it can be easily verified as having been added. When the table is viewed again (after navigating away to another page within the Web browser), the rules are back to the default sorting method.

Note: While every table has a Rule Order column, only in the IP Address table, the Domain Name table, and the TCP/UDP Port table mentioned above do you need to worry about the Rule Order when adding new rules, because they allow overlapping IP addresses or domain names.

Table "Default Action" Settings

Each table also has an associated "default action," which may be set to **Allow** or **Deny**.

Depending on the nature of the "default action," a given individual rule may or may not have any effect. For example, if the TCP "default action" is to Allow packets associated with outbound connections and to Deny packets associated with inbound connections, then having a rule to allow outbound HTTP (i.e., TCP port 80) connections would be redundant, because that traffic would already have been allowed by the "default action."

The "default action" defines the action that will be performed when no other specific rules in that particular table applies to a given type of packet.

- In general, if the "default action" of a table is to **Deny**, then most rules should be set to **Allow** specific exceptions.
- Similarly, if the "default action" is to **Allow**, then most rules should be to **Deny** specific exceptions.

Note: It is generally agreed that it is safer to discard traffic unless you specifically need to allow it, so a "default action" of Deny is likely to be more secure (or at least more convenient) than a "default action" of Allow. The firewall will compare each packet to the firewall tables in the

following order, from the lower-numbered, more fundamental parameters to the higher-numbered, more complex parameters.

a. EtherType table
b. IP Address table
c. IP Option table
d. IP Protocol table
e. TCP Option table
f. UDP/TCP Port table
g. ICMP table
h. Domain Name table

Note: Packets of a specific protocol, such as TCP, will not be processed by the table of an unrelated protocol, such as ICMP.

Configuration Dependencies

Under certain configurations, the firewall might not function as expected even though its functionality is still consistent with the actual rules that were configured. In particular, it is possible to provide the firewall with conflicting configuration directives, yet it might not be obvious that this is the case. This situation may arise because of the many ways in which traffic can be allowed or denied and the overlapping scopes of the various firewall tables.

For example, suppose that you had configured the firewall to allow certain types of ICMPv4 traffic but had also configured it to block all IPv4 packets. If you had forgotten that the latter was the case, you might wonder why the allowed ICMPv4 traffic was not getting through. In this case, you would have to realize that you cannot expect ICMPv4 traffic to flow unless you allow at least IP Protocol number 0x01 and EtherType 0x0800 for IPv4.

Other less obvious cases are also possible. For example, if all inbound packets with IP options are blocked, then IGMP Reports will not be received by the stack, since all IGMP Reports have an IP Router Alert option included in the packet.

Recommendations

Note: There are many ways to configure different parameters, which could cause *unintended* and *troublesome* consequences.

Therefore, it is best to work step-by-step through a configuration, building up one layer of rules at a time. Once a given configuration is known to be effective, then it is possible to amend the configuration slightly and re-verify the old configuration, while verifying the new configuration as well. Ultimately, the configuration will converge on a set of rules that meets the stated requirements.

Note: Attempting to set up the final configuration in a single big step can sometimes enable interdependencies that prevents things from working as intended and result in difficult troubleshooting.

Firewall Statistics

All packets generate statistics when passing through the firewall, whether they are allowed or denied.

Each packet increments one of these packet counts—UDP, TCP, ICMP, or Other—as well as one of the TCP and UDP connection counts if it is a connection-initiating packet.

The firewall statistics allow you to:

- Get an idea of the kind of traffic your computer is exchanging
- Get an idea of the amount of the traffic being allowed or denied
- Enable verification of whether a recently changed firewall rule is operating as intended

For example, suppose that you wanted to add a rule to deny TCP packets to any port between 1002 and 1009. To do so you can use the ForceWare Network Access Manager Web interface and follow these steps:

1. From the NVIDA **Firewall** > **Information** menu, click any of the graph or table choices.
2. For example, to see statistics about the Firewall interface presented in a graphical format, click **Graphical** to display a page similar to Figure 4.4.

   For detailed Help on options, click the **Help** tab.

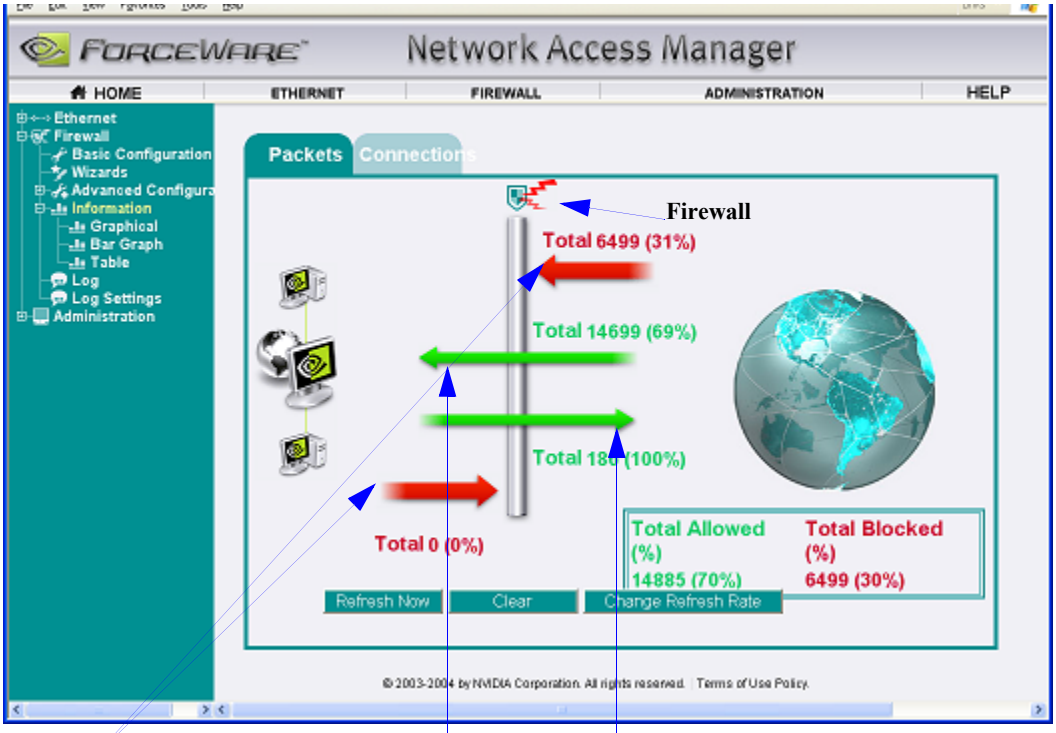
   a. To view statistics based on the number of packets, click the **Packets** tab.

   b. To view statistics based on the number of connections, click the **Connections** tab.

c After noting the current TCP statistics, you can add a **TCP Port Rule** to block the 1002 to 1009 range.

d Then you can send some test packets to verify that such packets were actually blocked.

Figure 4.4 Graphical Information for Packets



Red arrows represent packets or connections that are stopped by the firewall.

Arrows pointing to the computer icon represent received packets or incoming connections.

Arrows originating from the computer icon represent transmit packets or outgoing connections.

3 In order to send TCP traffic to a particular port, you can open a command prompt window and type:

```
telnet foo.example.com 1003
```

where

`foo.example.com` is any valid domain name or IP address that will normally let a packet be sent through the firewall.

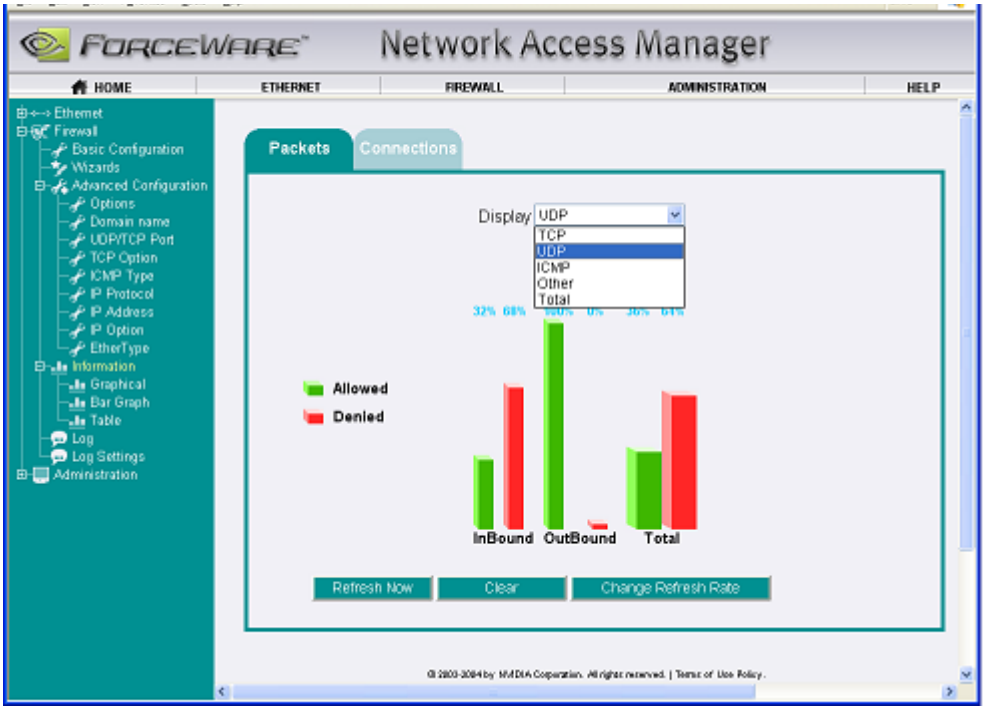
`1003` is actually any number between 1002 and 1009 that should be blocked.

The Telnet program will attempt to connect and the expected result (if the rule has been set up properly) is that the Telnet connection attempt should eventually time out because the packets associated with that connection have been blocked.

4 After performing the above test, you can click the **Bar graph** or **Table** option from the Information menu to verify whether the "Outbound TCP connections denied" count or the "Outbound TCP packets denied" count has increased by an amount consistent with the tests that were performed.

A sample **bar graph** is shown in Figure 4.5.

Figure 4.5 Bar Graph of Packet Activity



A sample **table** of Firewall statistics is shown in Figure 4.6 and Figure 4.7.

Figure 4.6 Table (Statistics) of Packet Activity — 1st section

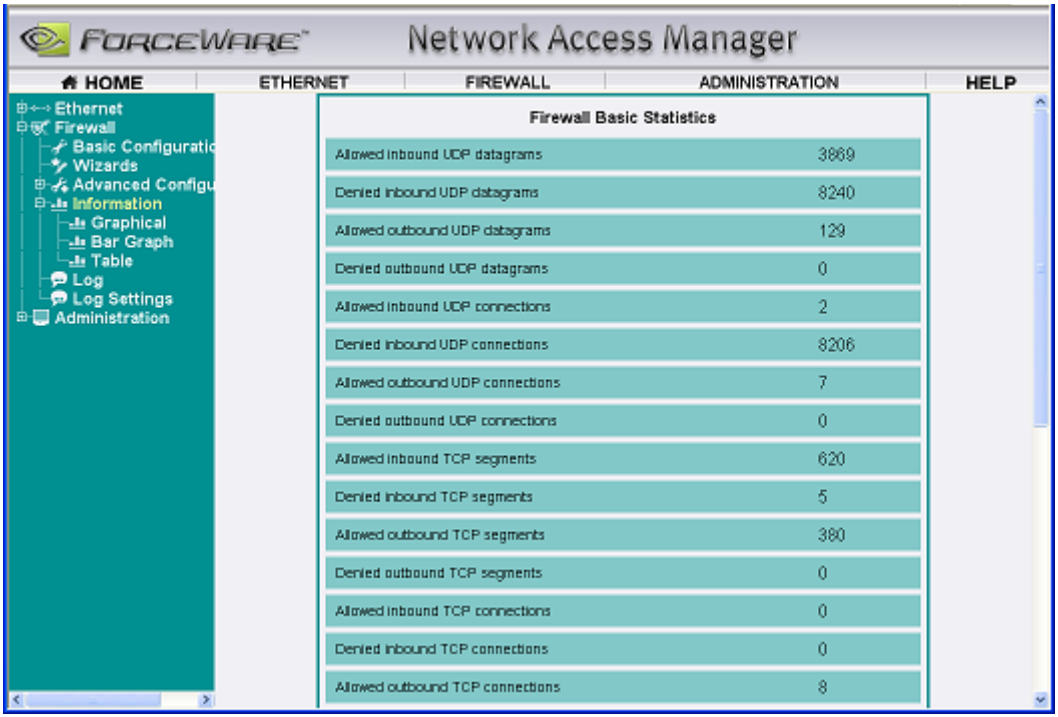
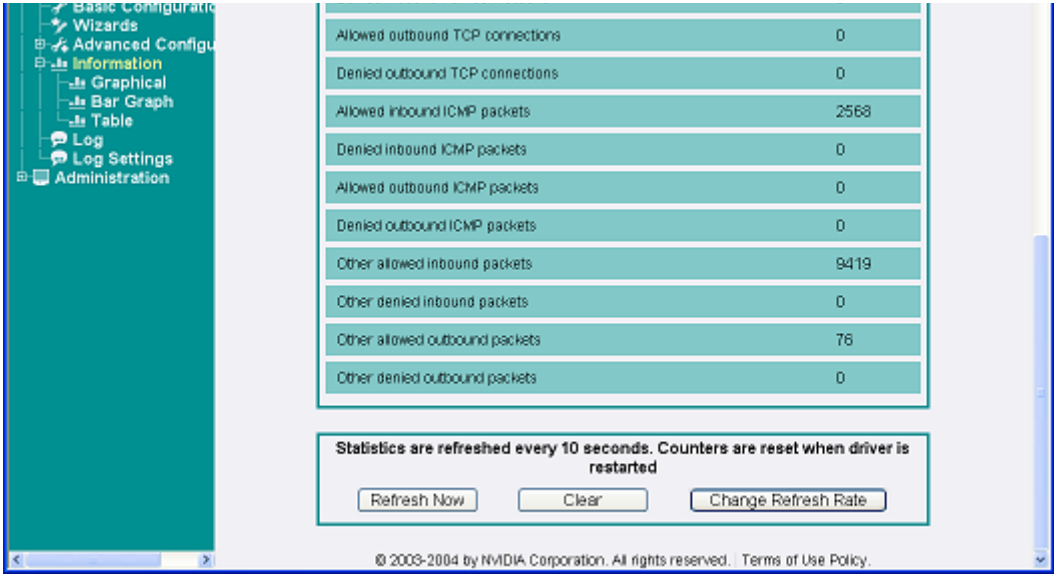


Figure 4.7 Table (Statistics) of Packet Activity — 2nd section



Firewall Logging

In addition to statistics, the firewall generates log entries whenever a packet is dropped, and for all other significant events. When a packet is dropped by the firewall, the log message saved by the firewall corresponds to the first table or rule that denied the packet, as described in the "Advanced Configuration" on page 33 section.

For example, if the firewall generates a "Blocked IP option" message because a TCP packet has a disallowed IP option, the dropped packet might not have passed the TCP rules, but since it was blocked by the IP option table first, "Blocked IP option" is the message saved by the firewall.

In the previous section "Firewall Statistics" on page 39, the telnet packet that was generated also causes a "Blocked port" message for port 1003 — unless another table blocks it first, in which case a log message for that table will be generated. In the latter case, the timestamps in the log messages can be used to correlate those log entries that were created during the test.

Other events that generate log entries include changing to a different profile, packets dropped by an advanced firewall security option, enabling and disabling an NVIDIA network interface, and any other changes to the firewall configuration.

Note: Log entries are saved in batches so that the most recent logs may take a short time to appear in the ForceWare Network Access Manager Web interface.

1. To view the log page (Figure 4.8), click **Log** from the menu.
2. Then use the links at the bottom of the page (**First**, **Previous**, **Next**, and **Last**) to navigate.
3. If you see too many log entries being generated, you can do one of the following:
   - Click **Clear All Logs** *or*
   - Choose **Log Settings** from the menu and consider changing the log "filter" setting from the **User Log Filters** list, as shown in Figure 4.9.

Figure 4.8 Firewall Logging Messages

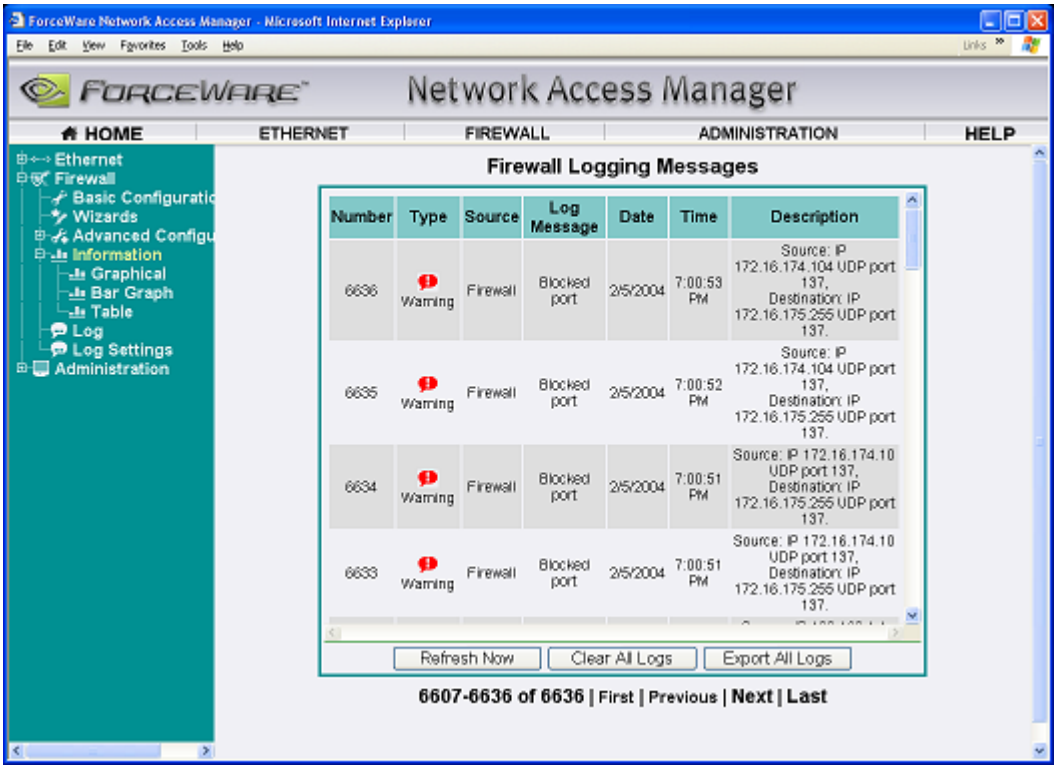
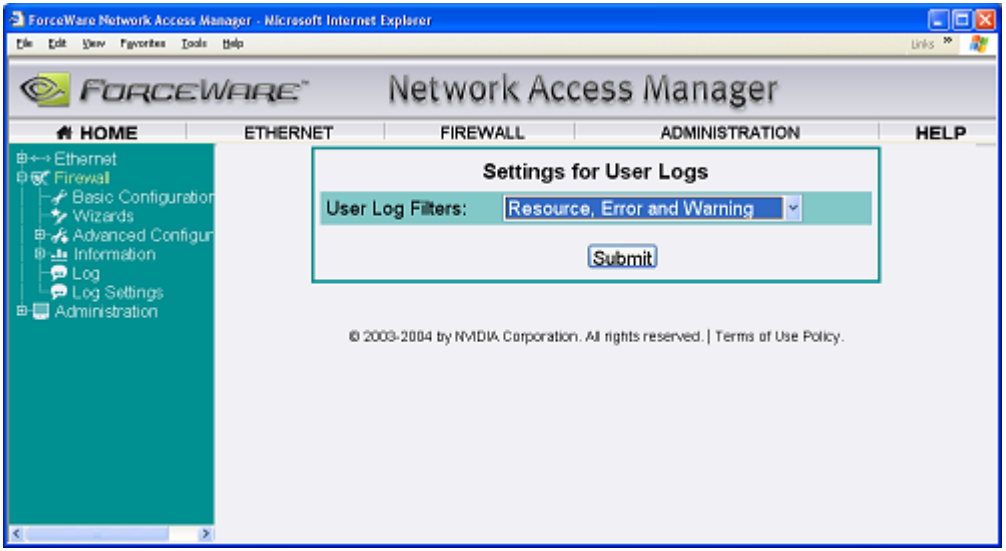


Figure 4.9 User Log Settings



CHAPTER 5

ADMINISTRATIVE TASKS

This chapter contains the following topics:

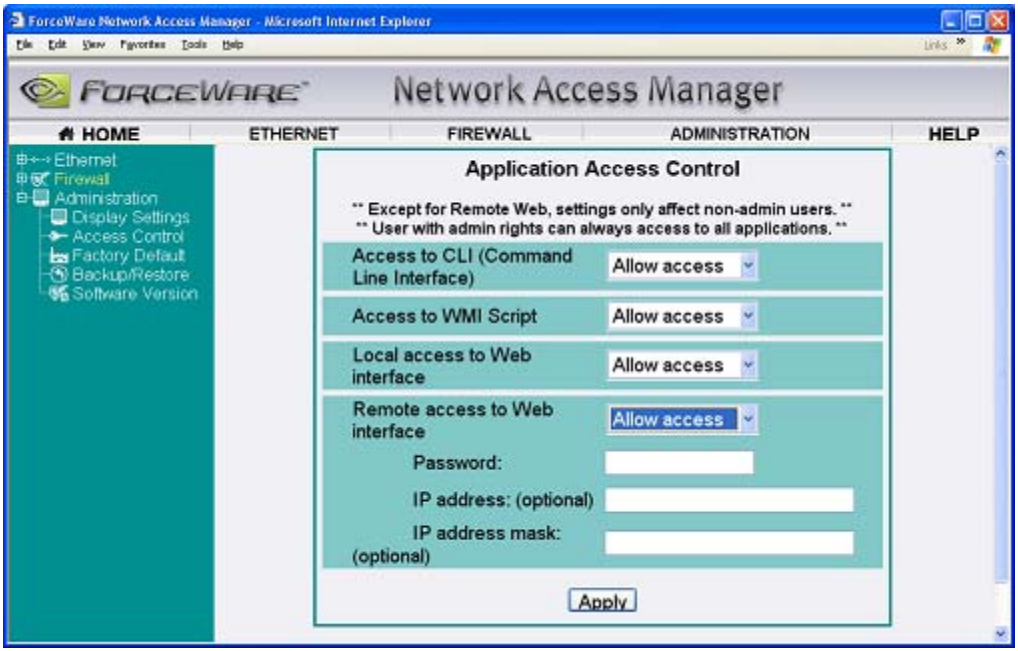
Accessing the Administration Menu

1. Open the **ForceWare Network Access Manager** Web menu.
2. Click the **Administration** menu on the left of the window to expand it so that you can see the various menu choices.
3. Click the menu item to display its associated page on the right.

Application Access Control Page

From the Administration menu, click **Access Control** to display the **Application Access Control** page (Figure 5.1).

Figure 5.1 Application Access Control Settings



This Application Access Control page lets you configure the application access permissions. Note the following about these permissions:

- Can only be configured from the local computer by an Administrator.

  An administrator on a local computer has access to all applications and configuration information — i.e., WMI scripts, the command line, and the Web interfaces, provided they are installed on the computer. The access control settings do not affect the Administrator.

- Cannot be viewed, accessed, or configured *remotely*, even by an Administrator.

- Apply only to non-Administrator users and remote users.

Note: Most of the access control in place will work only if the applications are installed on the NTFS file system, so it is recommended that you use NTFS, however the application will still function if installed on a FAT file system.

Default Administrative Access Control Settings

Figure 5.1 shows the *default* access settings of the ForceWare Network Access Manager software.

Note: You can also control access by using nCLI parameters such as `AccessCLI`, `AccessWMIScript`, etc.

Table 5.1 NVIDIA Firewall — Personal vs. Professional Features

Feature	Type of Access			
	nCLI Access	**WMI Script Access**	**Web Local Access**	**Web Remote Access**
Ethernet	Any user	Any user	Any user	Administrator only
Firewall	Administrator only	Administrator only	Administrator only	Administrator only
Changing access settings	Administrator only	Administrator only	Administrator only	None

Command Line Access

Note: The **Access to CLI** parameter is displayed only if the nCLI program is installed on the computer.

Default: Allow access

This field lets you specify whether to **Allow** or **Deny** command line access to the non-Administrator users.

If local command line access is denied, non-Administrator users cannot access the Network Access Manager. Regardless of this setting, users with Administrator privileges can always access the Web interface.

WMI Script

Default: Allow access.

This field lets you specify whether to **Allow** or **Deny** WMI scripting access to the non-Administrator users.

If disabled, no instances of WMI classes, which are part of the NVIDIA namespace, will be available through WMI script or other third party WMI application.

Administrator users can always access WMI using scripts.

Local Web Access

Default: Local Web access is **Allow**.

This options allows or denies access to the Web interface from the local computer.

If local Web access is denied, non-Administrator users cannot access the Network Access Manager. Regardless of this setting, users with Administrator privileges can always access the Web interface.

Remote Web Access

Default: Remote Web access is **Deny**.

Note: Enabling remote access is a security risk, since the configuration information is passed over the network in the clear (it is not encrypted).

When connecting to the Web interface from a remote computer using the following command:

```
http://<computer name>:3476
```

type **admin** as the user name, as shown below:

username: **admin**

password: _____ (password is blank by default)

Note: The password for this account can be changed.

The username and the password are sent over the network in the clear (it is not encrypted). The remote user then obtains Administrator access rights.

Additional Notes

- Remote access to Network Access Manager is most suitable from a home environment.
- Remote access to Network Access Manager provides limited access to the IP address/mask and can also be restricted based on the IP address or subnet address.
- Remote access to the Network Access Manager's Web-based interface is unencrypted and can be sniffed easily. It is highly recommended that you connect to this interface from the local network or a secure channel such as a **virtual private network (VPN)**.+

Password

Default: By default there is no password — that is, the password string is empty.

When you enable remote Web access, you can set a password.

Note: The user name for remote access is "`admin`."

IP Address and IP Address Mask (optional)

Default: By default there is no IP address or mask.

An IP address or a subnet (specified as a combination of an IP address and an IP address mask) can be used to restrict remote access to the computer such that access is limited to computers on the indicated IP subnet.

Note: To restrict access to only one computer, you can specify an IP address and no IP address mask. Specifying an IP address mask *without* an IP address is *invalid*.

Restore Factory Defaults

Note: Only Administrator users can restore factory default values to the firewall.

1. Click **Ethernet** or **Firewall** to enable one of these options:
   - Click **Ethernet** to restore factory default values to all the Ethernet-related parameters.
   - Click **Firewall** to restore factory default values to all the Firewall-related parameters.
2. After you select either **Ethernet** or **Firewall**, click **Start Restore** to restore the Ethernet or Firewall factory default values.

   An alert appears asking you to confirm whether you want to wipe out your current settings and replace them with the default values.
3. To proceed click **OK**. To cancel the operation, click **Cancel**.

Display Settings

The **Display Settings** page allows you to configure the font size for the pages and the refresh rate for the statistics pages.

- **Statistics refresh rate (Min 1, Max 65535)** controls the refresh rate of all the statistics pages in the Web interface.
  - **Range of values**: 1 to 65535 seconds
  - **Default**: 10 seconds
- **Font size** controls the font size used in the Web interface. The options are:
  - **Default font**
  - **Small font**

Click **Apply** for the changes to take effect.

Backup/Restore

The Backup/Restore page allows you to backup your configuration to a file or restore your configuration from a file you specify.

- Click **Backup** to launch the "Backup Configuration" page described below, which will allow you to backup your configuration to a file.
- Click **Restore** to launch the "Restore User Configuration" page described below, which will allow you to restore the configuration you have backed up in a file.

Backup Configuration

The **Backup Configuration** page will allow you to export the current configuration into a file. You can select the filename and also provide a brief description to be added to the top of the file. Once the backup is completed, a link to the file will be provided. You can right click on the link and save the file to any folder you want.

Note: Only Administrator users can backup the firewall configuration.

- **Backup filename** is the filename of the backup file created.

  Note: The *default* file name is `export.txt`
- **Description**. You can enter a short description of the configuration you are backing up. This description will be added to the top of the file along with the date and time of the backup.

- **Configuration**. You can choose either the **Ethernet** or the **Firewall** component to backup.

  Note: If you don't choose one of the components, you will get an empty backup file.

- **Backup**. Click **Backup** to start backing up the configuration settings for the selected components.

Restore User Configuration

Note: Only an Administrator users can restore the firewall configurations.

This **Restore User Configuration** page lets you restore or import the configuration settings from a backup file, which will replace all your current configuration with the values is the file.

- **Configuration File to Upload.** Browse the folders in your computer and choose the backup file with the configuration you want to restore.

  Note: If you don't specify a file, the last configuration you exported will be restored.

- **Restore**. Click this button to restore configuration values contained in the specified file.

  Note: A warning will be displayed indicating that the network interface might have to be restarted for these settings to take effect. You might lose connection to the server and that you will be able to get back to the page by clicking the browser **Refresh** button once the changes are applied. To proceed click **OK**. To cancel the operation, click **Cancel**.

  At the end of the restore operation, a log appears, indicating any errors in the restore operation. You can restore the previous settings by clicking the **Restore Backup** button.

ForceWare Network Access Manager Software Version

From the main ForceWare Network Access Manager menu, click **Administration - Software Version** to display the **Network Access Manager Software Version** page.

This page displays the version information for all the ForceWare Network Access Manager files you have installed on this computer.

Note: The version information is useful when you contact the computer manufacturer for technical support.

CHAPTER 6

USING WMI SCRIPT

This chapter contains the following topics:

Before You Begin

Using WMI script language is recommended *only* for Administrators who are already familiar with programming in WMI script and who have become familiar with the syntax and characteristics of configuration parameters — see "Ethernet Parameters Reference" on page A-70 and "NVIDIA Firewall Parameters Reference" on page B-108.

Note: For further information, you may want to consult the Microsoft documentation on WMI scripting.

Benefits of Using WMI Script

WMI script programming is being used by the IT staff of larger corporations to carry out day to day maintenance work. The overall benefits of using WMI scripts include:

- **Industry standard** — WMI can be implemented using languages such as Visual Basic Script and JavaScript.
- **Ease of use**
- **Common scripts** — allow access to NVIDIA ForceWare Network Access Manager data.
- **Flexibility** — The WMI script user can utilize the power of the script languages to meet almost any requirements. For example, as an Administrator, you can write a WMI script to scan for Yahoo Messenger on a computer and open the appropriate port if the computer user has sufficient rights.
- **Remote use** — you can run WMI script remotely and use it as a deployment tool in an organization. See "Configuration Deployment" on page 21.

Overview

WMI technology is Microsoft Windows's implementation of **Web-Based Enterprise Management (WBEM)**, an industry standard for management infrastructure that supports **Common Information Model (CIM)**, **Managed Object Format (MOF)**, and a common programming interface.

WMI consists of a management infrastructure (CIM object manager) and WMI custom Providers that communicate with each other through a common programming interface using **Component Object Model (COM)**.

The WMI technology also provides support for third-party Custom Providers. **Custom Providers** can be used to service requests related to managed objects that are environment-specific.

Providers typically do the following:

- Use the **MOF** language to define and create classes.
- Use the **WMI API** to
  - access the **CIM Object Manager** (**CIMOM**) object repository
  - respond to CIMOM requests made initially by applications.

The ForceWare Network Access Manager solutions supports

- **CIM** extension schemas
- **Custom Providers**.

For further details. see the following Web site:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwmi/html/wmiscript.asp

Advanced Topics

NVIDIA Namespace

NVIDIA ForceWare Network Access Manager classes are located under `root/NVIDIA namespace` in the WMI repository.

Note: It is strongly recommended that you do not modify anything in the NVIDIA namespace; for example, do not add or remove classes, or update their qualifiers. Modifying these items can prevent the proper functioning of the ForceWare Network Access Manager software.

WMI Provider

NVIDIA implements an extensible instance provider to manage the NVIDIA-specific objects. It is a COM in-proc server.

Synchronization

NVIDIA management framework ensures that only one Web, nCLI, or WMI script user interface is running at any given time. This feature is implemented to avoid data synchronization problems and improve the user experience.

Note: Within the WMI script, you can execute more than one script at any given time. However, doing so can potentially introduce data inconsistency. NVIDIA recommends that you run only one script at a time.

Sample Scripts

WMI script usage samples are provided in the following *subdirectories* under the *default* path of **`c:\nvidia\NetworkAccessManager`**, or your user-specified path:

- **`samples\Eth`**
- **`samples\Firewall`**

For example, Firewall WMI script examples are in:

`sample\Firewall\PerFireWMIScriptExamples.js`

You can cut and paste the appropriate command and use them in a batch file or the command line.

CHAPTER 7

USING THE COMMAND LINE INTERFACE (CLI)

This chapter contains the following major sections:

Conventions Used

Text in "code" font (`this is code font`) means it is text that is displayed on your screen. Text in bold "code" font (**`bold code font`**) indicates text you type on your computer.

About Examples Used

Examples are used to show how to use the nCLI command and parameters in "Expert" mode (not Interactive mode) to configure some of the networking features of the ForceWare Network Access Manager application. You can simplify the example to suit your needs.

Note: Examples are also provided in the `samples` subdirectory, under the default path of **`c:\nvidia\NetworkAccessManager`**, or your user-specified path.

Parameters

The `nCLI` command accepts the following classes of parameters:

- **Single** parameters contain a single value of some type.
- **Table** parameters contain data grouped in rows. Each row follows a fixed structure. You can only perform row operations on tables.
- **Group** parameters, such as **`Group get`** is useful in that you can view the value of all parameters inside a group with one command.
- **Namespace** parameters are a collection of tables and other parameters. Namespace is a way to group parameters. You can only browse into a namespace. No `Set` or `Get` commands are allowed on namespace parameters.

Modes of Operation

You can run nCLI in either **expert mode** or **interactive mode**. nCLI also supports import/export functions and expert commands grouped in batch files.

The key difference between expert mode vs. interactive mode is whether the control is switched back to command prompt when a command has completed.

- **Expert mode**. In expert mode, the control is switched back to the command prompt after a command has completed executing.

  From the command prompt, if you type **`ncli`** followed by a parameter, you exit to the command prompt after the command has completed.

- **Interactive Mode**: In interactive mode, the control remains in nCLI until you type **`quit`** to exit nCLI. You remain in the nCLI shell during interactive operations.

  You can enter interactive mode in two ways,

First Method

a From the command prompt, type `ncli` and press `Enter`.

The nCLI command prompt (`nCLI>`) appears to indicate nCLI is ready to accept a command.

b You can now type commands in the nCLI mode without having to prefix the keyword **`ncli`**.

Second Method

Enter an incomplete command from the command prompt. For example:

```
ncli set ASFSupport
```

nCLI automatically enters interactive mode. When this command completes, you will exit to the command prompt.

Using Single Parameters

`Get` and **`Set`** are the two most frequently used nCLI operations.

- **`Get`** is used to retrieve the setting of a parameter and can be invoked on single, group, and table parameters.
- **`Set`** is used to change or update the current setting of a parameter. It can be used in an "expert" mode, where the command is done in one line, or it can be used in "interactive" mode.

Single parameter `Get` and `Set` operations are discussed with examples in the sections that follow.

Set (Expert Mode)

Using the **`Set`** command in expert mode is intended for expert users to set a single parameter on a single computer. Using expert set requires knowing the correct (error-free) format or selection for the parameter and, therefore, requires familiarity with the distinguished name of the single parameter.

Some frequently set parameters, such as **`ASFSupport enable`** or **`ASFSupport disable`**, are usually set using expert mode.

Note: These commands can also be included in script or batch files.

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli set ASFSupport
enable
```

Set (Interactive Mode)

Using interactive set doesn't require too much prior knowledge of the parameter. In the following case, the parameter to be set, `ASFSupport`, is a selection, so the two choices are shown to help you select a value.

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli set ASFSupport
NVIDIA ForceWare Network Access Manager Framework Version
01.00
ASFSupport:
1 Disable
2 Enable
choose one(Enable): 1
```

Get

```
C:\nvidia\NetworkAccessManager\bin>ncli get ASFSupport
NVIDIA ForceWare Network Access Manager Framework Version
01.00
ASFSupport enable
```

Help

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli help ASFSupport
NVIDIA ForceWare Network Access Manager Framework Version
01.00
Enable or disable ASF (Alert Standard Format). ASF is an in-
dustry specification that defines alerting capability in
both OS-present and OS-absent environments.
```

Using Table Parameters

A table is a collection of groups (rows) that share the same fields (columns). Tables are frequently used to store the settings for firewall rules, filters, and statistics. Each row inside the table is uniquely identified by a **key**. A key is composed of one or more of fields of a row.

Note: Only *expert users* need to know the key format and composition.

nCLI supports both interactive and expert operations on tables.

- **Interactive** mode is recommended for average users.
- **Expert** operations on tables are usually executed through batch files. Expert users can also use the `export/import` method and text file to set up tables quickly.

Add Row

The following example shows how to add three rows to an empty table (`NV_FwlEtherType`), edit the table (see "Edit Row" on page 61), and then delete (see "Delete Row" on page 61) one row.

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli addrow NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version
01.00
EtherType:2048
EtherTypeName:IP
EtherTypeRule
1 Deny
2 Allow
choose one: 2
C:\nvidia\NetworkAccessManager\bin>ncli addrow
NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version
01.00
EtherType:2054
EtherTypeName:ARP
EtherTypeRule
1 Deny
2 Allow
choose one: 2
C:\nvidia\NetworkAccessManager\bin>ncli addrow
NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version 01.00
EtherType:32923
EtherTypeName:AppleTalk
EtherTypeRule
1 Deny
2 Allow
choose one: 1
```

Edit Row

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli editrow NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version 01.00
```

#	EtherType	EtherTypeName	EtherTypeRule
1	2048	IP	Allow
2	2054	ARP	Allow
3	32923	AppleTalk	Deny

```
Select a row to edit: 3
EtherType(32923)=2056
EtherTypeName(AppleTalk)=Frame Relay ARP / Inverse ARP
EtherTypeRule:
1 Deny
2 Allow
choose one(Deny): 2
```

Delete Row

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli delrow NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version
01.00
```

#	EtherType	EtherTypeName	EtherTypeRule
1	2048	IP	Allow
2	2054	ARP	Allow
3	2056	Frame Relay A..	Allow

```
Select a row to delete: 3
Are you sure? (y/n): Y
```

Help

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli help NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version 01.00
Firewall rules for different Data Link Layer protocols
Firewall rules for different Data Link Layer protocols (identified
by Ethernet type) including IP, IPX, NetBEUI, AppleTalk and other
protocols.
```

Set Table

Invoking the **nCLI `set`** command on table parameters guides you through different operations that can be performed on a table. In the following example, a row is added to the table, then edited, and finally deleted.

Note: The **`Set table`** command does not require that you to know the **`addRow`**, **`delRow`**, and **`editRow`** command names.

Examples

```
C:\nvidia\NetworkAccessManager\bin>ncli set NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version
01.00
Select an option: AddRow(A), EditRow(E), Purge(P), Del-
eteRow(D), Quit(Q):A
EtherType:32923
EtherTypeName:AppleTalk
EtherTypeRule
1 Deny
2 Allow
choose one: 1
```

```
C:\nvidia\NetworkAccessManager\bin>ncli set NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version
01.00
Select an option: AddRow(A), EditRow(E), Purge(P), Del-
eteRow(D), Quit(Q):E
EtherType(32923)=33079
EtherTypeName(AppleTalk)=IPX
EtherTypeRule:
1 Deny
2 Allow
choose one(Deny): 2
```

```
C:\nvidia\NetworkAccessManager\bin>ncli set NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version
01.00
Select an option: AddRow(A), EditRow(E), Purge(P), Del-
eteRow(D), Quit(Q):D
```

#	EtherType	EtherTypeName	EtherTypeRule
1	2048	IP	Allow
2	2054	ARP	Allow
3	33079	IPX	Allow

```
Select a row to delete: 3
Are you sure? (y/n): y
```

Get Table

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli get NV_FwlEtherType
NVIDIA ForceWare Network Access Manager Framework Version
01.00
```

#	EtherType	EtherTypeName	EtherTypeRule
1	2048	IP	Allow
2	2054	ARP	Allow

About Expert Commands

Due to the inherent complexity, expert commands are not as intuitive as interactive commands.The syntax of an expert command is shown below. Examples are also provided in the samples subdirectory, under the default path of **c:\nvidia\NetworkAccessManager**, or your user-specified path.

Syntax

ncli addrow <tablename>
<column1>=<column1value>,<column2>=<column2value>,..î

ncli editrow
<tablename>.<key1>=<key1value>,<key2>=<key2value>,..î
<column1>=<column1value>,<column2>=<column2value>,..î

ncli delrow <tablename>.<key1>=<key1value>,<key2>=<key2value>,..î

In the following example:

- A new row for IPv6 EtherType is added and initially set to `Allow`.
- The table is then edited with the IPv6 Ethertype rule set to `Deny`.
- Finally, the entire row is deleted.

Examples

```
C:\nvidia\NetworkAccessManager\bin>ncli addrow NV_FwlEtherType
"EtherType=34525,EtherTypeName=IPv6,EtherTypeRule=Allow"
```

```
C:\nvidia\NetworkAccessManager\bin>ncli editrow
NV_FwlEtherType.EtherType=34525"
"EtherType=34525,EtherTypeName=IPv6,EtherTypeRule=Deny"
```

```
C:\nvidia\NetworkAccessManager\bin>ncli delrow
NV_FwlEtherType.EtherType=34525î
```

About Other Table Commands

Note: The `purge` command is used to delete all the rows in the table; i.e., the entire table. Please use this command cautiously.

Syntax

`purge <`*`tablename`*`>`

Note: If the table has read-only access, the purge action will fail.

Browsing the Parameter Structure

The ForceWare networking parameters are organized in a tree structure. You can explore the tree structure. The browsing capability of nCLI is a powerful tool for non-expert use as one does not have to know the parameter's distinguished name before using the command.

List

The **`ls`** or **`dir`** command lists the children of the current parameter, as shown in the next example.

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli
NVIDIA ForceWare Network Access Manager Framework Version
01.00

ncli>ls
NS_Eth
NS_NvConfig
NS_Firewall
NS_UserLog
NS_Security

ncli>ls ns_eth
NS_EthStat
NS_EthConfig
NS_ASF
NV_DriverRestartCmd
NV_DriverRestartFlag

ncli>
```

Changing Directory

The `cd` command lets you browse through the parameter tree structure.

Example 1

```
C:\nvidia\NetworkAccessManager\bin>ncli
NVIDIA ForceWare Network Access Manager Framework Version
01.00
ncli>ls
NS_Eth
NS_NvConfig
NS_Firewall
NS_UserLog
NS_Security
ncli>cd NS_Eth
ncli>ls
NS_EthStat
NS_EthConfig
NS_ASF
NV_DriverRestartCmd
NV_DriverRestartFlag
ncli>cd ns_ethstat
ncli>ls
NV_NetworkGenStat
NV_EthStat
ncli>
```

Example 2

Note: Invoking the cd command by itself will bring you to the root level, as shown in the following example.

```
C:\nvidia\NetworkAccessManager\bin>ncli
NVIDIA ForceWare Network Access Manager Framework Version
01.00
ncli>cd ns_eth
ncli>cd ns_ethstat
ncli>cd
ncli>
```

Each ForceWare Network Access Manager parameter has a *unique* name, which can be used within `ncli>` to access each individual parameter."

Therefore, you do not need the complete path to get to a single parameter. The example below shows how this can help you quickly access a parameter.

```
C:\nvidia\NetworkAccessManager\bin>ncli
NVIDIA ForceWare Network Access Manager Framework Version
01.00
ncli>cd ASFSupport
ncli>pwd
<root>/NS_Eth/NS_ASF/NV_ASF/ASFSupport
```

```
ncli>
```

Current Working Directory

The `pwd` command is used to display the path to the current parameter.

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli
NVIDIA ForceWare Network Access Manager Framework Version
01.00
ncli>cd ns_ethstat
ncli>pwd
<root>/NS_Eth/NS_EthStat
ncli>cd
ncli>pwd
<root>
ncli>
```

Context-Sensitive Operations

`ls`, **`cd`**, and **`pwd`** commands allow you to browse through the parameters. When you have entered a current parameter, all the operations you invoke will be in the context of that parameter.

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli
NVIDIA ForceWare Network Access Manager Framework Version 01.00
ncli>cd NV_FwlEtherType
ncli>get
```

#	EtherType	EtherTypeName	EtherTypeRule
1	2048	IP	Allow
2	2054	ARP	Allow

```
ncli>help
Firewall rules for different Data Link Layer protocols
Firewall rules for different Data Link Layer protocols (identified
by Ethernet type) including IP, IPX, NetBEUI, AppleTalk and other
protocols.
ncli>addrow
EtherType:2056
EtherTypeName:FrameRelay ARP/Inverse IP
EtherTypeRule
1 Deny
2 Allow
choose one: 2
```

```
ncli>get
```

#	EtherType	EtherTypeName	EtherTypeRule
1	2048	IP	Allow
2	2054	ARP	Allow
3	2056	FrameRelay AR..	Allow

```
ncli>
```

Text File Processing

Text file processing is intended for expert users to quickly update complex parameters and perform large configurations.

For example, you can use the nCLI command line to perform interactive settings *only* on tables. Text file processing offers an alternative to the Get and Set parameter values in a flat text format.

Export

Export files follow a standard format that will make it compatible with Web-based management. That is, export files from nCLI can be imported using the Web-based management and export files from Web-based management can be imported using nCLI.

Syntax

`export /f<filename> <parameter_name>`

Note that either one or both of `/f<filename>` and `<parameter name>` may be omitted.

- If ***`/f <filename>`*** is omitted, the output of the export will stored in **`frontend\backup\cliexport.txt`** under the directory where ForceWare Network Access Manager software is installed.
- If ***`<parameter_name>`*** is omitted, then only the current parameter and its children will be exported. An example is shown below.

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli
NVIDIA ForceWare Network Access Manager Framework Version
01.00
ncli>export
.....................................................
...................Finished
ncli>
```

Import

Before importing new parameter settings, old parameter settings are backed up to prevent any problems during import that could throw the system into an unknown state. If necessary, the backup file can be used to restore the system to the previous state.

Note: If nCLI encounters problems in importing parameters, it will abort and instruct you to restore to the previous state. Use the `restore` command for recovery.

Syntax

`import /f <filename>`

If **`/f <filename>`** is omitted, the default file **`frontend\backup\cliexport.txt`** under the directory where ForceWare Network Access Manager software will be read and imported.

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli
NVIDIA ForceWare Network Access Manager Framework Version 01.00
ncli>import
Reading text and importing
.............................................................
.......Backing up to clibackup.txt in case of failure
.............................................................
.............................................................
Finished Import.
ncli>
```

Selective Export

Selective export allows you to export only the parameter branch specified.

Syntax

`export /f <file name> <parameter_name>`

Example

To export only the `NS_xxxx` namespace, the following command can be used.

`ncli export /f c:\xxxx_export.txt ns_xxxx`

```
NVIDIA ForceWare Network Access Manager Framework Version
01.00

..Finished
```

Context Export

nCLI lets you browse into a parameter branch and export it.

Example

```
C:\nvidia\NetworkAccessManager\bin>ncli
NVIDIA ForceWare Network Access Manager Framework Version
01.00
ncli>cd ns_eth
ncli>export
ncli>
```

As a result, only the `NS_Eth` branch is exported.

Glossary

See "Glossary" on page 147.

APPENDIX

ETHERNET PARAMETERS REFERENCE

Note: For references to all the individual parameters, categorized by group, see the entries listed for this appendix — **A. Ethernet Parameters: Reference** — in the "Table of Contents" on page iii.

Group: Remote Wakeup

Remote Wakeup

Parameter	WakeUp
Description	Enables or disables Ethernet remote wake up capability. When enabled, the user can remotely turn on the power of systems across the network. For example, a network administrator can use Remote Wake Up to perform after-hours maintenance from a remote location without requiring a technician to be physically present.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthWakeUp **Single**: WakeUp
Usage example:	`nCLI Set "WakeUp" "Enable"`
Access	ReadWrite
Data type	Selection
User selection	Disable *or* Enable

Remote Wakeup by Magic Packet

Parameter	**WakeUpMagic**	
Description	Enables or disables the magic packet wake-up feature. When this feature is enabled, networked computers that are in a low power state receive the "magic packet" to wake up.	
Comment	If WakeUp is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthWakeUp **Single**: WakeUpMagic	
Usage example:	`nCLI Set "WakeUpMagic" "Enable"`	
Access	ReadWrite	
Restart network:	Network restart is required.	
Data type	Selection	
User selection	Disable	Enable

Remote Wakeup (Pattern Match)

Parameter	**WakeUpPattern**	
Description	Enables or disables the pattern match remote wakeup feature. When this feature is enabled, networked computers that are in a low power state receive a packet that contains a pattern specified by the operating system's network protocol to wake up.	
Comment	If WakeUp is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthWakeUp **Single**: WakeUpPattern	
Usage example:	`nCLI Set "WakeUpPattern" "Enable"`	
Access	ReadWrite	
Restart network:	Network restart is required.	
Data type	Selection	
User selection	Disable	Enable

Remote Wakeup (Link State Change)

Parameter:	**WakeUpLink**	
Description	Enables or disables the WakeUpLink feature. Change in the link state refers to the connection or disconnection of the Ethernet network cable. When a networked computer is in a low power state, a change in the link state wakes up the computer.	
Comment	If WakeUp is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthWakeUp **Single**: WakeUpLink	
Usage example:	`nCLI Set "WakeUpLink" "Enable"`	
Access	ReadWrite	
Network restart:	Required	
Data type	Selection	
User selection	Disable	Enable

Remote Wake Up from Hibernate or Shutdown

Parameter	**WakeUpS4S5**	
Description	Enables or disables the Remote Wake Up from Hiberate or Shutdown feature. Hibernate means that all devices in a networked computer are turned off. This state is saved to the computer's hard disk and is then used for a fast startup. Shutdown means that the operating system will shut down and the BIOS will be re-initialized during wake up.	
Comment	If WakeUp is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthWakeUp **Single**: WakeUp S4S5	
Usage example:	`nCLI Set "WakeUpS4S5" "Enable"`	
Access	ReadWrite	
Network restart:	Required	
Data type	Selection	
User selection	Disable	Enable

Group: Protocol Offload

Checksum Offload

Parameter	**EthOffloadChkSum**	
Description	Enables or disables the Ethernet checksum offload feature. Offloads increase the system performance by offloading TCP/IP CPU-intensive tasks to hardware.	
Comment	This feature is not supported by WMI scripting.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Offload **Single**: EthOffloadChkSum	
Usage example	`nCLI Set "EthOffloadChkSum" "Enable"`	
Access	ReadWrite	
Network restart	Required	
Data type	Selection	
User selection	Disable	Enable

IPv4 Transmit Checksum Offload

Parameter	**EthOffloadIPv4TxChkSum**	
Description	Enables or disables the IPv4 Transmit Checksum Offload feature. When this feature is enabled, the operating system passes the task of calculating IP (Internet Protocol) checksums for transmitted packets to the Ethernet hardware.	
Comment	This parameter is not supported by WMI scripting. If EthOffloadChkSum is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Offload **Single**: EthOffloadIPv4TxChkSum	
Usage example:	`nCLI Set "EthOffloadIPv4TxChkSum" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

IPv4 Receive Checksum Offload

Parameter:	**EthOffloadIPv4RxChkSum**	
Description	Enables or disables the IPv4 Receive Checksum Offload feature. When this feature is enabled, the operating system passes the task of calculating IP checksums for received packets to the Ethernet hardware.	
Comment	This parameter is not supported by WMI scripting. If EthOffloadChkSum is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Offload **Single**: EthOffloadIPv4RxChkSum	
Usage example:	`nCLI Set "EthOffloadIPv4RxChkSum" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

UDP Transmit Checksum Offload

Parameter	**EthOffloadUDPTxChkSum**	
Description	Enable or disables the UDP (User Datagram Protocol) Transmit Checksum Offload feature. When this feature is enabled, the operating system can use the Ethernet hardware to calculate UDP checksums for transmitted packets.	
Comment	Not supported through WMI script. If EthOffloadChkSum is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Offload **Single**: EthOffloadUDPTxChkSum	
Usage example:	**nCLI Set "EthOffloadUDPTxChkSum" "Enable"**	
Access	ReadWrite	
Data type	Selection	
User selection	Enable	Disable

UDP Receive Checksum Offload

Parameter	**EthOffloadUDPRxChkSum**	
Description	Enables or disables the UDP Receive Checksum Offload feature. When the feature is enabled, the operating system can use the Ethernet hardware to calculate UDP checksums for received packets.	
Comment	This parameter is not supported by WMI scripting. If EthOffloadChkSum is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Offload **Single**: EthOffloadUDPRxChkSum	
Usage example:	`nCLI Set "EthOffloadUDPRxChkSum" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

TCP Transmit Checksum Offload

Parameter	**EthOffloadTCPTxChkSum**	
Description	Enables or disables the TCP Transmit Checksum Offload feature. When the feature is enabled, the operating system can use the Ethernet hardware to calculate TCP checksums for transmitted packets.	
Comment	This parameter is not supported by WMI scripting. If EthOffloadChkSum is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Offload **Single**: EthOffloadTCPTxChkSum	
Usage example:	`nCLI Set "EthOffloadTCPTxChkSum" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

TCP Receive Checksum Offload

Parameter	**EthOffloadTCPRxChkSum**	
Description	Enables or disables the TCP Receive Checksum Offload feature. When the feature is enabled, the operating system can use the Ethernet hardware to calculate TCP checksums for received packets.	
Comment	This parameter is not supported by WMI scripting. If EthOffloadChkSum is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Offload **Single**: EthOffloadTCPRxChkSum	
Usage example:	`nCLI Set "EthOffloadTCPxChkSum" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

TCP Large Send Offlload

Parameter	**EthOffloadTxLargeSend**	
Description	Enables or disables the TCP Large Send OfflLoad feature. When the feature is enabled, the operating system can utilize the Ethernet hardware capabilities to segment large TCP packets into smaller packets. **Note:** This feature applies to packet transmissions only.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Offload **Single**: EthOffloadTxLargeSend	
Usage example:	`nCLI Set "EthOffloadTxLargeSend" "Enable"`	
Access	ReadWrite	
Network restart	Required.	
Data type	Selection	
User selection	Disable	Enable

Group: Microsoft Operating System VLAN (Virtual LAN)

Microsoft Operating System VLAN

Parameter	**EthMSVLAN**	
Description	Specifies the Virtual LAN (VLAN) ID returned by the Microsoft operating system. The VLAN ID is an identifier used by a networked computer to determine its associated VLAN. VLAN allows a set of networked computers to function as if they were not connected to the same wire even though they may be physically connected to the same segments of a Local Area Network (LAN).	
Comment	The Microsoft VLAN ID overrides the NVIDIA EthVLAN and EthVLANID settings. When the Microsoft VLAN ID is 0 (zero), the NVIDIA EthVLAN and EthVLANID are used.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_MSVLAN **Single**: EthMSVLAN	
Usage example:	`nCLI Get "EthMSVLAN"`	
Access	Read	
Data type	Number (32 bit)	
Maximum value	4095	**Minimum Value:** 0

Group: VLAN (Virtual LAN)

VLAN Support

Parameter	**EthVLAN**	
Description	Enables or disables VLAN support. VLAN allows a network of computers to function as if they are not connected to the same wire even though they may be physically located on different segments of a LAN.	
Comment	The Microsoft VLAN ID overrides the NVIDIA EthVLAN and EthVLANID values. When the Microsoft VLAN ID is 0 (zero), the NVIDIA EthVLAN and EthVLANID are used.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_MSVLAN_Setting **Single**: EthVLAN	
Usage example:	`nCLI Set "EthVLAN" "Disable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

VLAN ID

Parameter	**EthVLANID**
Description	The VLAN ID is an identifier used by a computer to determine its associated VLAN. A value of 0 (zero) means VLAN is disabled. VLAN allows a set of networked computers to function as if they were not connected to the same wire even though they may be physically connected to same segments of a LAN.
Comment	The Microsoft VLAN ID overrides the NVIDIA EthVLAN and EthVLANID values. When the Microsoft VLAN ID is 0 (zero), the NVIDIA EthVLAN and EthVLANID are used.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_MSVLAN_Setting **Single**: EthVLANID
Usage example:	`nCLI Set "EthVLANID" "0"`
Access	ReadWrite
Data type	Number (32 bit)
Maximum value	4095 **Minimum value:** 0

Group: Jumbo Frame

Jumbo Frame Payload Size

Parameter	**EthJumboSize**			
Description	Specify the Ethernet jumbo frame payload size. Jumbo frame supports larger Ethernet packet sizes to reduce server overhead and increase throughput. Payload size of 1,500 means Jumbo Frame is disabled.			
Comment	Jumbo frame is supported only when the connection speed is 1000 Mbps.			
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_EthJumbo **Single**: EthJumboSize			
Usage example:	`nCLI Set "EthJumboSize" "1500"`			
Access	ReadWrite			
Network restart:	Required.			
Data type	Selection			
User selection	1500	2500	4500	9000

Group: Driver Optimization

Ethernet Driver Optimization

Parameter	EthOptimization
Description	Allows Ethernet driver optimization by adjusting Ethernet driver operating parameters to suit different needs.
Comment	This driver optimization profile feature is not supported by WMI scripting. WMI script users must configure parameters within the Ethernet Performance group individually. Profiles are listed and described in the **User selection** section below.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ Eth_Optimization **Group**: NV_ EthOptimization **Single**: EthJumboSize
Usage example:	`nCLI Set "EthOptimization" "CPU Utilization"`
Access	ReadWrite
Data type	Selection
User selection	**CPU Utilization** is a setting that optimizes to lower the amount of time CPU spent in processing network traffic. **Note:** This is the recommended and *default* setting. **Throughput** is a setting that maximizes the amount of network traffic sent and received. **Multimedia** is a setting that reduces the time spent per network interrupt to allow time-critical media devices to be serviced.

Group: Ethernet Performance

Number of Receive Buffers

Parameter	**EthNoOfRxBuff**									
Description	Specifies the number of receive buffers allocated by the NVIDIA Ethernet driver. Receive buffers are memory blocks used to store packets received from the network.									
Comment	For optimal performance, the number of receive buffers need to be at least TWICE the number of receive descriptors.									
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ Eth_Config **Group**: NV_ Eth_Performance **Single**: EthNoOfRxBuff									
Usage example:	`nCLI Set "EthNoOfRxBuff" "512"`									
Access	ReadWrite									
Network connection:	Restarting the network is required..									
Data type	Selection									
User selection	2	4	8	16	32	64	128	256	512	

Number of Receive Buffer Descriptors

Parameter	**EthNoOfRxDesc**						
Description	Number of receive buffer descriptors available to the Ethernet hardware. This value determines the number of receive buffers that may be queued for the hardware.						
Comment	For optimal performance, the number of receive buffers need to be set to at least *twice* the number of receive descriptors.						
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ Eth_Config **Group**: NV_ Eth_Performance **Single**: EthNoNoOfRxDesc						
Usage example:	`nCLI Set "EthNoOfRxDesc" "64"`						
Access	ReadWrite						
Netwrork connection:	Restarting the network is required.						
Data type	Selection						
User selection	2,	4	8	16	32	64	128	256

Number of Transmit Buffer Descriptors

Parameter	**EthNoOfTxDesc**						
Description	Specifies the number of transmit buffer descriptors available to the Ethernet hardware. This value determines the number of transmit buffers that may be queued for the hardware.						
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ Eth_Config **Group**: NV_ Eth_Performance **Single**: EthNoNoOfRxDesc						
Usage example	`nCLI Set "EthNoOfTxDesc" "256"`						
Access	ReadWrite						
Restart network	Yes, required for setting to take effect.						
Data type	Selection						
User selection	2 4	8 16	32	64	128	256	512	1024

Maximum Transmit Frames Queued

Parameter	**EthMaxTxPktQueue**				
Description	Specifies the maximum number of frames which may be queued by the Ethernet driver.				
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ Eth_Config **Group**: NV_ Eth_Performance **Single**: EthMaxTxPktQueue				
Usage example:	`nCLI Set "EthMaxTxPktQueue" "1024"`				
Access	ReadWrite				
Network Connection:	Restarting network is required.				
Data type	Selection				
User selection	2 4	8 16	32 64	128 256	512 1024

Number of Receive Packets to Process per Interrupt

Parameter	**EthNoOfRxPktToProcessEachTime**							
Description	Specifies the number of receive packet to process per interrupt.							
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ Eth_Config **Group**: NV_ Eth_Performance **Single**: EthNoOfRxPktToProcessEachTime							
Usage example:	`nCLI Set "EthNoOfRxPktToProcessEachTime" "1280"`							
Access	ReadWrite							
Network Connection:	Restarting network is required.							
Data type	Selection							
User selection	10	20	40	80	160	320	640	1280

Number of Transmit Packet to Process per Interrupt

Parameter	**EthNoOfTxPktToProcessEachTime**							
Description	Specifies the number of transmit packet to process per interrupt.							
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_ Eth_Performance **Single**: EthNoOfTxPktToProcessEachTime							
Usage example:	`nCLI Set "EthNoOfTxPktToProcessEachTime" "1280"`							
Access	ReadWrite							
Network connection:	Restarting the network is required.							
Data type	Selection							
User selection	5	10	20	40	80	160	320	640	1280

Interrupt Interval

Parameter	**EthPollingInterval**
Description	Specifies the time (in milliseconds) between hardware interrupts in the hardware polling mode.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Performance **Single**: EthPollingInterval
Usage example:	`nCLI Set "EthPollingInterval" "425"`
Access	ReadWrite
Network connection:	Restarting the network is required..
Data type	Selection
User selection	0, 425

Group: Traffic Prioritization

IEEE 802.1p Support

Parameter	**Eth8021p**	
Description	Enables or disables Ethernet IEEE 802.1p support. IEEE 802.1p allows frames to be grouped into priority classes.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_8021p **Single**: Eth8021p	
Usage example:	`nCLI Set "Eth8021p" "Disable"`	
Access	ReadWrite	
Network connection:	Restarting the network is required.	
Data type	Selection	
User selection	• Disable	• Enable

Group: Ethernet Speed/Duplex

Configurable Ethernet Speed/Duplex Settings

Parameter	**EthSpeed**	
Description	Specifies the configurable Ethernet speed/duplex settings.	
Comment	For systems equipped with Gigabit Ethernet PHY (physical layer transceivers), the "Autonegotiate for 1000 Mbps" selection is available. Otherwise, only the 100/10 Mbps selections are available.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Speed **Single**: EthSpeed	
Usage example:	`nCLI Set "EthSpeed" "Full Autonegotion"`	
Access	ReadWrite	
Restart network?	Yes, required for changes to take effect.	
Data type	Selection	
User selection	▪ Full Autonegotiation ▪ Autonegotiate for 1000 mbps Full Duplex ▪ Autonegotiate for 100 mbps Full Duplex ▪ Autonegotiate for 100 mbps Half Duplex	▪ Autonegotiate for 10 mbps Full Duplex ▪ Autonegotiate for 10 mbps Half Duplex ▪ Force 100 mbps Full Duplex ▪ Force 100 mbps Half Duplex ▪ Force 10 mbps Full Duplex ▪ Force 10 mbps Half Duplex

Group: Ethernet Information

Link Speed

Parameter	EthLinkSpeed
Description	Specifies the current speed (in Mbps) of the Ethernet device.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthInfo **Single**: EthLinkSpeed
Usage example:	`nCLI Get "EthLinkSpeed"`
Access	Read
Data type	Number (32 bit)
Maximum Value	10000
Minimum Value	0

Maximum Link Speed

Parameter	EthLinkMaxSpeed
Description	Specifies the maximum speed (in Mbps) at which the Ethernet interface can operate.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthInfo **Single**: EthLinkMaxSpeed
Usage example:	`nCLI Get "EthLinkMaxSpeed"`
Access	Read
Data type	Number (32 bit)
Maximum Value	10000
Minimum Value	0

Duplex Setting

Parameter	EthDuplex	
Description	Specifies the current Ethernet interface duplex setting. Full duplex means that the Ethernet interface on both ends of a link can receive and transmit data simultaneously over the cable. Half duplex means that either the transmit or the receive operation can occur at a given time.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthInfo **Single**: EthDuplex	
Usage example:	`nCLI Get "EthDuplex"`	
Access	Read	
Data type	Selection	
User selection	Half Duplex	Full Duplex

Link Status

Parameter	EthConnectStatus	
Description	Displays the current Ethernet link status. When the Ethernet link is disconnected, the remote configuration tool will not function.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthInfo **Single**: EthConnectStatus	
Usage example:	`nCLI Get "EthConnectStatus"`	
Access	Read	
Data type	Selection	
User selection	Connected	Disconnected

Promiscuous Mode

Parameter	EthPromiscuous	
Description	When this parameter is enabled, all packets (including frames addressed for other stations) that arrive at this Ethernet interface are received.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthInfo **Single**: EthPromiscuous	
Usage example:	`nCLI Get "EthPromiscuous"`	
Access	Read	
Data type	Selection	
User selection	Disable	Enable

Permanent Ethernet Address

Parameter	EthAddressPermanent
Description	Specifies the fixed Ethernet address encoded in the hardware.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_EthInfo **Single**: EthAddressPermanent
Usage example:	`nCLI Get "EthAddressPermanent"`
Access	Read
Data type	MAC Address

Group: Ethernet Address

Current Ethernet Address

Parameter	EthAddressCurrent
Description	Specifies the Ethernet address currently being used. The Ethernet interface then uses the Current Ethernet Address in place of the Permanent Ethernet Address.
Comment	Format of Ethernet address should be: *XX:XX:XX:XX:XX:XX*
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_Address **Single**: EthAddressCurrent
Usage example:	`nCLI Set "EthAddressCurrent" "0C:12:34:56:78:9A"`
Access	ReadWrite
Network connection:	Restarting the network is required.
Data type	MAC Address

Group: Network Interface information

Computer (Machine) Name

Parameter	MachineName
Description	Specifies the unique name that is used to identify a computer on the network domain. The computer (machine) name is specified through the operating system and must be unique within a network domain.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_InterfaceInfo **Single**: MachineName
Usage example:	`nCLI Get "MachineName"`
Access	Read
Data type	String
Maximum length	64

IP Address

Parameter	IPAddress
Description	Specifies the IP address of the current Ethernet interface.
Comment	If an interface has multiple IP addresses and masks, only the first set returned by the operating system is shown.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_InterfaceInfo **Single**: IPAddress
Usage example:	`nCLI Get "IPAddress"`
Access	Read
Data type	String
Maximum length	64

IP Address Mask

Parameter	IPAddressMask
Description	Specifies the IP address mask of the current Ethernet interface.
Comment	If an interface has multiple IP addresses and masks, only the first set returned by the operating system is shown.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_InterfaceInfo **Single**: IPAddressMask
Usage example:	`nCLI Get "IPAddressMask"`
Access	Read
Data type	String
Maximum length	64

Group: Factory Default

Factory Default

Parameter	**EthDefault**	
Description	Restores the Ethernet factory default settings.	
Comment	Restore factory default feature is not available through WMI scripting.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Group**: NV_Eth_FactoryDefault **Single**: EthDefault	
Usage example:	`nCLI Set "EthDefault" "Restore"`	
Access	ReadWrite	
Data type	Selection	
User selection	NoRestore	Restore

Table: Multicast Address List

Multicast Address List

Table Parameter	**NV_Eth_MulticastAddress**
Description	Specifies a list of multicast addresses from which the Ethernet interface will receive frames. The Ethernet multicast packet refers to packets addressed to a group of recipients.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Table:** NV_Eth_MulticastAddress
Usage example:	`nCLI Get "NV_Eth_MulticastAddress"`
Access	Read
Single parameter	**EthMulticast** (See the next tabe for details on the **EthMulticast** parameter.)

Multicast Addresses (Single Parameter)

Parameter	**EthMulticast**
Description	The Ethernet multicast packet refers to packets addressed to a group of recipients.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthConfig **Table:** NV_Eth_MulticastAddress **Single:** EthMulticast
Access	Read
Table key	This parameter is a key to the table
Data type	MAC Address

Group: Ethernet Statistics

Frames Received with Alignment Error

Parameter	**EthReceiveErrorAlign**
Description	Specifies the number of received frames with alignment errors.
H*ierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group:** NV_EthStat **Single:** EthReceiveErrorAlign
Usage example:	`nCLI Get "EthReceiveErrorAlign"`
Access	Read
Data type	Number (64 bit)

Frames Transmitted After One Collision

Parameter	EthTransmitOneCollision
Description	Specifies the number of frames that successfully transmitted after encountering one collision.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group:** NV_EthStat **Single:** EthTransmitOneCollision
Usage example:	`nCLI Get "EthTransmitOneCollision"`
Access	Read
Data type	Number (64 bit)

Frames Transmitted After Two or More Collisions

Parameter	EthTransmitMoreCollision
Description	Specifies the number of frames that successfully transmitted after encountering two or more collisions.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group:** NV_EthStat **Single:** EthTransmitMoreCollision
Usage example:	`nCLI Get "EthTransmitMoreCollision"`
Access	Read
Data type	Number (64 bit)

Frames Transmitted After Deferral

Parameter	EthTransmitDeferred
Description	Specifies the number of frames that successfully transmitted after the Ethernet hardware defers transmission at least once.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group:** NV_EthStat **Single:** EthTransmitDeferred
Usage example:	`nCLI Get "EthTransmitDeferred"`
Access	Read
Data type	Number (64 bit)

Display Name Frames Exceed Maximum Collision

Parameter	EthTransmitMaxCollision
Description	Specifies the number of frames not transmitted because of excessive collisions.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group:** NV_EthStat **Single:** EthTransmitMaxCollision
Usage example:	`nCLI Get "EthTransmitMaxCollision"`
Access	Read
Data type	Number (64 bit)

Frames with Overrun Errors

Parameter	EthReceiveOverrun
Description	Specifies the number of frames not received because of overrun errors. An overrun error occurs when the Ethernet hardware receives more data than it can process.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_EthStat **Single**: EthReceiveOverrun
Usage example:	`nCLI Get "EthReceiveOverrun"`
Access	Read
Data type	Number (64 bit)

Frames with Underrun Errors

Parameter	EthTransmitUnderrun
Description	Specifies the number of frames not transmitted because of underrun errors. An underrun error occurs when the Ethernet hardware cannot transmit frames because the data is not available within the expected time.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_EthStat **Single**: EthTransmitUnderrun
Usage example:	`nCLI Get "EthTransmitUnderrun"`
Access	Read
Data type	Number (64 bit)

Frames with Heartbeat Failure

Parameter	EthTransmitHeartbeatFail
Description	Specifies the number of frames transmitted without detection of the collision-detect heartbeat.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_EthStat **Single**: EthTransmitHeartbeatFail
Usage example:	`nCLI Get "EthTransmitHeartbeatFail"`
Access	Read
Data type	Number (64 bit)

Carrier Sense (CRS) Signal Lost

Parameter	EthTransmitTimesCRSLost
Description	Specifies the number of times the CRS signal has been lost during packet transmission.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_EthStat **Single**: EthTransmitTimesCRSLost
Usage example:	`nCLI Get "EthTransmitTimesCRSLost"`
Access	Read
Data type	Number (64 bit)

Late Collisions

Parameter	EthTransmitLateCollisions
Description	The number of collisions detected after the normal detection period.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat\ **Group**: NV_EthStat **Single**: EthTransmitLateCollisions
Usage example:	`nCLI Get "EthTransmitLateCollisions"`
Access	Read
Data type	Number (64 bit)

Group: General Networking Statistics

Successfully Transmitted Frames

Parameter	TransmitOK
Description	Specifies the number of frames transmitted without errors.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_ NetworkGenStat **Single**: TransmitOK
Usage example:	`nCLI Get "TransmitOK"`
Access	Read
Data type	Number (64 bit)

Successfully Received Frames

Parameter	ReceiveOK
Description	Specifies the number of frames that the network card has received without errors.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_NetworkGenStat **Single**: ReceiveOK
Usage example:	`nCLI Get "ReceiveOK"`
Access	Read
Data type	Number (64 bit)

Transmit Failures

Parameter	TransmitError
Description	Specifies the number of frames that failed to transmit.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_NetworkGenStat **Single**: TransmitError
Usage example:	`nCLI Get "TransmitError"`
Access	Read
Data type	Number (64 bit)

Receive Failures

Parameter	ReceiveError
Description	Specifies the number of frames that are received but not passed to the operating system because of errors.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_NetworkGenStat **Single**: ReceiveError
Usage example:	`nCLI Get "ReceiveError"`
Access	Read
Data type	Number (64 bit)

No Receive Buffers

Parameter	ReceiveNoBuffer
Description	The number of frames that are dropped because of lack of space for receive buffers.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_NetworkGenStat **Single**: ReceiveNoBuffer
Usage example:	`nCLI Get "ReceiveNoBuffer"`
Access	Read
Data type	Number (64 bit)

Direct Frames Received

Parameter	ReceiveFramesDirect
Description	The number of packets received without errors and addressed to the local Ethernet address.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_NetworkGenStat **Single**: ReceiveFramesDirect
Usage example:	`nCLI Get "ReceiveFramesDirect"`
Access	Read
Data type	Number (64 bit)

Multicast Frames Received

Parameter	ReceivedFramesMulticast
Description	Specifies the number of multicast frames received without errors.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_NetworkGenStat **Single**: ReceiveFramesMulticast
Usage example:	`nCLI Get "ReceiveFramesMulticast"`
Access	Read
Data type	Number (64 bit)

Broadcast Frames Received

Parameter	**ReceiveFramesBroadcast**
Description	Specifies the number of broadcast frames received without errors.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_EthStat **Group**: NV_NetworkGenStat **Single:** ReceiveFramesBroadcast
Usage example:	`nCLI Get "ReceiveFramesBroadcast"`
Access	Read
Data type	Number (64 bit)

Group: Alert Standard Format

ASF Support

Parameter	**ASFSupport**	
Description	Enables or disables the ASF (Alert Standard Format) feature. ASF is a industry specification that defines alerting capability in both operating system-present and operating system-absent environments.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASF **Single:** ASFSupport	
Usage example:	`nCLI Set "ASFSupport" "Disable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

ASF Destination IP Address

Parameter	**ASFDestIPAddr**
Description	Specifies the IP address of the managing station computer that is receiving the ASF alert frames. For ASF to be functional, the destination IP address must be specified.
Comment	Only the IPv4 (not IPv6) address is supported. **Note:** If ASFSupport is set to Disable, this parameter value is ignored.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASF **Single:** ASFDestIPAddr
Usage example:	`nCLI Set "ASFDestIPAddr" ""`
Access	ReadWrite
Data type	String
Maximum length	15

ASF Send Count

Parameter	**ASFSendCount**			
Description	Specifies the number of times an ASF alert will be sent out for a given event. If the value is more than one, the alert is sent at an interval of approximately 1 second. This is a global setting applied across all events.			
Comment	If ASFSupport is set to Disable, this parameter value is ignored.			
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASF **Single:** ASFSendCount			
Usage example:	`nCLI Set "ASFSendCount" "1"`			
Access	ReadWrite			
Data type	Selection			
User selection	0	1	2	3

Group: ASF Information

ASF Destination MAC Address

Parameter	**ASFDestMACAddr**
Description	Displays the MAC address of the managing station computer that is receiving the ASF alert frames.
Comment	If ASFSupport is set to Disable, this parameter value is ignored.
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventInfo **Single**: ASFDestNACAddr
Usage example:	**nCLI Get "ASFDestMACAddr"**
Access	Read
Data type	MAC Address

Group: System Fails to Boot Alert

System Fails to Boot Alert

Parameter	**ASFEventBootFailure**	
Description	This ASF alert is triggered when the operating system fails to start up.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventBootFailure **Single**: ASFEventootFailure	
Usage example:	**`nCLI Set "ASFEventBootFailure" "Disable"`**	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: Fan Problem Alert

Fan Problem Alert

Parameter	**ASFEventFanProblem**	
Description	This alert is triggered if the CPU fan is running at a low speed or has stopped, which can cause the CPU or system temperature to increase.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventFanProblem **Single**: ASFEventFanProblem	
Usage example:	`nCLI Set "ASFEventFanProblem" "Disable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: ASF SMBus Error

ASF SMBus Error

Parameter	**ASFEventSMBusError**	
Description	This alert packet is sent when there is a SMBus (System Management Bus) error. The SMBus is a two-wire interface through which the system can communicate with simple power-related chips.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventSMBusError **Single**: ASFEventSMBusError	
Usage example:	`nCLI Set "ASFEventSMBusError" "Disable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: ASF WOL Alert

ASF Wake On Lan (WOL) Aler

Parameter	**ASFEventWOL**	
Description	This alert is triggered when the system is wakened through the wake on LAN feature.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventWOL **Single**: ASFEventWOL	
Usage example:	`nCLI Set "ASFEventWOL" "Disable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: ASF Heartbeat Alert

ASF Heartbeat Alert Interval

Parameter	**ASFHeartbeatInterval**					
Description	Set the interval (in seconds) between ASF heartbeat alerts.					
Comment	If ASFSupport is set to Disable, this parameter value is ignored.					
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventHeartbeatInterval **Single**: ASFEventHeartbeatInterval					
Usage example:	`nCLI Set "ASFHeartbeatInterval" "10 seconds"`					
Access	ReadWrite					
Data type	Selection					
User selection	10 seconds 20 seconds	30 seconds 45 seconds	1 minute 2 minutes	3 minutes 5 minutes	7.5 minutes	10 minutes

Group: ASF Operating System Hung Alert

ASF Operating System Hung Alert

Parameter	**ASFEventOSHung**	
Description	This alert is triggered when the operating system is hung and the driver software or the operating system is not servicing the interrupts generated by the network interfaces.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventOSHung **Single**: ASFEventOSHung	
Usage example:	`nCLI Set "ASFEventOSHung" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: ASF Power Button Alert

ASF Power Button Alert

Parameter	**ASFEventPowerButton**	
Description	Enables or disables the power button alert. This alert is triggered each time the user presses the power button for shutting down or turning on the computer.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventPowerButton **Single**: ASFEventPowerButton	
Usage example:	`nCLI Set "ASFEventPowerButton" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: ASF System Hot Alert

ASF System Hot Alert

Parameter	**ASFEventSystemHot**	
Description	This alert is triggered when the temperature in the computer has exceeded a threshold limit.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventSystemHot **Single**: ASFEventSystemHot	
Usage example:	**`nCLI Set "ASFEventSystemHot" "Enable"`**	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: ASF CPU Overheated Alert

ASF CPU Overheat Alert

Parameter	**ASFEventCPUOverheated**	
Description	This alert is triggered when the temperature of the CPU exceeds a threshold.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventCPUOverheated **Single**: ASFEventCPUOverheated	
Usage example:	**`nCLI Set "ASFEventCPUOverheated" "Enable"`**	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: ASF CPU Overheated Alert

ASF CPU Hot Alert

Parameter	**ASFEventCPUHot**	
Description	This alert is triggered when the fan in the CPU is not functioning or the CPU temperature is increasing.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventCPUHot **Single**: ASFEventCPUHot	
Usage example:	`nCLI Set "ASFEventCPUHot" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: ASF Case Intrusion Alert

ASF Case Intrusion Alert

Parameter	**ASFEventCaseIntrusion**	
Description	This alert is triggered when the computer's case is opened.	
Comment	If ASFSupport is set to Disable, this parameter value is ignored.	
Hierarchy	**Namespace**: NS_Eth **Namespace**: NS_ASF **Group**: NV_ASFEventCaseIntrusion **Single**: ASFEventCaseIntrusion	
Usage example:	`nCLI Set "ASFEventCaseIntrusion" "Disable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

APPENDIX

B

NVIDIA FIREWALL PARAMETERS REFERENCE

Note: For references to all the individual parameters, categorized by group, see the entries listed for this appendix — **B. NVIDIA Firewall Parameters: Reference** — in the "Table of Contents" on page iii.

Group: Configure Firewall Security Level

Configure Firewall Security Level

Parameter	**FwlProfiles**							
Description	Selects a default security level or configure a custom security level, which is a set of rules that determines the policy that the firewall follows.							
Comment	This parameter is not supported through WMI script. For CLI user who wants to customize the firewall settings and not use a pre-defined profile, change the firewall security level to one of the custom levels: **Note:** For details on the settings, see the next section.							
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlProfiles **Single**: FwlProfiles							
Usage example	`nCLI Set "FwlProfiles" "Medium"`							
Access	ReadWrite							
Data type	Selection							
User selection	Off	Low	Medium	High	Lockdown	Custom1	Custom2	Custom3

About the FwlProfiles Settings

The **FwlProfiles** parameter is supported through the WMI scripting language. For CLI user who wants to customize the firewall settings and not used a pre-defined profile, change the firewall security level to one of the custom levels described below.

Lockdown blocks all traffic, both in and out, except locally generated ASF alerts.

The **High** setting has the following features and functionality:

- Allows the least amount of traffic through.
- Only outbound connections may be established. Inbound connections are not allowed. Inbound traffic is allowed only if it is in response to an outbound packet that was seen previously on a valid connection.
- Encompasses what is commonly known as "stealth mode" in which the station cannot be "pinged" and is not permitted to generate any ICMP error messages, except where necessary to permit normal operation.
- Allows VPNs, including those based on IPsec (requiring AH, ESP, L2TP, IKE, UDP port 500), as well as those that rely on **point-to-point punneling protocol (PPTP)**, which uses generic routing encapsulation (GRE).
- Restricts traffic by prohibiting IP and/or TCP options that might be misused, as well as by preventing the spoofing of IP source addresses (for both IPv4 and IPv6).

The **Medium** setting has the following features and functionality:

- Medium is the factory "default" setting when the firewall is enabled.
- Does not have the "stealth" features associated with the High setting and therefore allows most (but not all) ICMP error messages to be sent and received.
- Blocks most incoming connections with the "default action" of **Deny**. In order to allow file transfers through MSN Messenger and Yahoo! Messenger, incoming connections to port 80 must be allowed.

  Note: These applications *will not work* with the High setting.
- Allows dynamic ports to be opened up from the inside only:

  Default *in*: Deny

  Default *out*: Allow
- Supports outgoing NetMeeting calls.
- As with the High setting, allows VPNs based on both IPsec and on PPTP.

- Restricts traffic by prohibiting IP and/or TCP options that might be misused, as well as by preventing the spoofing of IP source addresses (for both IPv4 and IPv6).

The **Low** settings allows “safe” incoming connections, denying those that are known to be dangerous and defaulting to *allow* TCP or UDP connections for which a rule has not been specified. The *Low* settings has the following features and functionality:

- Allows mostly all ICMP traffic, except for sending router-oriented (e.g., router advertisement) or deprecated (e.g., source quench) (Type, Code) pairs.
- Allows bi-directional dynamic ports to be opened.

  Default *in*: Allow

  Default *out*: Allow

  Thus, the Low setting will support the NetMeeting application in either direction.
- As with the High and Medium settings, also allows VPNs, based on both IPsec and PPTP.
- As with the High and Medium settings, also restricts traffic by prohibiting IP and/or TCP options that might be misused as well as by preventing the spoofing of IP source addresses (for both IPv4 and IPv6).

Off is the most permissive setting in that it allows all traffic, both in and out.

Group: Configure Professional Firewall Options

Disallow Promiscuous Mode

Parameter	FwlPromiscuous	
Description	When this parameter is enabled, the firewall prevents applications from setting the NVIDIA network interface to promiscuous mode. Promiscuous mode is primarily used by packet sniffing software.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlOptions **Single**: FwlPromiscuous	
Usage example:	`nCLI Set "FwlPromiscuous" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Enable	Disable

Disallow DHCP Server

Parameter	FwlDHCPServer	
Description	When this option is enabled, the firewall prevents a **DHCP (Dynamic Host Configuration Protocol)** server process in the computer from using the NVIDIA network interface to communicate using the DHCP protocol. The DHCP server is used to assign IP addresses to client computers.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlOptions **Single**: FwlDHCPServer	
Usage example:	`nCLI Set "FwlDHCPServer" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Block Outbound Spoofed IP Packets

Parameter	FwlAntiIPSpoofing	
Description	When this parameter is enabled, the firewall blocks any application on the NVIDIA network interface from sending network traffic using an IP address different than the one assigned to the interface. Such network packets are called spoofed IP packets, and this feature, also known as "anti-IP-spoofing," is intended to prevent the NVIDIA network interface from participating in distributed denial of service attacks.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlOptions **Single**: FwlAntiIPSpoofing	
Usage example:	`nCLI Set "FwlAntiIPSpoofing" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Block Spoofed ARP Packets

Parameter	FwlAntiARPSpoofing	
Description	When this parameter is enabled, the firewall filters out any ARP packets sent by an offending computer (i.e, a computer that pretends to be another computer by altering the local ARP cache). Such network packets are called spoofed ARP packets and this feature is also known as "anti-ARP-spoofing".	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlOptions **Single**: FwlAntiARPSpoofing	
Usage example:	`nCLI Set "FwlAntiARPSpoofing" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Block UDPv4 with No UDP Checksum

Parameter	**FwlChecksumUDP**	
Description	When this parameter is enabled, the firewall drops any UDP datagram that has no UDP checksum if it is inside an IPv4 packet (UDP checksums are optional when used over IPv4, but are mandatory when used over IPv6).	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlOptions **Single**: FwlChecksumUDP	
Usage example:	`nCLI Set "FwlChecksumUDP" "Enable"`	
Access	ReadWrite	
Data type	Selection	
User selection	Disable	Enable

Group: EtherType Default Rule

EtherType Default Rule

Parameter	**FwlEtherTypeDefault**	
Description	This rule is applied when a packet contains an EtherType that does not match any rule in the EtherType rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlEtherTypeDefault **Single**: FwlEtherTypeDefault	
Usage example:	`nCLI Set "FwlEtherTypeDefault" "Deny"`	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Group: IP Address/Mask Default Rule

IP Address/Mask Default Action

Parameter	**FwlIPDefault**	
Description	This action is applied when a packet contains an IP address/mask that does not match any rule in the IP rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlIPDefault **Single**: FwlIPDefault	
Usage example:	`nCLI Set "FwlIPDefault" "Allow"`	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Group: Domain Name Default Rule

Domain Name Default Rule

Parameter	**FwlDomainDefault**	
Description	This rule is applied when a DNS packet contains a domain name that does not match any rule in the domain name rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlDomainDefault **Single**: FwlDomainDefault	
Usage example:	`nCLI Set "FwlDomainDefault" "Allow"`	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Group: IP Option Default Rule

Inbound IP Option Default Rule

Parameter	**FwlIPOptionDefaultIn**	
Description	This rule is applied when an inbound packet contains an IP option that does not match any rule in the IP option rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlIPOptionDefault **Single**: FwlIPOptionDefaultIn	
Usage example:	**`nCLI Set "FwlIPOptionDefaultIn" "Deny"`**	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Outbound IP Option Default Rule

Parameter	**FwlIPOptionDefaultOut**	
Description	This rule is applied when an outbound packet contains an IP option that does not match any rule in the IP option rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlIPOptionDefault **Single**: FwlIPOptionDefaultOut	
Usage example:	**`nCLI Set "FwlIPOptionDefaultOut" "Deny"`**	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Group: IP Protocol Default Rule

IP Protocol Default Rule

Parameter	**FwlIPProtocolDefault**	
Description	This rule is applied when a packet contains an IP protocol that does not match any rule in the IP protocol rule table	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlIPProtocolDefault **Single**: FwlIPProtocolDefault	
Usage example:	`nCLI Set "FwlIPProtocolDefault" "Deny"`	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Group: Port Number Default Rule

Inbound Port Number Default Rule

Parameter	**FwlPortDefaultIn**	
Description	This rule is applied when an inbound packet contains a UDP or TCP port that does not match any rule in the Port rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlPortDefault **Single**: FwlPortDefaultIn	
Usage example	`nCLI Set "FwlPortDefaultIn" "Deny"`	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Outbound Port Number Default Rule

Parameter	**FwlPortDefaultOut**	
Description	This rule is applied when an outbound packet contains a UDP or TCP port that does not match any rule in the Port rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlPortDefault **Single**: FwlPortDefaultOut	
Usage example:	`nCLI Set "FwlPortDefaultOut" "Allow"`	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Group: TCP Options Default Rule

TCP Options Default Rule

Parameter	**FwlTCPOptionDefault**	
Description	This rule is applied when a packet contains a TCP option that does not match any rule in the TCP option rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlTCPOptionDefault **Single**: FwlTCPOptionDefault	
Usage example:	`nCLI Set "FwlTCPOptionDefault" "Deny"`	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Group: ICMP Messages Default Rule

Inbound ICMP Default Rule

Parameter	**FwlICMPDefaultIn**	
Description	This rule is applied when an inbound packet contains an ICMP type/code pair that does not match any rule in the ICMP rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlICMPDefault **Single**: FwlICMPDefaultIn	
Usage example:	`nCLI Set "FwlICMPDefaultIn" "Deny"`	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Outbound ICMP Default Rule

Parameter	**FwlICMPDefaultOut**	
Description	This rule is applied when an outbound packet contains an ICMP type/code pair that does not match any rule in the ICMP rule table.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlICMPDefault **Single**: FwlICMPDefaultOut	
Usage example:	`nCLI Set "FwlICMPDefaultOut" "Deny"`	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Group: Clear Firewall Statistics

Clear Firewall Statistics

Parameter	**FwlStatClearAll**
Description	Clears all firewall statistics.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStatClear **Single**: FwlStatClearAll
Usage example:	`nCLI Set "FwlStatClearAll" "Clear"`
Access	ReadWrite
Data type	Selection
User selection	Clear

Group: Firewall Statistics

Allowed Inbound UDP Datagrams

Parameter	**FwlStatUDPInPktsAllowed**
Description	Specifies the number of inbound UDP datagrams allowed by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatUDPInPktsAllowed
Usage example:	`nCLI Get "FwlStatUDPInPktsAllowed"`
Access	Read
Data type	Number (64 bit)

Denied Inbound UDP Datagrams I

Parameter	**FwlStatUDPInPktsDenied**
Description	Number of inbound UDP datagrams denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatUDPInPktsDenied
Usage example	`nCLI Get "FwlStatUDPInPktsDenied"`
Access	Read
Data type	Number (64 bit)

Allowed Outbound UDP Datagrams

Parameter	**FwlStatUDPOutPktsAllowed**
Description	Number of outbound UDP datagrams allowed by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatUDPOutPktsAllowed
Usage example:	`nCLI Get "FwlStatUDPOutPktsAllowed"`
Access	Read
Data type	Number (64 bit)

Denied Outbound UDP Datagrams

Parameter	**FwlStatUDPOutPktsDenied**
Description	Number of outbound UDP datagrams denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatUDPOutPktsDenied
Usage example:	`nCLI Get "FwlStatUDPOutPktsDenied"`
Access	Read
Data type	Number (64 bit)

Denied Inbound UDP Connections

Parameter	FwlStatUDPInConnectionsDenied
Description	Number of inbound UDP connections denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatUDPInConnectionsDenied
Usage example:	`nCLI Get "FwlStatUDPInConnectionsDenied"`
Access	Read
Data type	Number (64 bit)

Allowed Outbound UDP Connections

Parameter	FwlStatUDPOutConnectionsAllowed
Description	Number of outbound UDP connections allowed by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatUDPOutConnectionsAllowed
Usage example:	`nCLI Get "FwlStatUDPOutConnectionsAllowed"`
Access	Read
Data type	Number (64 bit)

Denied Outbound UDP Connections

Parameter	FwlStatUDPOutConnectionsDenied
Description	Number of outbound UDP connections denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatUDPOutConnectionsDenied
Usage example:	`nCLI Get "FwlStatUDPOutConnectionsDenied"`
Access	Read
Data type	Number (64 bit)

Allowed Inbound TCP Segments

Parameter	**FwlStatTCPInPktsAllowed**
Description	Number of inbound TCP segments allowed by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatTCPInPktsAllowed
Usage example:	`nCLI Get "FwlStatTCPInPktsAllowed"`
Access	Read
Data type	Number (64 bit)

Denied Inbound TCP Segments

Parameter	**FwlStatTCPInPktsDenied**
Description	Number of inbound TCP segments denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Namespace**: NS_Firewall **Single**: FwlStatTCPInPktsDenied
Usage example:	`nCLI Get "FwlStatTCPInPktsDenied"`
Access	Read
Data type	Number (64 bit)

Allowed Outbound TCP Segments

Parameter	**FwlStatTCPOutPktsAllowed**
Description	Number of outbound TCP segments allowed by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatTCPOutPktsAllowed
Usage example:	`nCLI Get "FwlStatTCPOutPktsAllowed"`
Access	Read
Data type	Number (64 bit)

Denied Outbound TCP Segments

Parameter	**FwlStatTCPOutPktsDenied**
Description	Number of outbound TCP segments denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatTCPOutPktsDenied
Usage example:	`nCLI Get "FwlStatTCPOutPktsDenied"`
Access	Read
Data type	Number (64 bit)

Allowed Inbound TCP Connections

Parameter	**FwlStatTCPInConnectionsAllowed**
Description	Number of inbound TCP connections allowed by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatTCPInConnectionsAllowed
Usage example:	`nCLI Get "FwlStatTCPInConnectionsAllowed"`
Access	Read
Data type	Number (64 bit)

Denied Inbound TCP Connections

Parameter	**FwlStatTCPInConnectionsDenied**
Description	Number of inbound TCP connections denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatTCPInConnectionsDenied
Usage example:	`nCLI Get "FwlStatTCPInConnectionsDenied"`
Access	Read
Data type	Number (64 bit)

Allowed Outbound TCP Connections

Parameter	FwlStatTCPOutConnectionsAllowed
Description	Number of outbound TCP connections allowed by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatTCPOutConnectionsAllowed
Usage example:	`nCLI Get "FwlStatTCPOutConnectionsAllowed"`
Access	Read
Data type	Number (64 bit)

Denied Outbound TCP Connections

Parameter	FwlStatTCPOutConnectionsDenied
Description	Number of outbound TCP connections denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatTCPOutConnectionsDenied
Usage example:	`nCLI Get "FwlStatTCPOutConnectionsDenied"`
Access	Read
Data type	Number (64 bit)

Allowed Inbound ICMP Packets

Parameter	FwlStatICMPInPktsAllowed
Description	Number of inbound ICMP packets allowed by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatICMPInPktsAllowed
Usage example:	`nCLI Get "FwlStatICMPInPktsAllowed"`
Access	Read
Data type	Number (64 bit)

Denied Inbound ICMP Packets

Parameter	FwlStatICMPInPktsDenied
Description	Number of inbound ICMP packets denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatICMPInPktsDenied
Usage example:	`nCLI Get "FwlStatICMPInPktsDenied"`
Access	Read
Data type	Number (64 bit)

Allowed Outbound ICMP Packets

Parameter	FwlStatICMPOutPktsAllowed
Description	Number of outbound ICMP packets allowed by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatICMPOutPktsAllowed
Usage example:	`nCLI Get "FwlStatICMPOutPktsAllowed"`
Access	Read
Data type	Number (64 bit)

Denied Outbound ICMP Packets

Parameter	FwlStatICMPOutPktsDenied
Description	Specifies the number of outbound ICMP packets denied by the firewall.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatICMPOutPktsDenied
Usage example:	`nCLI Get "FwlStatICMPOutPktsDenied"`
Access	Read
Data type	Number (64 bit)

Other Allowed Inbound Packets

Parameter	FwlStatOtherInPktsAllowed
Description	Specifies the number of inbound packets allowed by the firewall that are not UDP, TCP, or ICMP.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatOtherInPktsAllowed
Usage example:	`nCLI Get "FwlStatOtherInPktsAllowed"`
Access	Read
Data type	Number (64 bit)

Other Denied Inbound Packets

Parameter	FwlStatOtherInPktsDenied
Description	Number of inbound packets denied by the firewall that are not UDP, TCP, or ICMP.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatOtherInPktsDenied
Usage example:	`nCLI Get "FwlStatOtherInPktsDenied"`
Access	Read
Data type	Number (64 bit)

Other Allowed Outbound Packets

Parameter	FwlStatOtherOutPktsAllowed
Description	Number of outbound packets allowed by the firewall that are not UDP, TCP, or ICMP.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatOtherOutPktsAllowed
Usage example:	`nCLI Get "FwlStatOtherOutPktsAllowed"`
Access	Read
Data type	Number (64 bit)

Other Denied Outbound Packets

Parameter	**FwlStatOtherOutPktsDenied**
Description	Specifies the number of outbound packets denied by the firewall that are not UDP, TCP, or ICMP.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlStat **Single**: FwlStatOtherOutPktsDenied
Usage example:	`nCLI Get "FwlStatOtherOutPktsDenied"`
Access	Read
Data type	Number (64 bit)

Group: Factory Default

Factory Default

Parameter	**FwlDefault**	
Description	Specifies to restore all firewall settings to the factory default.	
Comment	This parameter is not supported through WMI scripting.	
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_Fwl_Default **Single**: FwlDefault	
Usage example:	`nCLI Set "FwlDefault" "NoRestore"`	
Access	ReadWrite	
Data type	Selection	
User selection	NoRestore	Restore

Group: Flush DNS Cache

Flush DNS Cache

Parameter	FwlFlushDNS
Description	Specifies to flush the operating system DNS cache.
Comment	DNS cache needs to be flushed when Firewall Domain Name configuration is changed.
Hierarchy	**Namespace**: NS_Firewall **Group**: NV_FwlFlushDNS **Single**: FwlFlushDNS
Usage example	`nCLI Set "FwlFlushDNS" "Clear"`
Access	ReadWrite
Data type	Selection
User selection	Clear

Table: EtherType Rules

Table parameter	NV_FwlEtherType		
Description	Specifies table to configure EtherType firewall rules. As part of the Ethernet header, the EtherType is used to identify the type of Ethernet payload. Example payloads include IPv4, AppleTalk, IPX, and NetBEUI.		
Comment	For EtherType that does not match any rule in the table, default setting in FwlEtherTypeDefault will be used.		
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlEtherType		

Usage example	`nCLI AddRow "NV_FwlEtherType" "EtherType=2048,EtherTypeName=Internet Protocol version 4 (IPv4) (RFC 791),EtherTypeAction=Allow"`
	`_____`
	`nCLI EditRow "NV_FwlEtherType.EtherType=2048" "EtherTypeName=Address Resolution Protocol (ARP) (RFC 826),EtherTypeAction=Allow"`
	`_____`
	`nCLI DelRow "NV_FwlEtherType.EtherType=2048"`		

Access	ReadWrite		
Single parameters	EtherType	EtherTypeName	EtherTypeAction

Ether Type

Parameter	EtherType
Description	The EtherType identifies the type of Ethernet payload. Some examples and their hexadecimal values include IPv4 (0x0800), AppleTalk (0x809B), IPX (0x8137) and NetBEUI (0x8191).
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlEtherType **Single**: EtherType
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	Number (32 bit)
Maximum value	65535
Minimum value	1501

EtherType Name

Parameter	EtherTypeName
Description	Name associated with the EtherType.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlEtherType **Single**: EtherTypeName
Access	ReadWrite
Data type	String
Maximum Length	60

EtherType Action

Parameter	EtherTypeAction	
Description	Specifies action for the EtherType.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlEtherType **Single**: EtherTypeAction	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Table: IP Address/Mask Rule

Table parameter	**NV_FwlIP**				
Description	Specifies table to configure firewall rules based on IP addresses/masks.				
Comment	For IP address/mask pair that does not match any rule in the table, default setting in FwlIPDefault will be used.				
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIP				
Usage example	`nCLI AddRow "NV_FwlIP" "IPRemoteIP=0000:0000:0000:0000:0000:FFFF:0000:0000,IPRemoteIPMask=32,IPAction=Allow"` `nCLI DelRow "NV_FwlIP.IPRemoteIP='0000:0000:0000:0000:0000:FFFF:0000:0000',IPRemoteIPMask='32'"`				
Access	ReadWrite				
Single parameter	IPRemoteIP	IPRemoteIPMask	IPLocalIP	IPLocalIPMask	IPAction

Remote IP Address

Parameter	**IPRemoteIP**
Description	IP address of the remote machine or subnet.
Tree	**Namespace:** NS_Firewall **Table:** NV_FwlIP **Single:** IPRemoteIP
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	IP Address

Remote IP Address Mask

Parameter	**IPRemoteIPMask**
Description	IP address mask of the remote machine or subnet.
Tree	**Namespace**: NS_Firewall **Table**: NV_FwlIP **Single**: IPRemoteIPMask
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	IP Mask Length

IP Action

Parameter	**IPAction**	
Description	Specifies the action for network traffic.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIP **Single**: IPAction	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Table: Domain Names Rule

Table parameter:	**NV_FwlDomain**	
Description	Specifies the table to configure domain name rules. Domain name is a user-friendly name used to identify a Web site; for example, **www.nvidia.com**. The firewall blocks DNS lookups of domain names. You can bypass this filter by directly entering an IP address (if the IP address is known) instead of a domain name to access a Web site.	
Comment	CLI users need to flush DNS cache for domain name rules to take effect. To flush DNS cache, set FwlFlushDNS. For a given domain name that does not match any rule in the table, the default setting in FwlDomainDefault will be used.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlDomain	
Usage example:	**nCLI AddRow "NV_FwlDomain" "DomainName=www.dummy.com,DomainAction=Deny"** **nCLI EditRow "NV_FwlDomain.DomainName='www.dummy.com'" "DomainAction=Deny"** **nCLI DelRow "NV_FwlDomain.DomainName='www.dummy.com'"**	
Access	ReadWrite	
Single Parameter	DomainName DomainAction	DomainLocalIP DomainLocalIPMask

Domain Name

Parameter	**DomainName**
Description	Domain name of the computer or Web site
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlDomain **Single**: DomainName
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	String
Maximum Length	127

Domain Action

Parameter	DomainAction	
Description	Specifies action for network traffic.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlDomain **Single**: DomainAction	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Table: IP Option Rules

Table parameter	NV_FwlIPOption		
Description	Specifies the table to configure IP option rules. IPv4 options are added to the basic IPv4 header to provide additional features beyond those that are supported by the standard IPv4 packet's header. The standard 20-byte IPv4 header can be expanded to have up to 40 bytes of options. IPv6 options have no fixed size, but are otherwise similar to IPv4 options and provide for many of the same features.		
Comment	For an IP option that does not match any rule in the table, the default setting in FwlIPOptionDefault will be used.		
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPOption		

Usage example	`nCLI AddRow "NV_FwlIPOption" "IPOptionNumber=0,IPOptionName=End of Option List,IPOptionVersion=IPv4,IPOptionActionIn=Allow,IPOptionActionOut=Allow"`
	`nCLI EditRow "NV_FwlIPOption.IPOptionNumber=0,IPOptionVersion=4" "IPOptionName=Pad-1 (i.e., one octet of padding),IPOptionActionIn=Allow,IPOptionActionOut=Allow"`
	`nCLI DelRow "NV_FwlIPOption.IPOptionNumber=0,IPOptionVersion=4"`		

Access	ReadWrite		
Single parameter	IPOptionNumber IPOptionName	IPOptionVersion IPOptionActionIn	IPOptionActionOut

IP Option Number

Parameter	IPOptionNumber	
Description	IP option number. IPv4 options are added to the basic IPv4 header to provide additional features beyond those that are supported by the standard IPv4 packet's header. The standard 20-byte IPv4 header can be expanded to have up to 40 bytes of options. IPv6 options have no fixed size, but are otherwise similar to IPv4 options and provide for many of the same features.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPOption **Single**: IPOptionNumber	
Access	ReadWrite	
Table key	This parameter is a key to the table	
Data type	Number (32 bit)	
Maximum Value	255	**Minimum Value:** 0

IP Option Name

Parameter	IPOptionName
Description	Specifies name associated with the IP option number.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPOption **Single**: IPOptionName
Access	ReadWrite
Data type	String
Maximum Length	60

IP Version

Parameter	**IPOptionVersion**	
Description	Specifies whether rule is for IPv4 or IPv6.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPOption **Single**: IPOptionVersion	
Access	ReadWrite	
Table key	This parameter is a key to the table	
Data type	Selection	
User selection	IPv4	IPv6

IP Inbound Action

Parameter	**IPOptionActionIn**	
Description	Specifies action for inbound network traffic.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPOption **Single**: IPOptionActionIn	
Access	ReadWrite	
Data type	Selection	
User selection	Allow	Deny

IP Outbound Action

Parameter	**IPOptionActionOut**	
Description	Specifies action for outbound network traffic.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPOption **Single**: IPOptionActionOut	
Access	ReadWrite	
Data type	Selection	
User selection	Allow	Deny

Table: IP Protocol Rule

Table parameter	**NV_FwlIPProtocol**		
Description	Specifies table to configure IP protocol rules. IP protocol identifies the type of IP payload. ICMP, TCP and UDP are examples of common IP payloads.		
Comment	For an IP protocol that does not match any rule in the table, the default setting in FwlIPProtocolDefault will be used.		
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPProtocol		
Usage example:	**nCLI AddRow "NV_FwlIPProtocol" "IPProtocol=1,IPProtocolName=Internet Control Message Protocol for IPv4 (ICMP),IPProtocolAction=Allow"** **———————————————————** **nCLI EditRow "NV_FwlIPProtocol.IPProtocol=1" "IPProtocolName=Internet Group Management Protocol for IPv4 (IGMP),IPProtocolAction=Allow"** **———————————————————** **nCLI DelRow "NV_FwlIPProtocol.IPProtocol=1"**		
Access	ReadWrite		
Single Parameters	IPProtocol	IPProtocolName	IPProtocolAction

IP Protocol

Parameter	**IPProtocol**
Description	Specifies the IP protocol number. IP protocol identifies the type of IP payload. Common protocols and their decimal values include ICMP (1), TCP (6), and UDP (17).
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPProtocol **Single**: IPProtocol
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	Number (32 bit)
Maximum Value	255
Minimum Value	0

IP Protocol Name

Parameter	IPProtocolName
Description	Specifies a name for an IP protocol.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPProtocol **Single**: IPProtocolName
Access	ReadWrite
Data type	String
Maximum Length	60

IP Protocol Action

Parameter	IPProtocolAction	
Description	Specifies the action for network traffic.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlIPProtocol **Single**: IPProtocolAction	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Table: TCP/UDP Port Rule

Parameter name	**NV_FwlPort**			
Description	Specifies the table to configure TCP or UDP port rules. Port numbers are used by TCP or UDP to identify sending and receiving applications. Some common ports include HTTP (80), TELNET (23) and SMTP (25).			
Comment	For a TCP/UDP port that does not match any rule in the table, the default setting in FwlPortDefault will be used.			
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlPort			
Usage examples	**nCLI AddRow "NV_FwlPort" "PortActionIn=Deny,PortActionOut=Deny,PortRemoteIP=0000:0000:0000:0000:0000:FFFF:0000:0000,PortRemoteIPMask=32,PortName=Reserved,PortRangeBegin=0,PortRangeEnd=0,PortProtocol=Both"** **nCLI EditRow "NV_FwlPort.PortRemoteIP='0000:0000:0000:0000:0000:FFFF:0000:0000',PortRemoteIPMask='32',PortRangeBegin=0,PortRangeEnd=0,PortProtocol=0" "PortActionIn=Deny,PortActionOut=Allow,PortName=Time (RFC 868)"** **nCLI DelRow "NV_FwlPort.PortRemoteIP='0000:0000:0000:0000:0000:FFFF:0000:0000',PortRemoteIPMask='32',PortRangeBegin=0,PortRangeEnd=0,PortProtocol=0"**			
Access	ReadWrite			
Single parameter	PortActionIn PortActionOut PortRemoteIP	PortRemoteIPMask PortLocalIP PortRangeBegin	PortRangeEnd PortLocalIPMask	PortName PortProtocol

TCP/UDP Port Outbound Action

Parameter	PortActionOut	
Description	Specifies outbound action for the network connection.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlPort **Single**: PortActionOut	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Remote IP Address

Parameter	PortRemoteIP
Description	IP address of the remote machine or subnet.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlPort **Single**: PortRemoteIP
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	IP Address

Remote IP Subnet Mask

Parameter	PortRemoteIPMask
Description	Specifies the IP address mask of the remote machine or subnet.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlPort **Single**: PortRemoteIPMask
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	IP Mask Length

Port Name

Parameter	PortName
Description	Specifies thename associated with the TCP or UDP port range.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlPort **Single**: PortName
Access	ReadWrite
Data type	String
Maximum Length	100

Beginning Port Number

Parameter	PortRangeBegin
Description	Specifies the first UDP or TCP port in the range.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlPort **Single**: PortRangeBegin
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	Number (32 bit)
Maximum Value	65535

Ending Port Number

Parameter	PortRangeEnd
Description	Specifies the last UDP or TCP port in the range.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlPort **Single:** PortRangeEnd
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	Number (32 bit)
Maximum Value	65535

Port Protocol

Parameter	PortProtocol	
Description	Specifies whether the port protocol is UDP, TCP, or both.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlPort **Single**: PortProtocol	
Access	ReadWrite	
Table key	This parameter is a key to the table	
Data type	Selection	
User selection	UDP	TCP

Table: TCP Options Rule

Table parameter	NV_FwlTCPOption		
Description	Specifies the table to configure the TCP options rule. TCP options are added to the standard 20-byte TCP header to provide additional features that typically can only be used if they are negotiated at the beginning of a TCP connection.		
Comment	For a given TCP option that does not match any rule in the table, the default setting in FwlTCPOptionDefault will be used.		
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlTCPOption		

Usage examples:	`nCLI AddRow "NV_FwlTCPOption" "TCPOptionNumber=0,TCPOptionName=End of Option List (RFC 793),TCPOptionAction=Allow"`
	`nCLI EditRow "NV_FwlTCPOption.TCPOptionNumber=0" "TCPOptionName=No Operation (RFC 793),TCPOptionAction=Allow"`
	`nCLI DelRow "NV_FwlTCPOption.TCPOptionNumber=0"`		

Access	ReadWrite		
Single parameters	TCPOptionNumber	TCPOptionName	TCPOptionAction

TCP Option Number

Parameter	**TCPOptionNumber**	
Description	Represents the TCP option number. TCP options are added to the standard 20-byte TCP header to provide additional features that typically can only be used if they are negotiated at the beginning of a TCP connection.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlTCPOption **Single**: TCPOptionNumber	
Access	ReadWrite	
Table key	This parameter is a key to the table	
Data type	Number (32 bit)	
Maximum Value	255	**Minimum Value:** 0

TCP Option Name I

Parameter	**TCPOptionName**
Description	Specifies a name associated with a TCP option number.
Hierarchy	**Single**: TCPOptionName **Namespace**: NS_Firewall **Table:** NV_FwlTCPOption **Single**: TCPOptionName
Access	ReadWrite
Data type	String
Maximum Length	60

TCP Option Action

Parameter	**TCPOptionAction**	
Description	Specifies the action for network traffic containing a given TCP option number.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlTCPOption **Single**: TCPOptionAction	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

Table: ICMP Rules

Table parameter	**NV_FwlICMP**			
Description	Specifies the table to configure ICMP message rules. ICMP communicates error, diagnostic and control messages. Examples of ICMP messages include echo (i.e., ping) and 'destination unreachable'.			
Comment	For an ICMP message that does not match any rule in the table, the default setting in FwlICMPDefault will be used.			
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlICMP			

Usage examples:	`nCLI AddRow "NV_FwlICMP" "ICMPRemoteIP=0000:0000:0000:0000:0000:FFFF:0000:0000,ICMPRemoteIPMask=32,ICMPType=0,ICMPCode=0,ICMPName=Echo reply (RFC792),ICMPVersion=ICMPv4,ICMPActionIn=Allow,ICMPActionOut=Allow"`
	`————————————————————————`
	`nCLI EditRow "NV_FwlICMP.ICMPRemoteIP='0000:0000:0000:0000:0000:FFFF:0000:0000',ICMPRemoteIPMask='32',ICMPType=0,ICMPCode=0,ICMPVersion=4" "ICMPName=Not assigned,ICMPActionIn=Deny,ICMPActionOut=Deny`
	`————————————————————————`
	`nCLI DelRow "NV_FwlICMP.ICMPRemoteIP='0000:0000:0000:0000:0000:FFFF:0000:0000',ICMPRemoteIPMask='32',ICMPType=0,ICMPCode=0,ICMPVersion=4"`			

Access	ReadWrite			
Single Parameters	ICMPRemoteIP ICMPRemoteIPMask ICMPLocalIP	ICMPLocalIPMask ICMPType ICMPCode	ICMPName ICMPVersion	ICMPActionIn ICMPActionOut

Remote IP Address

Parameter	**ICMPRemoteIP**
Description	Specifies the IP address of the remote machine or subnet.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlICMP **Single**: ICMPRemoteIP
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	IP Address

Remote IP Subnet Mask

Parameter	**ICMPRemoteIPMask**
Description	Specifies the IP address mask of the remote machine or subnet.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlICMP **Single**: ICMPRemoteIPMask
Access	ReadWrite
Table key	This parameter is a key to the table
Data type	IP Mask Length

ICMP Type

Parameter	**ICMPType**	
Description	Specifies the ICMP type	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlICMP **Single**: ICMPType	
Access	ReadWrite	
Table key	This parameter is a key to the table	
Data type	Number (32 bit)	
Maximum value	255	**Minimum Value:** 0

ICMP Code

Parameter	**ICMPCode**	
Description	Specifies the ICMP code.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlICMP **Single**: ICMPCode	
Access	ReadWrite	
Table key	This parameter is a key to the table	
Data type	Number (32 bit)	
Maximum value	255	**Minimum Value:** 0

ICMP Name

Parameter	ICMPName
Description	Specifies a name for the ICMP type/code pair.
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlICMP **Single**: ICMPName
Access	ReadWrite
Data type	String
Maximum Length	120

ICMP Version

Parameter	ICMPVersion	
Description	Specifies whether the rule is for ICMPv4 or ICMPv6.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlICMP **Single**: ICMPVersion	
Access	ReadWrite	
Table key	This parameter is a key to the table	
Data type	Selection	
User selection	ICMPv4	ICMPv6

ICMP Inbound Action

Parameter	ICMPActionIn	
Description	Specifies the action for inbound network traffic.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlICMP **Single**: ICMPActionIn	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

ICMP Outbound Action

Parameter	ICMPActionOut	
Description	Specifies the action for outbound network traffic.	
Hierarchy	**Namespace**: NS_Firewall **Table:** NV_FwlICMP **Single**: ICMPActionOut	
Access	ReadWrite	
Data type	Selection	
User selection	Deny	Allow

APPENDIX

GLOSSARY

- **distinguished name**. In reference to the ForceWare Network Access Manager application, a distinguished name is the name that uniquely identifies a parameter. Each parameter has a distinguished name.
- **group parameter**. In reference to the ForceWare Network Access Manager application, a group parameter is a collection of single parameters that belong to a functionality set.
- **ICMP (Internet Control Message Protocol)** is a message control and error-reporting protocol between a host server and a gateway to the Internet. ICMP uses IP datagrams, but the messages are processed by the IP software and are not necessarily directly apparent to the application user.
- **IP (Internet Protocol).** IP is the method or protocol by which data is sent from one computer to another on the Internet. Each computer (known as a host) on the Internet has at least one IP address that uniquely identifies it from all other computers on the Internet. When you send or receive data (for example, an e-mail note or a Web page), the message gets divided into little chunks called packets. Each of these packets contains the sender's Internet address and the receiver's Internet address.

  When the sender needs to send a packet to a receiver on a different subnetwork, the packet is sent first to a to the sender's "default gateway" computer that understands a small part of the Internet. The gateway computer reads the destination address and forwards the packet to an adjacent gateway that in turn reads the destination address and so forth across the Internet until one gateway recognizes the packet as belonging to a computer within its immediate neighborhood or domain. That gateway then forwards the packet directly to the computer whose address is specified.

Because a message is divided into a number of packets, each packet can, if necessary, be sent by a different route across the Internet. Packets can arrive in a different order than the order in which they were sent. The Internet Protocol just delivers them. For applications requiring in-order delivery, it's up to a higher-layer protocol to ensure proper sequencing across a packet stream.

IP is a connectionless protocol, which means that there is no continuing connection between the end points that are communicating. Each packet that travels through the Internet is treated as an independent unit of data without any relation to any other unit of data.In the Open Systems Interconnection (OSI) communication model, IP is in layer 3, the Networking Layer.

The most widely used version of IP today is **IPv4.** However, IPv6 is also beginning to be supported. IPv6 provides for much longer addresses and therefore for the possibility of many more Internet users. IPv6 includes the capabilities of IPv4 and any server that can support IPv6 packets often can also support IPv4 packets.

- **namespace parameter**. In reference to the ForceWare Network Access Manager application, a namespace parameter is the largest container of parameters. A namespace parameter contains multiple group parameters and/or table parameters.
- **nCLI (NVIDIA command line interface)**. In ForceWare Network Access Manager, `nCLI` is a command line interface that can be used to configure and monitor NVIDIA networking components. nCLI can run in either export or interactive mode.
- **single parameter**. In ForceWare Network Access Manager, a single parameter is the smallest parameter unit. It contains a name and value pair.
- **table parameter**. In ForceWare Network Access Manager, a table parameter is a collection of group parameters (rows) that share the same fields (columns). Table parameters are frequently used as place holders for firewall rules, filters, and statistics. Each row inside the table is uniquely identified by a key. A key is composed of one or more of fields of a row.
- **TCP (Transmission Control Protocol)** is a set of rules (protocol) used along with the Internet Protocol (IP) to send data in the form of message units between computers over the Internet. While IP takes care of handling the actual delivery of the data, TCP takes care of keeping track of the individual units of data (called segments) that a message is divided into for efficient routing through the Internet.

  TCP is known as a connection-oriented protocol, which means that a connection is established and maintained until such time as the message or messages to be exchanged by the application programs at each end have been exchanged. TCP is responsible for ensuring that a message is divided into the

packets that IP manages and for reassembling the packets back into the complete message at the other end. In the Open Systems Interconnection (OSI) communication model, TCP is in layer 4, the Transport Layer.

- **UDP (User Datagram Protocol)** is a communications protocol that offers a limited amount of service when messages are exchanged between computers in a network that uses the IP. UDP is an alternative to the TCP and, together with the IP, is sometimes referred to as UDP/IP.

  Like the TCP, the UDP uses the IP to actually get a data unit (called a datagram) from one computer to another. Unlike TCP, however, UDP does not provide the service of dividing a message into packets (datagrams) and reassembling it at the other end. Specifically, UDP doesn't provide sequencing of the packets that the data arrives in. This means that the application program that uses UDP must be able to make sure that the entire message has arrived and is in the right order. Network applications that want to save processing time because they have very small data units to exchange (and therefore very little message reassembling to do) may prefer UDP to TCP. The **Trivial File Transfer Protocol (TFTP)** uses UDP instead of TCP.

  UDP provides two services not provided by the IP layer. It provides port numbers to help distinguish different user requests and, optionally, a checksum capability to verify that the data arrived intact.

  In the **Open Systems Interconnection (OSI)** communication model, UDP, like TCP, is in layer 4, the Transport Layer.