USER MANUAL

# Tun KERNEL™

## TCP/IP Communication for Windows 3.x

Version 8.50

ESKER
MAKING OPEN SYSTEMS A REALITY

# PREFACE

**Tun KERNEL** for Windows is a set of programs and DLLs implementing the TCP/IP protocol; it also incorporates the necessary tools for the configuration and testing of the protocol.

**Tun KERNEL** for Windows is a complementary product to the **Tun EMUL**, **Tun NET** and **Tun SQL** software packages (see table below).

| | **WINDOWS** | **MS-DOS** |
|---|---|---|
| **Tun KERNEL** | TCP/IP protocol stacks for Windows 3.x only | TCP/IP protocol stacks for MS-DOS (TSR) |
| **Tun NET** | TCP/IP applications (NIS, NFS Client and Server, PING, Printer redirection and sharing, FTP Client and Server, TELNET, RSH Client and Server, TAR, WALL, TFTP, TIME), and electronic mail and fax application | TCP/IP applications for MS-DOS (NFS, Printer sharing, FTP, TELNET, TAR ...) |
| **Tun EMUL** | Comprehensive terminal emulator (asynchronous emulation, IBM 3270 IBM 5250) | Comprehensive terminal emulator for MS-DOS (asynchronous emulation) |
| **Tun SQL** | ODBC drivers for Client-Server mod under TCP/IP (Oracle, Informix, Sybase DBMS, Progress, DB2) and database revamping tools | N/A |
| **TCP/IP Network Services** | Browser NIS, Printer redirection and sharing (LPR, LPD) | N/A |

**Tun KERNEL** for Windows is delivered as standard as part of the **Tun PLUS** software package which incorporates all the above software. It is also included with the **Tun NET**, **Tun EMUL** and **Tun SQL** packages when they are purchased separately.

**Tun KERNEL** for Windows can be installed independently of the other programs; in any case, the installation programs incorporated in **Tun PLUS**, **Tun NET**, **Tun EMUL** and **Tun SQL** include the option to install **Tun KERNEL**automatically.

# TABLE OF CONTENTS

# CHAPTER 1 - INSTALLATION

## PACKAGE CONTENTS

Please make sure your **Tun KERNEL** package contains the following:

- **Tun KERNEL** user manual (TCP/IP Communication for Windows 3.x).
- A CD-ROM.
- A user license.
- A sealed envelope containing a serial number and activation key.
- Miscellaneous technical bulletins (if applicable).
- A registration card.

**Note:** Opening the sealed envelope indicates that you have accepted the terms and conditions of the User License (indicated on the sealed envelope) for using **Tun KERNEL**.

## HARDWARE AND SOFTWARE REQUIREMENTS

To integrate a PC running **Tun KERNEL** into a TCP/IP network, you will need the following equipment:

- A 100% PC 486 compatible micro-computer or Pentium.
- 8-16 MB RAM.
- Windows 3.x.

## BEFORE YOU BEGIN

Before installing **Tun KERNEL**, check to see if either the environment or the Network Card you intend to use require a special configuration procedure.

The **Appendices** in this manual address some of the circumstances which may require special attention. In particular, procedures are given for installing **Tun KERNEL** in multi-protocol environments:

- TCP/IP with Novell NetWare
- TCP/IP with Windows for Workgroups and NetWare
- TCP/IP with Microsoft LAN Manager

## PRODUCT INSTALLATION

**The following instructions may be ignored if you have acquired Tun KERNEL as part of the Tun PLUS package Tun NET, Tun EMUL or Tun SQL since the installation procedure of these products automatically proposes the installation of Tun KERNEL.**

The following procedure describes the installation of the **Tun KERNEL** package under Windows 3.x:

**1.** Insert the CD-ROM into the CD-ROM drive (generally, drive D) and follow one of these steps:

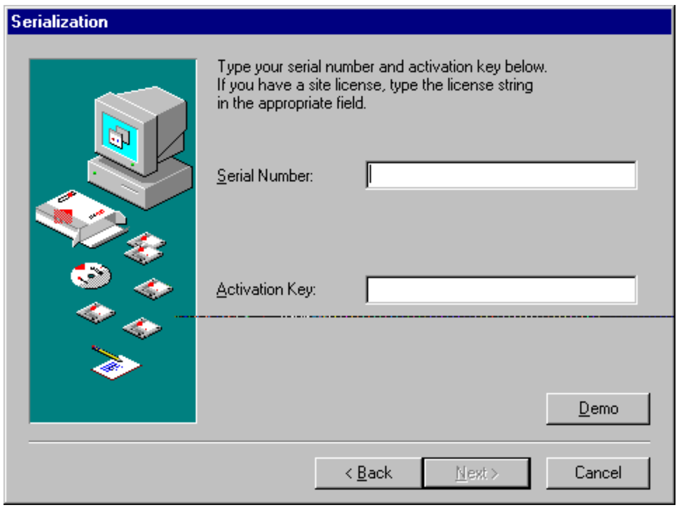
- From the **Program Manager**, select **File➔Run** and type the following command:

**D:\INSTALL.EXE**

- From the **File Manager**, open the drive containing the CD-ROM (drive D) et double-click on the file INSTALL.EXE.

| | |
|---|---|
| Note: | If you are using the **Tun KERNEL** installation disks instead of the CD-ROM, insert the disk labeled "Tun Setup Disk 1/2" into the disk drive (generally drive A), and follow the same procedure with the following command: **A:\SETUP.EXE** |

2. After the Welcome dialog box, you will see the following installation window:


Serialization

Type your serial number and activation key below.
If you have a site license, type the license string
in the appropriate field.

Serial Number:

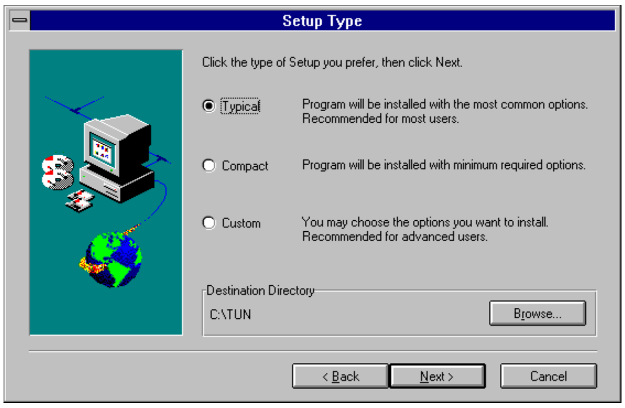
Activation Key:

Enter the serial number and the activation key of the software. This information can be found in the sealed envelope accompanying the software.

Click the button **Demo** if you wish to install the demonstration version of **Tun PLUS**. A serial number and activation key will be proposed for the demo installation.

Then click the button **Next**.

3. If the serial number and activation key are correct, the following window will appear:


Setup Type

Click the type of Setup you prefer, then click Next.

[✓ Typical] Program will be installed with the most common options. Recommended for most users.

Compact Program will be installed with minimum required options.

Custom You may choose the options you want to install. Recommended for advanced users.

Destination Directory
C:\TUN

This dialog box lets you choose the type of installation you want and the installation directory.

### Type of installation

There are three types of installation:

- **Typical**: installs the necessary components for normal usage of **Tun KERNEL**
- **Compact**: installs the minimum number of components for **Tun KERNEL** to function.
- **Custom**: allows you to choose the components you wish to install.

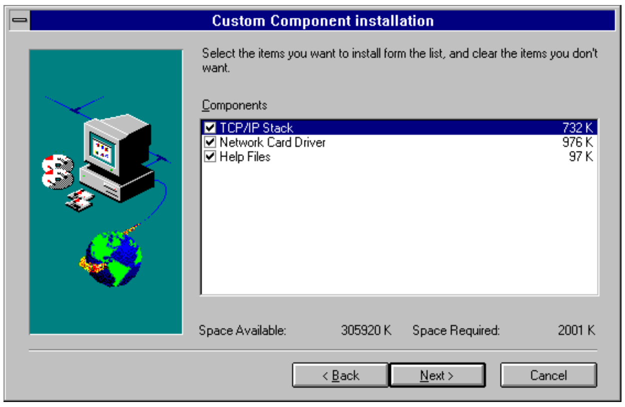
### Installation directory

Win 3.X

The default directory for the installation of **Tun KERNEL** under Windows 3.x is C:\TUN. The directory may be changed by clicking the button **Browse...**.

Click the button **Next** when you are ready.

4. If you have chosen the custom installation, the following dialog box will appear:


Custom Component installation

Select the items you want to install form the list, and clear the items you don't want.

Components

| | |
|---|---|
| ☑ TCP/IP Stack | 732 K |
| ☑ Network Card Driver | 976 K |
| ☑ Help Files | 97 K |

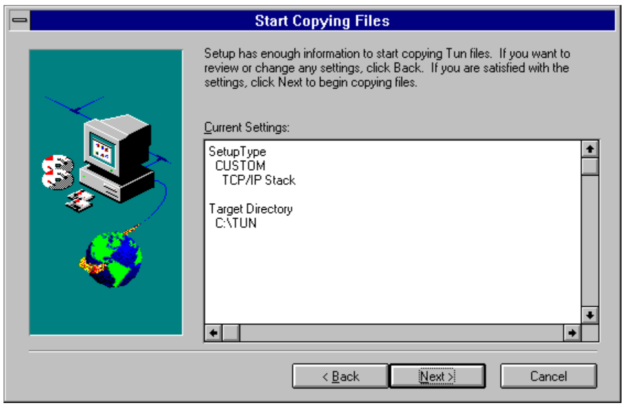
Space Available: 305920 K Space Required: 2001 K

Select the check boxes corresponding to the components you wish to install:

- TCP/IP Stack: DLLs et VxD drivers for TCP/IP stack.
- Network Card Driver: Packet Driver, NDIS Driver.
- Help Files: on-line help for the TCP/IP stack.

Then click the button **Next**.

5. The following window will appear.


Start Copying Files

Setup has enough information to start copying Tun files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Next to begin copying files.

Current Settings:

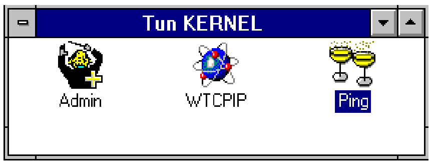
```
SetupType
 CUSTOM
   TCP/IP Stack

Target Directory
 C:\TUN
```

Then click the button **Next** if you are satisfied with the installation options.

6. At the end of the installation process, a window proposes the immediate configuration of **Tun KERNEL**. If you accept, the installation procedure switches immediately to the **Tun KERNEL** configuration screens as described in the chapter **Setting up the TCP/IP Kernel** in this manual. If you

refuse, you can always return to this point later by clicking on the **Administrator** icon in the **Tun KERNEL** or **Tun NET** groups.

7. The installation process has now been completed. You should be able to see the new **Tun KERNEL** group with the following icons:



| | |
|---|---|
| **Note**: | If the **Tun NET** software package has already been installed on the PC, the**Tun KERNEL** icons will be included in the**Tun NET** group under Windows Program Manager. |

## Tun KERNEL CONFIGURATION

**Tun KERNEL** is delivered with an interactive program (WADM2.EXE) for configuring the product. After installing the program files, this program must be run (by clicking on **Administrator** in the **Tun NET** or **Tun KERNEL** groups) in order to set up the various program features.

## DIRECTORY STRUCTURE AND INSTALLED FILES

The default installation procedure creates the directory**\TUN\KERNELW**, and installs the following files:

### Supervisor

| | |
|---|---|
| WADM2.EXE | Supervisor / configuration program |
| WTCPDLG.DLL | TCP/IP configuration interface |
| *.LG | Language files |
| *.INI | Program initialization files |

### TCP/IP Kernel

| | |
|---|---|
| WTCPIP.EXE | TCP/IP scheduler |
| TUNTCPIP.DLL | Multi-interface TCP/IP stack |
| WINSOCK.DLL | Standard TCP/IP Interface Library |
| WPING.EXE | Connection testing utility |
| WSNMPD.DLL | SNMP Agent |
| BOOTP.DLL | BOOTP client |
| WLOG.EXE | TCP/IP trace utility |
| HOSTTAB | Host table |
| SERVICES | TCP services table |
| PROTOCOL | Protocol table |
| AUTOTCP.BAT | Batch file for loading LAN card drivers |
| PACKET.* | Packet driver database |
| NDIS.* | NDIS driver database |
| ODI.* | ODI driver database |
| CLARKSON*.* | Clarkson (Crynwr) packet drivers |
| DRIVERS*.* | Additional LAN card drivers |

# CHAPTER 2 - INTRODUCTION TO Tun KERNEL

## WHAT IS TCP/IP?

TCP/IP is a protocol suite used for connecting computer networks, and for exchanging data between host machines. **TCP** stands for **Transmission Control Protocol**; **IP** stands for **Internet Protocol**. TCP/IP is a standard that is described in its entirety in a set of documents called**RFC**s (Requests For Comment).

All systems that respect TCP/IP standards are able to communicate with each other, regardless of their software and hardware architectures. Today, TCP/IP is the most widely used protocol for interconnecting heterogeneous machines, both locally and across great distances.

The strength of TCP/IP lies in the **standardization** of information transmission, as well as the **specification** of numerous powerful application tools: Terminal Emulation, File Transfer, Messaging, Remote Command Execution, Network Administration...). The TCP/IP protocol suite includes a complete set of advanced API calls. The set of utilities associated with TCP/IP is often referred to as**ARPA Utilities**.

Today, with a widely-installed and commercialized base, TCP/IP is in a position to replace the OSI (Open Systems Interconnection) reference model as the "standard" for network communications.

TCP/IP runs over a wide variety of hardware, depending on the types networks and connections being used. For example, Local Area Networks (LANs) typically use **Ethernet** and **Token Ring**, while Wide Area Networks (WANs) are often structured around different kinds of serial lines and telephone connections.

Remote connections over serial lines using **SLIP** (Serial Line Internet Protocol) and **PPP** (Point-to-Point Protocol) are becoming increasingly popular, especially for accessing the Internet from PCs equipped with modems. A more primitive protocol, SLIP is useful for remote connections, but does not allow dynamic control or error correction. PPP overcomes the weaknesses of SLIP, and is a standard Internet protocol.

## WHAT IS Tun KERNEL?

**Tun KERNEL** for Windows is a network software package that provides a complete set of TCP/IP communications tools. As its name implies, **Tun KERNEL** for Windows provides the following programs and utilities for use under Microsoft Windows:

- TCP/IP kernels based on MS Windows **DLL** and **VxD** standards that entirely implement the TCP/IP communications protocol
- TCP (Transmission Control Protocol)
- UDP (User Datagram Protocol)
- ICMP (Internet Control Message Protocol)
- IP (Internet Protocol)
- ARP (Address Resolution Protocol)
- WinSock 1.1 API (Applications Programming Interface)
- A test program (WPING.EXE)
- An SNMP agent
- A BOOTP client
- An auto-dialing program for a modem for SLIP and PPP connections
- A menu-driven administration program for managing all components of**Tun KERNEL**

## PHYSICAL INTERFACES

**Tun KERNEL** provides a multi-interface TCP/IP kernel that is capable of handling several different physical interfaces at the same time. For example, a PC can be simultaneously connected to a LAN via an Ethernet or Token Ring network card, and to the Internet via PPP or SLIP and a modem.

### Local Area Network interfaces

In order to be able to use the vast majority of **Network Interface Cards** available on the market today, **Tun KERNEL**'s TCP/IP kernel uses the *de facto* **packet driver** standard. Loaded under DOS before entering Windows, packet drivers are small resident programs specific to the type of network card in the PC. Packet drivers handle the interface between the TCP/IP kernel and the physical network (LAN) card.

The packet drivers delivered standard with **Tun KERNEL** are **freeware**, developed by Clarkson University (USA), and now distributed by **Crynwr Software**. In addition, network card manufacturers often supply packet drivers with their cards.

In order to use **ODI** drivers (Novell Netware) and **NDIS** (Microsoft LAN Manager, Windows for Workgroups), **Tun KERNEL** provides converters to adapt these environments to the standard needed by **Tun KERNEL**. These converters may be used to set up configurations with multiple protocols over a single LAN card.

### Telephone line interfaces

**Tun KERNEL** can handle serial ports COM1 - COM9 for PPP and SLIP connections. When used with modems, the TCP/IP kernel automatically controls dialing, connections and disconnections.

# CHAPTER 3 - SETTING UP THE TCP/IP KERNEL

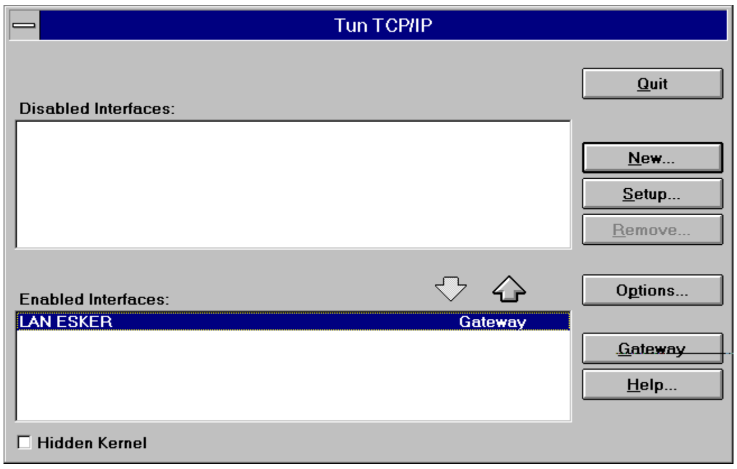
## TCP/IP SETUP

The configuration program supervises the different interfaces (create, modify, activate, deactivate) and defines the general TCP/IP options (SNMP, server table, services table...).

There are several ways to start the TCP/IP configuration module in **Tun KERNEL**:

- By clicking on the TCP/IP button in the **Administrator** control panel.
- By clicking on the **Network** icon (then **Other network**) in the Windows Control Panel.

Any of these methods brings the kernel configuration menu to the screen, as shown in the figure below. This dialog box presents all the parameters relating to the operation of the TCP/IP kernel:



LAN ESKER Gateway

By default, when the TCP/IP kernel is running, its presence is indicated by an icon. The icon, which does not need to be always visible, can be hidden by selecting the check box **Hidden Kernel**.

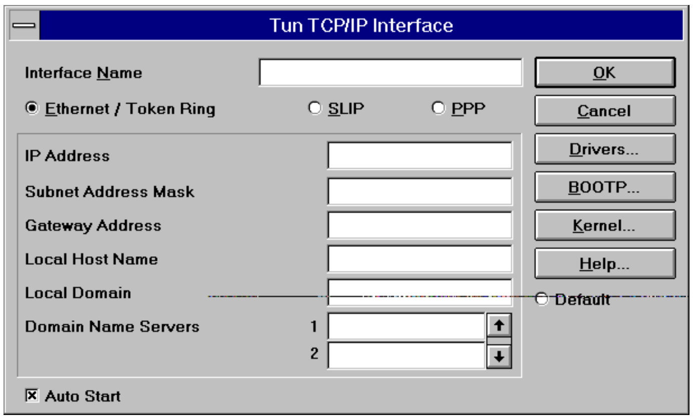
Summary of the steps described in this section:

- Create one or more interfaces by clicking on **New**.
- Check the global configuration by clicking on **Options**.
- In the case of multiple interfaces, determine which one will contain the **Gateway** for reaching hosts not belonging to any of the active interfaces.
- Activate the desired interface by clicking on the down arrow ⇩ .

## ETHERNET AND TOKEN RING INTERFACES

Run the program **Tun Admin+** in **Tun KERNEL** group, and then click on the **TCP/IP** button.

Click on **New** in order to define an interface:



Tun TCP/IP Interface

Interface Name

(● Ethernet / Token Ring) ○ SLIP ○ PPP

IP Address

Subnet Address Mask

Gateway Address

Local Host Name

Local Domain

Domain Name Servers 1 2

○ Default

[✓ Auto Start]

Follow these steps to define a new TCP/IP interface:

- Assign a name.
- Select the protocol type.
- Assign IP parameters (described below).
- Configure the network card parameters (by clicking on **Drivers**).
- Activate BOOTP (optional).
- Modify kernel parameters (optional).

| | |
|---|---|
| ☒ **Auto start** | Determines whether the interface will be activated every time the kernel is started |
| ◉ **Default** | Sets the interface as the default, which means that it will be activated automatically when a TCP/IP request is issued by any TCP/IP application (**telnet**, **ping**, **rsh**, etc.) |

Click on **OK** in order to save the parameters to the TCP/IP configuration file WTCPIP.INI (in the **Tun KERNEL** default directory). At this point, the program will prompt you to start the TCP/IP kernel. Answer "Yes" in order to have your changes take effect right away.

**Note**: If the kernel is running when you make interface changes, immediate update may not be possible, and your changes will take effect next time the kernel is started.

## Setting IP parameters

### IP Address

This fields contains the IP address of the PC. Most importantly, it must not be used by another machine on the network.

An IP address is composed of four fields of numbers, separated by a period (.). For example: *124.131.120.111*, *124.131.120.10*

In the same network, IP addresses must be set in "harmony". As a general rule, the first three fields are the same in small networks, with the last field unique to each host. For example, a UNIX server could have the address 124.131.124.**100**, and a PC in the same network could use 124.131.124.**101**.

In a large network, it is preferable to obtain an IP address from the system administrator. (See Appendix entitled **IP Addresses** for more details.)

### Subnet Address Mask

This field is filled in automatically according to the value given for the IP address. This information should be supplied only if your network is divided into subnetworks. Your network administrator should be able to supply you with a correct Subnet address mask.

### Gateway Address

The Gateway Address indicates the IP address of a gateway machine used to forward packets to other networks or subnetworks. The IP address format for this machine is the same as for any other host. If your network does not contain a gateway, do not enter anything in this field.

### Local Hostname

An alias name by which the PC is known to other machines on the network.

### Local Domain

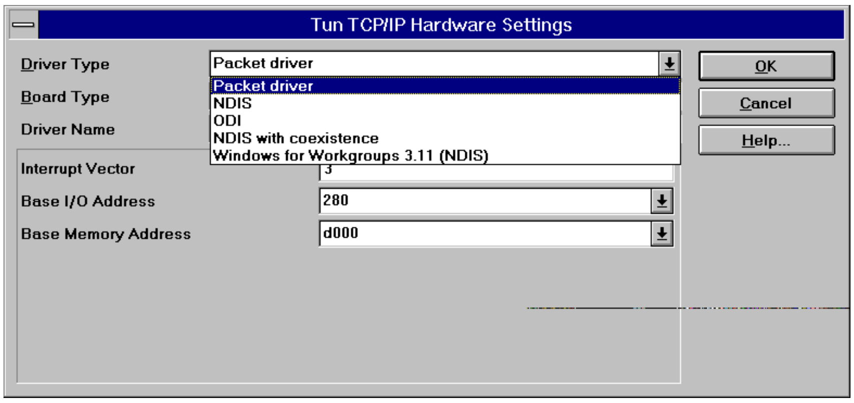
Used if the network is divided into Domains, or is part of a larger Domain. This name should be supplied by the network administrator (optional field).

### Domain Name Server Address(es)

On larger networks, one or more **Name Servers** may be used to store host names and their corresponding IP addresses. Instead of having host tables on every PC, which can be hard to manage, a **Name Server** centralizes the information for everyone to share. The IP address format for this machine is the same as for any other host. If your network does not contain any Name Servers, do not enter anything in this field.

## LAN CARD DRIVERS

Click on **Drivers...** to configure your LAN card:


Tun TCP/IP Hardware Settings
OK
Cancel
Help...
Packet driver
NDIS
ODI
NDIS with coexistence
Windows for Workgroups 3.11 (NDIS)

Driver Type: Packet driver
Board Type
Driver Name
Interrupt Vector: 3
Base I/O Address: 280
Base Memory Address: d000

This window shows the network card's current hardware configuration, including driver type, name and hardware addresses.

## Driver Type

This field presents the different types of network drivers that may be used to handle communication between network cards and TCP/IP. The **Driver Interface type** you should use can depend on both the network card and coexistence between **Tun KERNEL** and other network protocols.

The table below shows the different types of LAN card drivers that you may use with **Tun KERNEL**.

| Driver Type | Comments |
|---|---|
| Packet Driver | Used when no other networking software will be runni on the PC (single protocol installation). There must be a packet driver for the intended LAN card (either supplied with **Tun KERNEL** or available from the manufacturer). |
| NDIS | Used when no other networking software will be runni on the PC (single protocol installation). May be used when no packet driver is available (almost all network cards support the NDIS protocol). |
| ODI | Used when the PC is already configured to run Novell Netware (multiple protocol installation). |
| NDIS coexistence | Used when the PC is already configured for another network product using NDIS (such as LAN Manager). |
| WFWG | Used when the PC is already runni **Windows for Workgroups** and the network card has been configure |

In general, packet drivers offer the most desirable solution in terms of simplicity, speed and memory use. However, if Windows for Workgroups is already running on your PC, the simplest solution is to select that option from the **Administrator** menu.

## Hardware Parameters

The fields given for setting a LAN card's hardware parameters will vary depending on your choice of Driver type. The field Board Type contains the name of the LAN board used for network connection. When you select the **Board Type**, suggested default values for the remaining fields are filled in automatically. The driver databases contain the list of supported LAN cards and their default parameters.

## LAN card descriptions

For proprietary reasons, not all of the packet drivers listed in the PACKET.DBA file (and therefore in the menu) are delivered with **Tun KERNEL**. These drivers are supplied by the LAN card manufacturer, and should be copied into the CLARKSON directory.

Likewise, the NDIS.DBA and ODI.DBA files may contain information about drivers which are not delivered with **Tun KERNEL**. Copy these drivers from the manufacturer's disk into the DRIVERS subdirectory.The following table lists the Packet Driversdelivered with **Tun KERNEL**:

| **Manufacturer** | **Model** |
|---|---|
| 3COM | 3C501, 3C503, 3C505, 3C507, 3C509, 3C523 |
| Allied-Telesis | AT1500, AT1700 |
| AQUILA Comm | Arlan 450 |
| ARCNET | ARCNET |
| AT&T | starLAN 1/10 |
| BICC Data Networks | ISOLAN 4110 |
| Cabletron | DNI Exxxx series |
| D-LINK | DE600 |
| David Systems | Ether-T PC/AT |
| DEC | DE100, DE200, DEPCA Rev E |
| EAGLE (NOVELL) | NE1000, NE2000, NE2100, NE2 |
| FUJITSU | EtherCoupler 86965 |
| HP | EtherTwist, Starlan, Thinlan, 27247B, 27252A |
| IBM | Token Ring |
| ICL (Nokia Data) | EtherTeam 16 |
| INTEL | EtherExpress |
| KODIAK | Raven 8, Raven 16, Kombo |
| MULTITEC | EN-301 |
| MYCRODYNE | EXOS2x5T |
| Mylex | LNE-390B |
| NCR | ET-105 |
| NTI 16 | NTI 16 |
| RACAL INTERLAN | NI5010, NI5210, NI6510, NI9210, ES3210 |
| SCHNEIDER & KOCH | SK-NET |
| SMC (WD) | Elite Ultra, WD8003, WD8013 |
| THOMAS-CONRAD | Tcnet |
| TIARA | LANCARD/E |
| UNGERMANN BASS | NICPC, NICPS2 |
| XIRCOM | Credit card adapter |
| ZDS | Z-note / IBM Thinkpad |

### Hardware settings

You may need to adjust the hardware settings such as **Interrupt vector (IRQ)**, **Base I/O Address**, and **Base Memory Address** to match your board.
When configuring your LAN card, it is important to avoid address and interrupt conflicts with other boards installed in your PC. If you select an interrupt vector that is used by another board, it is unlikely the kernel will load successfully.

IRQ 5 is a good interrupt to use for 8-bit LAN cards, as it corresponds to a rarely-used second parallel port. For 16-bit LAN cards, IRQ values above 9 are usually good to use. In general, try to avoid IRQ 3 or 4, as they correspond to standard COM ports.

You may scroll through the possible values in the **Base I/O** and **Base Memory Address** fields by clicking in the field with the mouse or using the space bar.

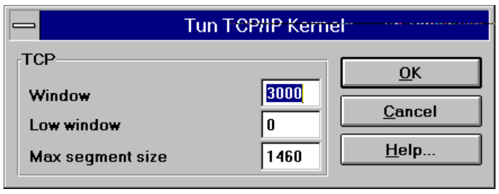
### Saving your changes

If you make any changes in the preceding dialog box, pressing the **OK** button will update the **AUTOTCP.BAT** file in the default directory. For these changes to take effect, you will need to re-boot the PC and run this batch file.

## Kernel Customization

**Note**: The default values described in this section are adequate for most situations.

The **Kernel...** button is used to specify several parameters relating to the TCP/IP kernel itself:



Window 3000
Low window 0
Max segment size 1460

### Window and Low window

These two options regulate the flow of data received by the TCP/IP kernel. Increasing their values may help make network performance more regular, but not necessarily faster. In general, the optimal values are approximately 3000 for **Window** (2 Ethernet packets). In Token Ring installations, this value may be best set at 1024 (especially when routers are used). 3000 is the default value.

### Max segment size

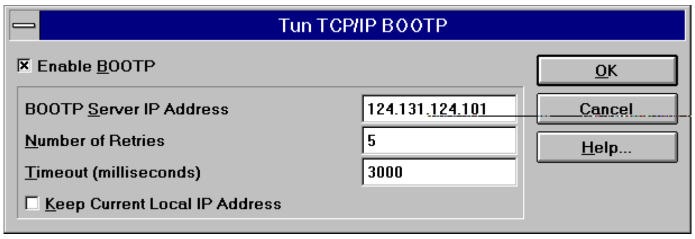
This value specifies the maximum size of *segments* (in TCP terminology), meaning the maximum amount of data contained in packets.

## BOOTP

BOOTP is a utility that retrieves IP parameters, including the IP and gateway addresses, from a remote server. It is often used as a tool to help administrators in large networks manage TCP/IP on PCs from a central location.

**Note**: More details on the BOOTP command and on server configuration are given in chapter 4 of this manual.

The **BOOTP...** button is used to set the parameters necessary for the TCP/IP kernel to send a BOOTP request during startup, before reading the WTCPIP.INI configuration file:



Tun TCP/IP BOOTP

☒ Enable BOOTP

BOOTP Server IP Address: 124.131.124.101

Number of Retries: 5

Timeout (milliseconds): 3000

☐ Keep Current Local IP Address

### Enable BOOTP

This field indicates whether or not BOOTP should be activated when the TCP/IP kernel is started.

### BOOTP server IP address

This field specifies the IP address of the BOOTP server. The address is optional however: if no address is given (default 0.0.0.0), then BOOTP sends out a general broadcast message and waits for a response.

### Number of retries

This parameter specifies how many times the BOOTP request should be re-broadcast if there is no reply from a server.

### Timeout

This field determines how long (in milliseconds) the PC should wait for a response from a BOOTP server.

### Keep current local IP address

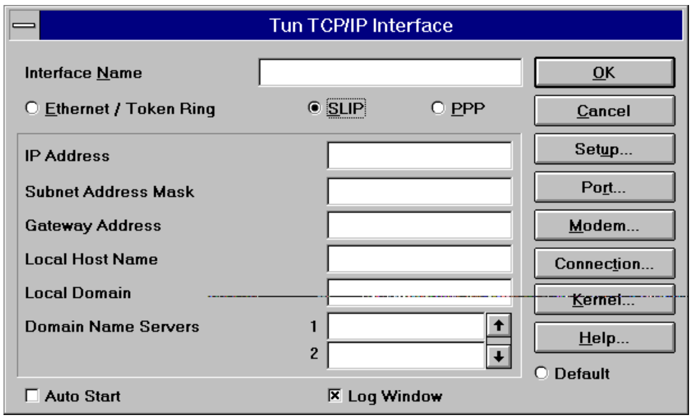
If the IP address currently in memory is different from that sent by the BOOTP server, the current one will remain unchanged.

### Saving changes

Click on **OK** to save your changes or click on **Cancel** to quit and return to the previous menu.

## SLIP AND PPP INTERFACES

By selecting PPP or SLIP as the protocol, you will be able to set all of the parameters for running TCP/IP over serial lines and modems. Click on **TCP/IP➔New** to get to the same menu as described in the previous section:



Chosen settings: [✓ SLIP], [✓ Log Window]

When you select PPP or SLIP, the buttons on the right hand side of this screen change from those used for LAN configurations. In fact, the screen shown above contains the same options as previously described, with the addition of:

- Serial port configuration.
- Modem configuration.
- Connection script definition.

On this screen:

| | |
|---|---|
| ☒ **Auto start** | Determines whether the interface will be activated every time the kernel is started |
| ◉ **Default** | Sets the interface as the default, which means that it will be activated automatically when a TCP/IP request is issued by any TCP/IP application (**telnet**, **ping**, **rsh**, etc.) |
| ☒ **Log Window** | Causes a "trace" window to be opened during connection, so you can follow visually the connection process (dialing, negotiation). |

## Setting IP parameters

### IP Address

This fields contains the IP address of the PC. Most importantly, it must not be used by another machine on the network.

An IP address is composed of four fields of numbers, separated by a period (.). For example: *124.131.120.111* or *124.131.120.10*

IP addresses in the same network must be compatible. As a general rule, the first three fields are the same in small networks, with the last field unique to each host. For example, a UNIX server could have the address 124.131.124.**100**, and a PC in the same network could use 124.131.124**101**.

In a large network, it is preferable to obtain an IP address from the system administrator. (See Appendix entitled **IP Addresses** for more details.)

### Subnet Address Mask

This field is filled in automatically depending on the value given for the IP address. This information should be supplied only if your network is divided into subnetworks. Your network administrator should be able to supply you with a correct Subnet address mask.

### Gateway Address

The Gateway Address indicates the IP address of a gateway machine used to forward packets to other networks or subnetworks. The IP address format for this machine is the same as for any other host. If your network does not contain a gateway, do not enter anything in this field.

### Local Hostname

An alias name by which the PC is known to other machines on the network.

### Local Domain

Used if the network is divided into Domains, or is part of a larger Domain. This name should be supplied by the network administrator. (Optional field)

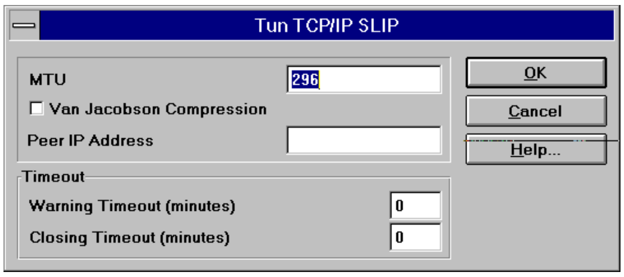
### Domain Name Servers

On larger networks, one or more **Name Servers** may be used to store host names and their corresponding IP addresses. Instead of having host tables on every PC, which can be hard to manage, a **Name Server** centralizes the information for everyone to share. The IP address format for this machine is the same as for any other host. If your network does not contain name servers, do not enter anything in this field.

Use of TCP/IP over the Internet almost always requires DNS services.

## SLIP specifics

Configuration for the actual kernel used for SLIP connections is quite simple, due to the nature of the SLIP protocol. Click on **Setup** to adjust the following parameters:



MTU: 296
☐ Van Jacobson Compression
Peer IP Address:
Warning Timeout (minutes): 0
Closing Timeout (minutes): 0

### MTU (Maximum Transmission Unit)

Maximum packet size used during the connection.

### Peer IP Address

Enter the address of the TCP/IP host to which you will be connecting, also called the *Peer.*

### Van Jacobson Compression

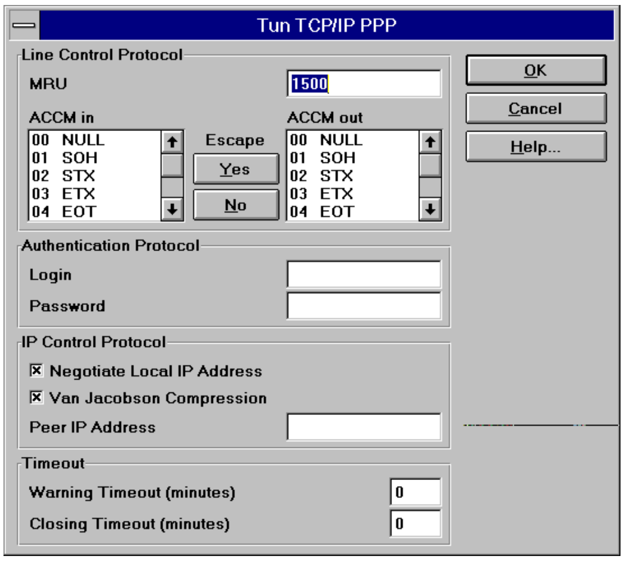
Some SLIP servers use a compression mechanism for the IP headers to improve the data flow in the telephone line. If the server to which you wish to connect has this option, you have to select the check box.

### Warning Timeout/Closing Timeout

Two timeout values are provided to reduce telephone costs. The field **Warning Timeout** contains a value expressed in minutes. This is the length of time the PPP link may remain inactive before a warning message is displayed. The field **Closing Timeout** is also in minutes. It refers to the length of time the PPP link may remain inactive before the telephone connection is cut. If the fields contain zero values (or if they contain no values at all), there are no warnings and no rupture of the communication if the line is inactive.

## PPP specifics

PPP configuration contains more options than SLIP. Click on **Setup** to view the following screen:


Tun TCP/IP PPP

Line Control Protocol

MRU: 1500

ACCM in:
```
00 NULL
01 SOH
02 STX
03 ETX
04 EOT
```

Escape: Yes / No

ACCM out:
```
00 NULL
01 SOH
02 STX
03 ETX
04 EOT
```

Authentication Protocol

Login:

Password:

IP Control Protocol

☒ Negotiate Local IP Address

☒ Van Jacobson Compression

Peer IP Address:

Timeout

Warning Timeout (minutes): 0

Closing Timeout (minutes): 0

### MRU

Specifies the maximum length of data packets supported by the interface

### ACCM in / ACCM out

These two lists allow you to define the format used for certain *control characters* that are sent and received by the PPP interface. Characters may be either:

- **Canonic** - characters are sent "as is"
- **Escape** - characters are accompanied by escape characters

A character should be accompanied by an escape character if there is a risk of it not being easily transmitted on the communication line. This is often the case for the characters between 0x00 and 0x20 which could be interpreted as flow control signals or line free signals. A character "escaped" in this way is not transmitted as it is but in an encoded form which renders it neutral.

By selecting control characters from the **in** and **out** lists, you may determine their ACCM value. Click on **Escape** : **Yes** or **No** in case you need to change any of the defaults.

### Authentication protocol

These fields allow you to specify a user name and password to be used during PPP connection (not related to user login). You must use them when connecting to a server that implements **Authentication Protocol** functions.

### Negotiate Local IP Address

When this option is used, the local IP address (in the PC) will be negotiated with the correspondent. If you give 0.0.0.0 as the IP address for the PC, the correspondent will automatically assign a correct address.

### Van Jacobson Compression

Some PPP servers use an IP header compression mechanism to improve the data flow in the telephone line. If the server to which you wish to connect imposes or uses this option, you should select the **Van Jacobson Compression** check box.

### Peer IP Address

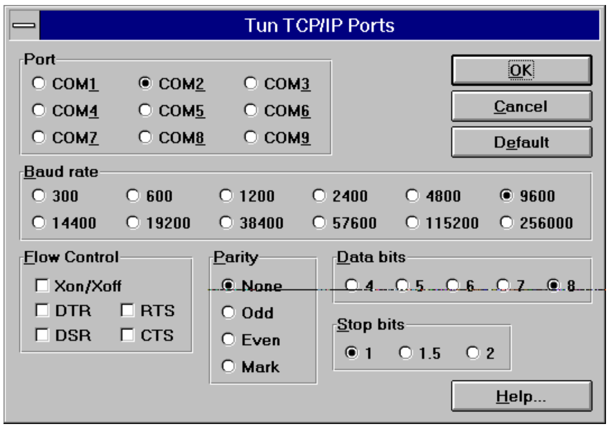
This field may be used to indicate the IP address of the host to which you will be connecting.

### Timeout

Two timeout values are provided to reduce telephone costs. The field **Warning Timeout** contains a value expressed in minutes. This is the length of time the PPP link may remain inactive before a warning message is displayed. The field **Closing Timeout** is also in minutes. It refers to the length of time the PPP link may remain inactive before the telephone connection is cut. If the fields contain a zero value (or if they contain no values at all), there are no warnings and no rupture of the communication if the line is inactive.

## SLIP/PPP: Serial ports

In SLIP and PPP configurations, it is necessary to define serial port and line speed parameters, as shown when you click on**Port**:



Port: COM1 [✓ COM2] COM3 COM4 COM5 COM6 COM7 COM8 COM9

Baud rate: 300 600 1200 2400 4800 [✓ 9600] 14400 19200 38400 57600 115200 256000

Flow Control: ☐ Xon/Xoff ☐ DTR ☐ RTS ☐ DSR ☐ CTS

Parity: [✓ None] Odd Even Mark

Data bits: 4 5 6 7 [✓ 8]

Stop bits: [✓ 1] 1.5 2

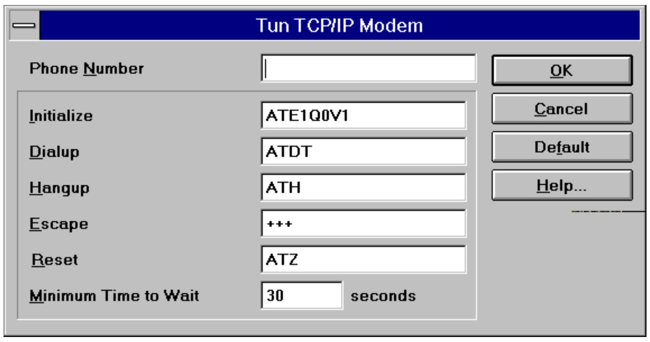
This screen allows you to specify:

- Which serial port to use (COM 1 through COM 9)
- Transmission speed
- Flow control
- Parity
- Data bits
- Stop bits

These options will vary with the type of hardware and host connection you use. Most often "8, none and 1" are used (Data, Parity, Stop Bits), and you should be able to find out very easily if you need to change anything.

## SLIP/PPP: Modem configuration

When using SLIP and PPP over modems, click on **Modem...** in order to enter the phone number and other modem parameters:



| | |
|---|---|
| Phone Number | |
| Initialize | ATE1Q0V1 |
| Dialup | ATDT |
| Hangup | ATH |
| Escape | +++ |
| Reset | ATZ |
| Minimum Time to Wait | 30 seconds |

### Default

Fills in the fields with the same default values as shown above

### Phone Number

Enter the telephone number of the PPP server in this field. You may insert a 2 second pause in a number by using a comma (","): 011,33,78935537

### Initialize

Modem initialization string

### Dial

Determines which dialing mode will be used (ATDT=dial tone, ATDP=dial pulse)

### Hangup

String sent by the peer modem at hangup

### Escape

String used to switch your modem into command mode

### Reset

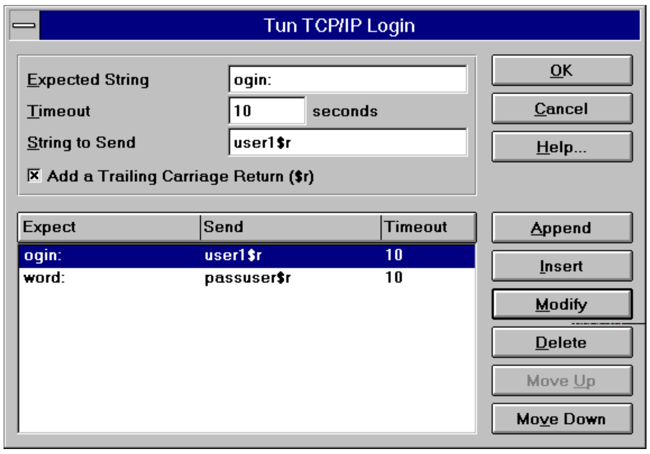
Contains the command to reset your modem

### Minimum Time...

Indicates the minimum amount of time to wait after dialing for the peer to respond. If the peer does not respond within this time period, then the connection is considered to have failed.

## SLIP/PPP: Connection scripts

Before being able to use a SLIP or PPP connection on a remote host, you will need to go through a login/password procedure. Click on **Connection...** in the Tun TCP/IP Interface dialog box to pre-define and automate this process:



Expected String: ogin:
Timeout: 10 seconds
String to Send: user1$r
☒ Add a Trailing Carriage Return ($r)

| Expect | Send | Timeout |
|---|---|---|
| ogin: | user1$r | 10 |
| word: | passuser$r | 10 |

To define a connection script, all you have to do is provide the sequences of characters exchanged by your PC and the responding host.

For each sequence you may define:

- The expected character string.
- A timeout.

- The character string to send.

Sequence order may be changed by selecting a sequence and then clicking on **Move up** et **Move Down**

If you select **Add a Trailing Carriage Return**, every sequence you enter will be automatically appended with a carriage return code ($r).

### Example using above figure

The expected string is **ogin:**, which must arrive in a time 10 seconds. When it is received, the string **user1** is sent, followed by a carriage return. Then we will wait 10 seconds for the string **word:** to arrive, at which point we will send **passuser1**. If the character strings do not arrive in the allotted time, the login fails.

It should be noted that login scripts are case sensitive. In our example, we use "ogin", intentionally omitting the "L", which on some UNIX systems is given as a capital "L", and on others as a lower-case "l". You should also pay attention to the form of the password prompt (password:, Password). The strings expected should be adapted to suit the host.
In order to see what your host requires, a quick telnet session using **Tun EMUL** or the Windows terminal should provide the necessary information.

## Syntax

### Expected String

The string "$i" (as an expected string) may be used to indicate that you are expecting the server to send an IP address as a character string

### String to send

- You may use "#dpa" as a character string in order to change the line parameters during connection
  - **d** specifies **data bits**
  - **p** specifies **parity** (O=odd, E=even, M=mark, N=none)
  - **a** specifies **stop bits** (1=one, 2=two, 3=one and a half)

  For example: **#8N1, #7E1**

  To send the character **#**, enter it twice (**##**).

- $n may be used to insert a "wait" period of "n" seconds.
- To send an ASCII code, use the backslash character (\) followed by the ASCII code in hexadecimal (**2 digits**).

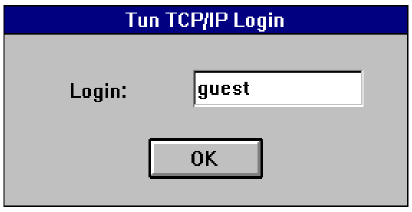
  For example:

| | |
|---|---|
| **\BC** | sends the character whose ASCII code is 0XBC. |
| **\0C** | sends the character whose ASCII code is 0X0C (LF). |

  To send the character ****, enter it twice (**\\**).

- Use question marks (**?**) to ask the user which string he wishes to send.

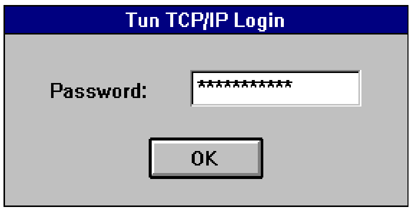
For example:
?Login:? displays the following dialog box:



The entry field allows you to enter the character string you wish to send to the server. The field takes the label indicated between the question marks.

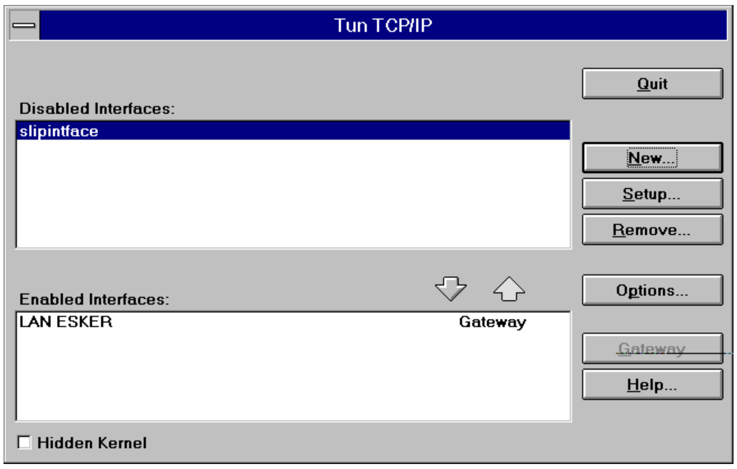
If the first character following the first question mark is an asterisk, the characters entered by the user will be replaced by asterisks.

For example:
?*Password :? displays the following dialog box:



## Loading a SLIP or PPP interface

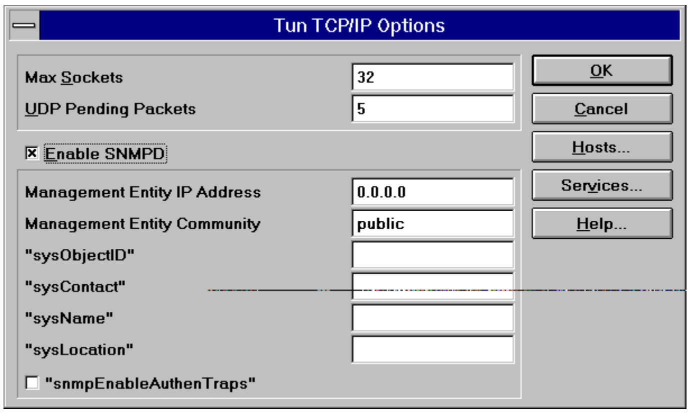
There is no need to quit Windows in order to load a SLIP or a PPP interface. Once the parameters are defined, the user may simply return to the following screen,



Select the interface to load and then press the down arrow: . If a login window was requested (the default is Yes), it should appear to inform the user of the progress of the login process. The login window is extremely useful for setting up the modem and the login script definition.

## GLOBAL KERNEL PARAMETERS

Several options that are not specific to any particular interface may be defined for the TCP/IP kernel by clicking on **Options**:



Max Sockets: 32
UDP Pending Packets: 5
☒ Enable SNMPD
Management Entity IP Address: 0.0.0.0
Management Entity Community: public
"sysObjectID"
"sysContact"
"sysName"
"sysLocation"
☐ "snmpEnableAuthenTraps"

### Timezone Offset

Expressed in minutes, the value given in this field represents the difference between your current time zone and Greenwich Mean Time. The value is most useful for NFS, which synchronizes the dates and times of file creation with respect to the server.

To find the offset of your time zone, take the difference between the values given by a **date** command on the PC and a **date -u** command on your UNIX server.

### Max sockets

This field determines the total number of process connections (sockets) that may be established by the PC. For example, each terminal emulation session requires one socket, each X11 window requires one socket, each remote printer requires one socket, and so on. Simply add up the anticipated number of processes, and enter that number into the **Max Sockets** field.

The default value is 32, with 256 as the maximum.

### UDP pending packets

This field specifies the maximum number of simultaneous UDP packets memorized by the TCP/IP kernel. NFS and *address lookup* on a name server use this type of connection. Increasing the value helps keep data flow smooth, but does not necessarily increase speed. The default value is 5.

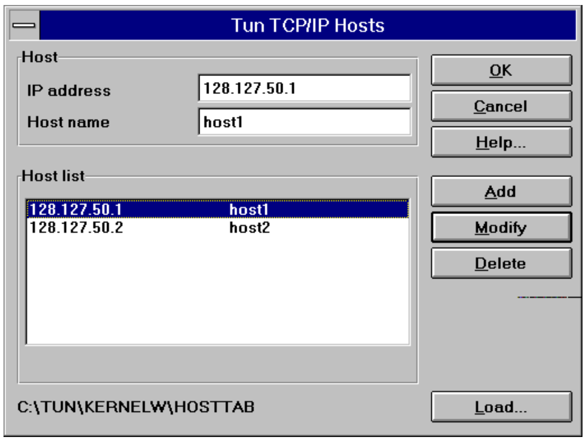
### Enable SNMPD

Use this check box to activate a MIB II-compliant SNMP agent (Simple Network Management Protocol). This allows remote administration of the PC by an SNMP administrator such as Netview/6000. For this option to take effect, it is necessary to stop and restart the TCP/IP Kernel. The SNMP protocol is managed by the executable file WSNMPD.EXE whose icon is visible or hidden depending on whether the Kernel icon is displayed or not.

# TCP/IP TABLES

## Editing a Host Table

**Host tables** (**Options➔Hosts**) are used to assign alias names to the hosts on your network. The IP address of a particular host is listed in the left column, and the alias name that you wish to assign is in the right column:



IP address: 128.127.50.1
Host name: host1

| | |
|---|---|
| 128.127.50.1 | host1 |
| 128.127.50.2 | host2 |

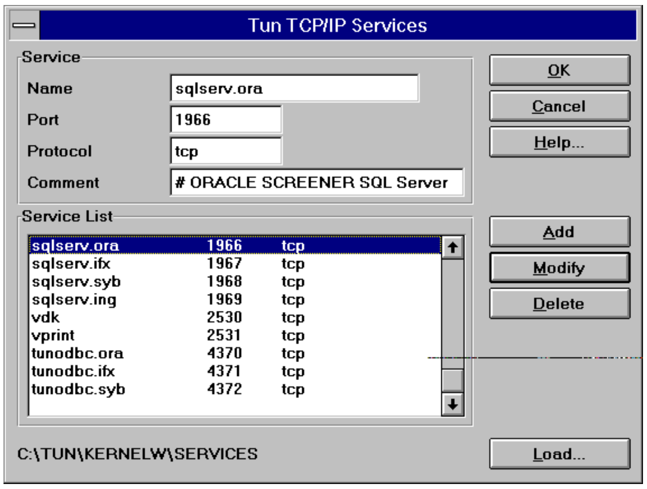
C:\TUN\KERNELW\HOSTTAB

After assigning an alias name to a host, you may then refer to that host by its name (rather than its IP address) when using terminal emulation, network utilities such as**ping**, or assigning NFS drives.

**Note**: You do not need to create or use a host table on the PC if your network uses**Name Server**.
A " # "character at the beginning of a line will render the line invalid.

## The Services table

To consult or modify the services table, click on the button**Options➔Services...**:



Name: sqlserv.ora
Port: 1966
Protocol: tcp
Comment: # ORACLE SCREENER SQL Server

| | | |
|---|---|---|
| sqlserv.ora | 1966 | tcp |
| sqlserv.ifx | 1967 | tcp |
| sqlserv.syb | 1968 | tcp |
| sqlserv.ing | 1969 | tcp |
| vdk | 2530 | tcp |
| vprint | 2531 | tcp |
| tunodbc.ora | 4370 | tcp |
| tunodbc.ifx | 4371 | tcp |
| tunodbc.syb | 4372 | tcp |

C:\TUN\KERNELW\SERVICES

The **Services** table lists software port numbers for well-known TCP services. When a client process such as **Tun VT320** or **Tun FTP** requests a server process, it consults the services table to find out which software port number to address on the server.

Usually this table does not need to be modified, but rather serves as a reference to programmers who wish to interface their programs with known services. It is also possible to add new services to the table.

## LOADING THE TCP/IP KERNEL

### DOS drivers

Even if the TCP/IP in **Tun KERNEL** for Windows is entirely implemented as DLL programs for Windows, it is still necessary to load the drivers for your network card before entering Windows. These drivers are usually quite small, around 4-5 Kb.

The file **AUTOTCP.BAT** is a batch file that loads a packet driver with the appropriate parameters. Here is a sample AUTOTCP.BAT:

```
@echo off
if "%1" = ="" goto start
if "%1" = ="START" goto start
if "%1" = ="start" goto start
if "%1" = ="STOP" goto stop
if "%1" = ="stop" goto stop
:error
echo "Usage : %0 START|STOP
goto end
:start
C:\TUN\KERNELW\CLARKSON\SMC_WD 0x62 5 300
C:\TUN\KERNELW\WINPKT 0x62
goto end
:stop
C:\TUN\KERNELW\TERMIN 0x62
:end
```

**Note**: In the above example, SMC_WD and WINPKT are small TSRs. Packet driver names vary with the type of card used, but **WINPKT is always necessary**. To unload these packet drivers from memory, run the batch file with the option "**stop**" (`AUTOTCP STOP` ), but **do not do this from a DOS session in Windows** It is not necessary to run this batch file if you only use SLIP and PPP interfaces.

### Starting the kernel under Windows

By default, the TCP/IP kernel is not started automatically when Windows is run, but rather starts when an application needs its services (WPING, WFTP, etc.).

It is also possible to start TCP/IP by clicking on the **TCP/IP** icon in the **Tun KERNEL** group.

To shut-down an active TCP/IP kernel, click on the **TCP/IP** icon and select **Close** from the system menu. The kernel may be shut-down and restarted by clicking on the up/down arrows in the **Administrator** program (⇧ ⇩ ).

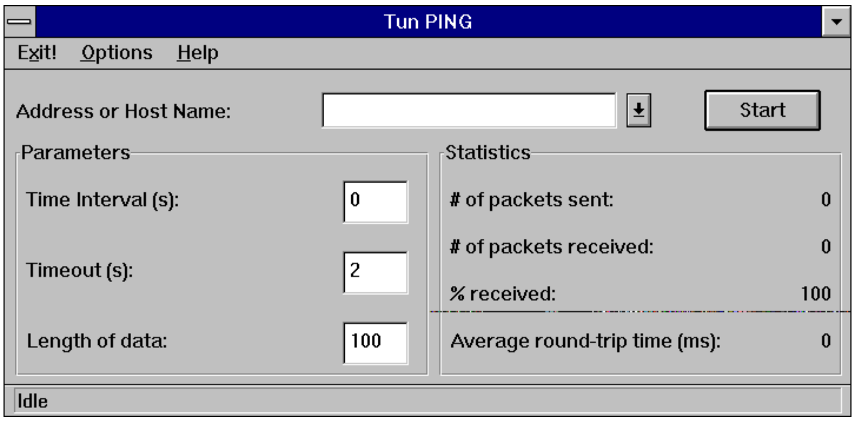
**Note**: Be careful *never* to load a DLL TCP/IP kernel at the same time as a TSR TCP/IP kernel!

Before entering Windows, make sure that neither ETHTCP.EXE or TOKTCP.EXE has been loaded.

## TESTING TCP/IP CONNECTIONS

The **Tun PING** utility is used to test network connections between two machines by sending an *ICMP_ECHO* request.

Run the program **Tun PING** by clicking on the PING icon in **Tun KERNEL** group.



To start the test, enter the IP address, or alias name (as entered in the host table), of the host you would like to reach in the field **Address or Host Name**.

Click on **Start** to begin the PING program.

If the TCP/IP kernel and network are functioning properly, and the server you are attempting to reach is configured correctly, the number of packets sent will equal the number of packets received. The percentage of packets received should be 100%. On a local area network, the **Average round-trip time** should be lower than 20 ms.

**Note**: **Length of data** may need to be adjusted when using WPING through routers.

If a connection is not 100% reliable, or does not work at all, then some link in the chain between Ping and the TCP/IP on the server is improperly configured (usually the network card). There are a few very simple places to begin looking for the cause of a PING failure:

- Make sure that the LAN card drives load correctly by watching the messages on the screen during startup. A **hard-coded** Ethernet or Token Ring address must be displayed when the packet driver is loaded.
- Make sure that there are no IRQ or address conflicts between the network card and other cards. Among other things, the card must pass manufacturer's diagnostics, and not have a memory address that is also used by EMM386.
- Check the quality of the network cable, and make sure that terminators are present if necessary (on thin Ethernet).
- Make sure that TCP/IP is running correctly on the server (by trying a *telnet* from the server to itself).

## Options

### Echos

By default, **Tun PING** sends ICMP requests to the server. In this configuration, **Tun PING** can only be used with **Tun KERNEL** stacks. To use **Tun PING** with different TCP/IP stacks, you may have it send UDP requests on the server's **echo** service by using the menu option **Options➔Echo UDP**.

### Beep

Select the option **Beep** if you wish to hear an audible signal of the packet emission and reception.

# CHAPTER 4 - USING BOOTP

## INTRODUCTION TO BOOTP

BOOTP is a utility for reading TCP/IP configuration information for PCs from a remote server, including parameters such as IP address, Gateways and Name servers. The values supplied by the BOOTP server are then used by the PC's TCP/IP kernel when it is started, which means it is using "guaranteed" reliable information. BOOTP may be used for configuring and updating machines with or without hard disks.

An indispensable tool for large networks, BOOTP allows administrators to centralize IP information for PCs on a UNIX server, instead of having each PC contain its own configuration files. Network changes, such as new gateways and name servers, can be easily added to a single file on the BOOTP server. PCs can be guaranteed reliable and up-to-date information because BOOTP is run before TCP/IP.

Here is how BOOTP functions:

- The BOOTP client (PC) creates a **BootRequest** packet that it broadcasts on the network.
- A BOOTP server responds with a **BootReply** that contains IP information from **/etc/bootptab.** In addition to the client's IP address, responding server name and configuration file to transfer, a **BOOTP** packet also contains a zone called the "Vendor Specific Area" which contains additional information in a particular format, such as Subnet mask, list of Time servers, Gateways and Domain name servers.
- Configuration files on the PC will be updated by the BOOTP reply if necessary.

## SERVER CONFIGURATION

The BOOTP database is contained in the file **/etc/bootptab** on the UNIX host. Follow these steps to configure the BOOTP server:

**1.** Edit **/etc/bootptab** and add one entry for each *bootp* client:

```
testpc:ip=128.2.11.157:ht=1:ha=0000C0915C2A:sm=255.255.0.0:\
:ds=128.2.11.200 128.2.11.80:\
:gw=128.2.11.200:\
:ns=128.2.11.80:\
:ts=128.2.11.200:\
:hn:vm=rfc1048:
```

The syntax used in the **/etc/bootptab** file is similar to that used in **/etc/termcap**. The name of the PC is always given as the first variable, other variables are listed below (with mandatory options marked with an asterisk).

| VALUE | * | DESCRIPTION |
|---|---|---|
| bf | | Name of boot file |
| cs | | Cookie Server address |
| ds | | Domain Name Server address |
| gw | | Gateway address |
| ht | * | Hardware type (Ethernet, Token Ring...). This must be specified before the **ha** parameter |
| ha | * | Physical address of the client's LAN card. This is the number that the BOOTP server uses to respond to a client request |
| hd | | Home Directory |
| hn | | Indicates that the server should send the client's "host name" |
| im | * | Print server |
| ip | * | IP address for the client (PC) |
| lg | | Log server |
| lp | | LPR server |
| ns | | IEN 116 Name Server |
| rl | | Resource Location Protocol Server |
| sm | | Subnet mask |
| tc | | Template host (points to another entry in **/etc/bootptab** in order to avoid repeating information |
| to | | Time Offset |
| ts | | Time Server |
| vm | * | Include the "vendor specific area" as recommended in RFC 1048 |

Some rules that apply to the syntax of **/etc/bootptab**:

- The hostname (PC) is always the first field.
- The **hardware address type** (ht) must be declared before the **hardware address** itself (hd).
- The gateway address must be a bootp server, otherwise there is no response.
- Specify only the NECESSARY fields (i.e., if using BOOTP with **Tun KERNEL** only, the home directory and boot file entries are not needed).

**2.** Edit **/etc/services** and make sure that the *bootps* service is entered in the **/etc/services** file, as shown here:

```
bootps              67/udp #Boot Server Port
bootpc              68/udp #Boot Client Port
```

**3.** Edit **/etc/inetd.conf**. The bootp server also needs to be entered in **/etc/inetd.conf**:

```
bootps       dgram         udp    wait   root   /etc/bootpd bootpd
```

**4.** The bootp server is automatically started every time a request is received on port 67, and **/etc/bootptab** is read.

## USING BOOTP

In **Tun KERNEL** for Windows, BOOTP is included in the TCP/IP kernel, and activated using the **BOOTP** option in the Supervisor.

BOOTP broadcasts a special packet on the network. If a server recognizes the physical address of the network card that sent the packet, then the information contained in **/etc/bootptab** is returned to the PC. The BOOTP program then updates the WTCPIP.INI file on the PC before starting the TCP/IP kernel.

# CHAPTER 5 - REFERENCE GUIDE

## INDEX

| | |
|---|---|
| **HOSTTAB** | Host table |
| **NDIS.CTL** | Secondary database for referenced NDIS drivers described in NDIS.DBA. |
| **NDIS.DBA** | Database containing the list and syntax of the NDIS drivers referenced within **Tun KERNEL** |
| **ODI.CTL** | Secondary database for referenced ODI drivers |
| **ODI.DBA** | Database containing the list and syntax of the ODI drivers referenced within **Tun KERNEL** |
| **PACKET.CTL** | Secondary database for supported packet drivers |
| **PACKET.DBA** | Database containing the list and syntax of the packet drivers referenced within **Tun KERNEL** |
| **SERVICES** | Services table |
| **WPING.EXE** | Sends an ICMP ECHO_REQUEST to a host |
| **WTCPIP.INI** | Configuration file for TCP/IP stacks |
| **WSNMPD.EXE** | SNMP agent |
| **WADM2.EXE** | **Tun KERNEL** Supervisor / Configuration |

## HOSTTAB HOSTTAB

Local host table.

*Description*

The host table is used to assign alias names for network hosts based on their IP addresses. In practice, it is easier to refer to a host by name than by address.

Each line contains a server and its address. If your network contains a name server, it is probably not necessary to use a local host table.

*Example*

```
121.131.118.1             xenix
121.131.118.10            scounix
121.131.118.80            sun
121.131.118.200           risc
121.131.118.42            me
```

*See also*

**SERVICES**

## NDIS.CTL NDIS.CTL

Secondary database for the NDIS drivers described in the NDIS.DBA.

*Description*

This file contains the list of translated-type parameters used in the NDIS.DBA. You may add items to this ASCII file as necessary.

Each line represents a "type", made up of variable number of fields separated by the character "|". The order and content of the fields is a follows:

- Label of the translated type (given in the NDIS.DBA)
- Label of the first parameter
- Translated value of the first parameter
- Label of the second parameter
- ...

**Note**: The label of a translated type cannot begin with the letters A, I, M, m, or H.

*See also*

**NDIS.DBA**

## NDIS.DBA NDIS.DBA

Database for the NDIS drivers referenced in**Tun KERNEL**

*Description*

This file contains the list and syntax of the NDIS drivers used by **Tun KERNEL**. You may add items to this ASCII file as necessary.

Each line represents a variable number of fields separated by the character "|". The order and content of the fields is as follows:

- Name of the NDIS driver
- List of network cards supported by the driver
- Execution syntax
- Symbolic name of the NDIS driver used in the PROTOCOL.INI
- Name of the NDIS driver loaded into memory
- Type of TCP/IP kernel to use (ETH= Ethernet, TOK=Token Ring). If not specified, the Ethernet kernel will be used
- Number of parameters to be presented on the**Tun KERNEL**Supervisor screen (maximum 5)
- Label of the first parameter displayed on the Supervisor screen
- Type of first parameter: (I=Integer, H=Integer in hexadecimal notation, A=I/O Address, M=Long memory-type address, m=Short memory-type address, XXX=translated type defined in the NDIS.CTL file.
- Default value of the first parameter
- Keyword for recognizing the first parameter in the PROTOCOL.INI
- Write format (C language) of the first parameter in the PROTOCOL.INI
- not used
- Label of the second parameter
- Second parameter type
- Default value for the second parameter
- ...

*Note*

The label of a translated type cannot begin with the letter A, I, M, m, or H.

*See also*

**NDIS.CTL**

## ODI.CTL ODI.CTL

Secondary database for the ODI drivers described in the ODI.DBA.

*Description*

This file contains the list of translated-type parameters used in the ODI.DBA. You may add items to this ASCII file as necessary.

Each line represents a "type", made up of variable number of fields separated by the character "|". The order and content of the fields is a follows:

- Label of the translated type (given in the ODI.DBA)
- Label of the first parameter
- Translated value of the first parameter
- Label of the second parameter
- ...

*Note*

The label of a translated type cannot begin with the letter A, I, M, m, or H.

*See also*

**ODI.DBA**

## ODI.DBA ODI.DBA

Database for the ODI drivers referenced within**Tun KERNEL**

*Description*

This file contains the list and syntax of the ODI drivers used by **Tun KERNEL**. You may add items to this ASCII file as necessary.

Each line represents a variable number of fields separated by the character "|". The order and content of the fields is as follows:

- Name of the ODI driver
- List of network cards supported by the driver
- not used
- not used
- not used
- Type of TCP/IP kernel to use (ETH= Ethernet, TOK=Token Ring). If not specified, the Ethernet kernel will be used
- Number of parameters to be presented on the**Tun KERNEL**Supervisor screen (maximum 5)

- Label of the first parameter displayed on the Supervisor screen
- Type of first parameter: (I=Integer, H=Integer in hexadecimal notation, A=I/O Address, M=Long memory-type address, m=Short memory-type address, XXX=translated type defined in the ODI.CTL file.
- Default value of the first parameter
- Keyword for recognizing the first parameter in the NET.CFG
- Write format (C language) of the first parameter in the NET.CFG

- Label of the second parameter
- Second parameter type
- Default value for the second parameter
- ...

*Note*

The label of a translated type cannot begin with the letter A, I, M, m, or H.

*See also*

**ODI.CTL**

## PACKET.CTL PACKET.CTL

Secondary database of the packet drivers described in the file PACKET.DBA.

*Description*

This file contains a list of the translated parameters used in the PACKET.DBA. Since it is an ASCII file, new information may be easily added.

Each line represents a **type**, with a variable number of fields separated by the character " | ". The order of the fields is as follows:

- Label of the translated **type** (used in PACKET.DBA)
- Name of the first parameter
- Translated value of the first parameter

- Name of the second parameter
- ...

*Note*

The label of a translated **type** may not begin with the letters A, I, M, m, or H.

*See also*

**PACKET.DBA**

## PACKET.DBA PACKET.DBA

Database containing the packet drivers referenced within**Tun KERNEL**

*Description*

This file contains the list and syntax of all the packet drivers referenced in the **Tun KERNEL** menus. Since it is an ASCII file, new packet drivers may be easily added.

Each line represents a packet driver, with a variable number of fields separated by the character " | ". The order of the fields is as follows:

- Name of the packet driver
- List of network cards supported by the packet driver
- Syntax for packet driver startup (C language format)
- Syntax for unloading the packet driver
- Type of TCP/IP kernel to use (ETH= Ethernet, TOK=Token Ring). If not specified, the Ethernet kernel will be used
- not used
- The number of parameters that are asked for in the supervisor program (maximum 5).

- The label for the first parameter, which will be displayed on the configuration screen of the Supervisor
- Data type of the first parameter (I:Integer, H:Integer in hexadecimal notation, A: I/O-type address, M:Long memory-type address, m: Short memory-type address, XXX: translated type defined in PACKET.CTL).
- Default value for the first parameter
- Not used
- Not used
- Not used
- Label of the 2nd parameter
- Type of the 2nd parameter
- Default value for the second parameter
- etc...

*See also*

**TERMIN.COM**

## SERVICES SERVICES

TCP/IP services table.

*Description*

The SERVICES file contains a list of the most frequently used services used in a TCP/IP environment.

Each line in the file describes a single service using the following syntax:

```
name          number/protocol          alias
```

where:

| | |
|---|---|
| **name** | is the official name of the service (i.e. telnet) |
| **number** | is the software port number used by the service |
| **protocol** | is the name of the TCP/IP sub-protocol used by the service (TCP, UDP, ICMP...) |
| **alias** | is a list of alternative names, separated by spaces. |

*See also*

**HOSTTAB**

## WADM2 WADM2

Configuration menu for **Tun KERNEL** for Windows.

*Syntax*

```
wadm2
```

*Description*

WADM2.EXE is an interactive program that is used to configure, test, and launch the various functions offered by **Tun KERNEL** for Windows.

This program is described in detail in earlier chapters of this manual.

## WPING WPING

Sends an ICMP ECHO_REQUEST towards a TCP/IP host.

*Syntax*

wping [-h"hostname][-k"Niskey"]

*Description*

**WPING** sends an ICMP ECHO_REQUEST to a TCP/IP host on the network, and then waits for a response. This utility is used to test the physical and logical connection between a PC and a host server, and therefore can only function after the TCP/IP kernel has been loaded into memory on the PC.

The command line option of WPING.EXE (WPING32.EXE) is:

| | |
|---|---|
| **`-h"hostname"`** | lets the user indicate the name or the IP address of the server . |
| **`-k"Niskey"`** | starts Tun PING with a connection to a NIS-defined server |

There are two messages generated by the PINGprogram (WPING.EXE):

1. "**Host Responding**" indicates that the connection is working correctly,
2. "**Host Unreachable**" indicates that connection has not been successful.

*Note*

There are numerous reasons why a connection might fail. Here are three of the most common:

1. The Network Interface Card is incorrectly installed or configured,
2. The IP address of the PC is incompatible with those of the server or network,
3. Cabling is faulty.

## WSNMPD WSNMPD

MIB II-compliant SNMP agent.

*Syntax*

```
WSNMPD [-h] [-s]
```

*Description*

This program implements an SNMP agent which complies with the MIB II standard. It supplies statistics on the functioning of the TCP/IP kernel in ASN.1-compatible format.

WSNMPD has no user interface. It can only be used on the TCP/IP network with an SNMP supervisor (IBM Netview 6000 or HPNetView).
The agent is especially useful for administrators of large TCP/IP networks.

The -h option runs the SNMP agent as a background process without an icon.
The -s option runs the SNMP agent as a background process but this time with
an icon visible (an American policeman's hat).

## WTCPIP.INI WTCPIP.INI

Configuration file for the TCP/IP stacks.

*Description*

The WTCPIP.INI file contains the list of parameters used by the TCP/IP stacks. The different fields and stacks have the following meanings:

| | |
|---|---|
| **[ARP]** | |
| Entries=16 | Max ARP entries memorized in the kernel table |
| **[Common]** | |
| Hosts | Name and position of the server table |
| InstDir= | Installation directory for**Tun KERNEL** |
| **[TCP/IP]** | |
| HiddenKernel=no | Let the kernel icon appear on the desktop |
| MaxNet=3 | Maximum of active network interfaces at the same time |
| Services= | Name and position of the services table |
| Protocol= | Name and position of the protocol table |
| DefNet=TCP/IP1 | Name of the default interface (refers to a section which has to be defined below) |
| LAN=Ethernet | Default interface type |
| TimezoneOffset= | Time difference in minutes in relation to GMT |
| SNMPD=no | The SNMP agent must be running |
| DefGateway= | Name of the interface whose "gateway" is to be the default |
| **[WINSOCK]** | |
| MaxSockets=32 | Maximum number of sockets open at the same time |
| **[UDP]** | |
| UDPWindow=3 | Maximum number of pending UDP packets |
| **[TCP]** | |
| ConnectTMO=12000 | Time (in milliseconds) of the timeout for the opening of a connection |
| ;**Interface Parameters** | |
| **[TCPIP1]** | **This section describes an interface** |
| AutoStart | Automatic startup of the interface |
| Name= | Name of the interface |
| Type= | Type of interface (ETHERNET, TOKENRING, SLIP, PPP) |
| Log= | Displays or hides a trace window when the interface is run (YES \| NO) |
| IPAddr= | Interface IP address |
| SubnetMask= | Interface subnet mask |
| Gateway= | IP address of the default gateway |
| DNS1= | IP address of the first DNS |
| DNS2= | IP address of the second DNS |
| DNS3= | IP address of the third DNS |
| DNSTMO=5 | Maximum timeout for a DNS request |
| Hostname= | Local machine name |
| Username= | User name |
| Domainname= | Name of the domain of the machine |
| PeerMSS | Maximum segment size of the peer |

| | |
|---|---|
| ;**Parameters of the TCP layer** | |
| MTU | Maximum size of a transfer unit (maximum size of an IP packet) |
| TCPWindow=(MTU*2) - 40 | Size of the TCP window |
| TCPMSS=MTU - 40 | Maximum size of a TCP segment |
| ;**Communication ports** | |
| BaudRate=9600 | Line transmission rate (110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 256000) |
| Bits=8 | Number of bits (4, 5, 6, 7, 8) |
| CTS=0 | Use of the CTS signal (0 or 1) |
| CTSTMO=30 | Timeout for the CTS signal (0 or 1) |
| Dialup= | Modem dialing command (ATDT) |
| DSR=0 | Use of the DSR signal (0 or 1) |
| DSRTMO=30 | Timeout for the DSR signal (seconds) |
| DTR=0 | Use of the DTR signal (0 or 1) |
| Escape= | Hayes Escape command (+++) |
| Hangup= | Command to close modem communication |
| IdleTime1=0 | Timeout (in seconds) after which the inactivity of the line will switch the interface to the "Warning" state |
| IdleTime2=0 | Timeout (in seconds) after which the inactivity of the line will close the interface (saves telephone connection time) |
| Init= | Modem initialization string |
| OnlineCheck=0 | Check of the carrier during the connexion |
| Parity=None | Line parity (None, Even, Odd, Mark) |
| Phone= | Telephone number |
| Port=COM2 | Name of the communication port used (COM1 to COM9) |
| Reset=ATZ | Modem reset command) |
| RTS=0 | Use of the RTS signal (0 or 1) |
| RXQUEUE | Size of the byte receiving queue |
| StopBits=1 | Number of stop bits (1, 1.5, 2) |
| TXQUEUE | Size of the sending queue |
| VJ=1 | Activation or deactivation of Van Jacobson compression of the IP and TCP packet headers (0 or 1) |
| XonXoff=0 | XonXoff data flow control (0 or 1) |
| ;**Modem and dial parameters** | |
| DialTMO=45 (ms) | Maximum timeout before obtaining the line |
| OKTMO=5 (ms) | Maximum timeout before obtaining an OK reply from the modem |
| OKDelay=0 (10ms) | Time interval necessary between sending two commands to the modem (useful for some old modems) |
| **;LAN parameters** | |
| BOOTPEnable=No | BOOTP feature enable/disable |
| BOOTPPreserveIPAddr | Ignore the BOOTP address, keep the IP address instead |
| BOOTPRetry=5 | Number of retry times to get the BOOTP address |
| BOOTPServer=0.0.0.0 | IP address of the BOOTP server |
| BOOTPTMO=1000 | Timeout before retrying to contact the BOOTP server |

| | |
|---|---|
| ;**ETHER parameters** | |
| LowBuf=0 | Minimum size of IP packet to be received |
| MTU=1500 | Maximum size of a transfer unit (maximum size of an IP packet) |
| ;**TOKEN parameters** | |
| LowBuf=0 | Minimum size of IP packet to be received |
| MTU=900 | Maximum size of a transfer unit (maximum size of an IP packet) |
| ;**PPP parameters** | |
| AuthPwd | User password to use for the connection |
| AuthUsr | User name to use for the connection |
| CHAPTMO=10 | Timeout to negociate the CHAP authentification |
| Conf=10 | Maximum number of configuration requests to send before acknowledging failure |
| MaxAuth=2 | Number of authentication attempts |
| MTU=0 | Maximum size of a transfer unit (maximum size of an IP packet) |
| Nak=5 | Maximum number of ConfNak to send before acknowledging failure |
| PPPDelay=0 | Timeout before sending the first configuration request when the connection is established |
| PPPTMO=3 | Seconds |
| Term=2 | Maximum number of termination requests to send before acknowledging failure |
| ;**SLIP parameters** | |
| MRU=296 | Maximum Receive Unit |
| MTU=296 | Maximum size of a transfer unit (maximum size of an IP packet) |
| ;**LCP options (PPP)** | |
| MRU=1500 | Maximum Receive Unit |
| Magic=Yes | Magic number |
| ReqACCM=0 | ACCM characters (0-31) for reception |
| ACCM0= | Escaped characters (0-31) for emission |
| ACCM1= | Escaped characters (32-63) for emission |
| ACCM2= | Escaped characters (64-95) for emission |
| ACCM3= | Escaped characters (96-127) for emission |
| ACCM4= | Escaped characters (128-159) for emission |
| ACCM5= | Escaped characters (160-191) for emission |
| ACCM6= | Escaped characters (192-223) for emission |
| ACCM7= | Escaped characters (224-255) for emission |
| **;IPCP options (PPP)** | |
| ReqIPAddr=0.0.0.0 | IP address to be used if there is no negotiation of the address |
| NegIPAddr=Yes | Negotiation of the IP address (Yes or No) |
| PeerIPAddr | IP address of the peer machine (PPP server) |
| VJ=0 | Activation or deactivation of Van Jacobson compression of the IP and TCP packet headers (0 or 1) |

# APPENDIX A - NETWARE COEXISTENCE

TCP/IP is a versatile protocol that is able to function in many different network environments, and over different topologies such as Ethernet and Token Ring. As the interconnection of different types of networks becomes more popular, it is increasingly necessary for PCs to be able to run several communications protocols at the same time.

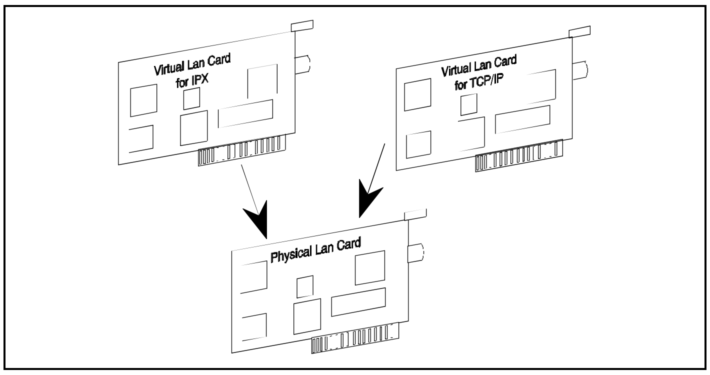
The **Tun KERNEL** Supervisor program automates installation of TCP/IP with Netware, by creating appropriate AUTOTCP.BAT and NET.CFG files. These files, as well as any differences in configuring Netware 3.11 and 3.12, are described in more detail in this chapter.

## USING ODI WITH ETHERNET

### Novell's ODI drivers

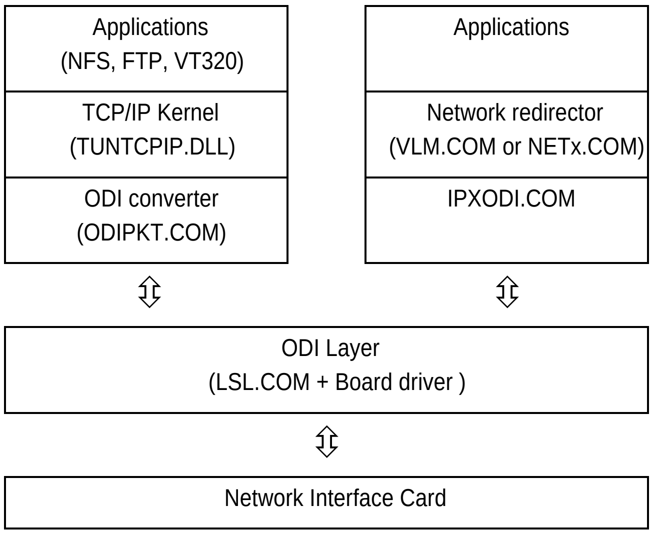
The most common way to configure network cards in PCs with Novell Netware is using ODI drivers (Open Datalink Interface). As ODI has become a firmly-established standard, most LAN cards on the market are delivered with them.

ODI drivers are used to transform a single physical LAN card into two "virtual" LAN cards, each of which is configured to handle a single protocol. TCP/IP and IPX/SPX therefore each address their own virtual LAN cards for network communications.



## ODI Protocol stack

In an **Ethernet** network, TCP/IP communicates using Ethernet II packets, and IPX/SPX uses Ethernet 802.3 packets.



In the above diagram:

1. The ODI layer is provided by Novell Netware's ODI drivers.
2. **Tun KERNEL** includes a packet driver converter, which acts as an interface between the TCP/IP kernel and the ODI drivers.

## Configuration files

The configuration files necessary for loading ODI and several network protocols at the same time are:

| | |
|---|---|
| **NET.CFG** | Delivered by Netware, the card manufacturer, or created using the **Tun KERNEL** Supervisor |
| **CONFIG.SYS** | Necessary for loading TCP/IP parameters into memory |
| **AUTOTCP.BAT** | As configured in **Tun KERNEL**, this batch file may be used or may serve as an example of the programs that need to be loaded. |

## NET.CFG (Ethernet)

The virtual LAN cards are configured through a text file called NET.CFG. The necessary additions in this example for an NE2000 card are given in bold characters:

```
LINK SUPPORT
     buffers 6 1600
     Max boards 4
LINK DRIVER NE2000
     int 5
     port 300
     Frame ETHERNET_802.3
     Frame ETHERNET_II
     Protocol IPX 0 ETHERNET_802.3
     Protocol ODIPKT 8137 ETHERNET_II
```

The NET.CFG file prepares the network card to use the different frame types required by Netware (802.3) and TCP/IP (Ethernet II).

## CONFIG.SYS (Ethernet)

When using **Tun KERNEL** for Windows, there is **no need to change** anything in your current CONFIG.SYS file.

## ODI Startup files (Ethernet)

Dual protocol configurations using ODI require different resident programs to be loaded than single protocol configurations.
For example, an **AUTOTCP.BAT** might execute the following commands:

```
lsl
ne2000
ipxodi
odipkt 1 98
winpkt 0x62
vlm (or netx)
```

This AUTOTCP.BAT file loads the following programs:

| | |
|---|---|
| **lsl(.com)** | Link Support Layer (also called the Link Services Layer); used for routing data packets between Network Interface Cards; also maintains LAN card, protocol, and packet buffer information. Furnished by Novell (in general, try to use the most recent version available). |
| **ne2000(.com)** | The ODI driver for an NE2000 network card; also known as the MLID (Multiple Link Interface Driver); receives and copies packets from the Link Support Layer. **Replace this with the driver for your LAN card** This driver is usually supplied by the board manufacturer. |

As the ODI driver is loaded, a message indicating the presence of both protocols configured in the NET.CFG should be displayed. For example:

```
Int 5, Port 300, Mem CA000, Node Address C01C2665
Max Frame 1514 bytes, Line Speed 10 Mbps
Board 1, Frame ETHERNET_802.3
Board 2, Frame ETHERNET_II
```

Note: The messages produced by the ODI driver are essential. Any difficulties encountered here indicate an incorrect net.cfg.

| | |
|---|---|
| **odipkt 1** | The packet driver for the TCP/IP kernel. The parameter "1" links the odipkt driver to the virtual board #1 (which is really the second virtual card defined in the NET.CFG; the first is #0.) |
| **ipxodi(.com)** | Replaces the file **ipx.com,** traditionally used for Novell-only networks. IPXODI does not need to be generated for a particular LAN card configuration, it addresses the virtual LAN card configured in the NET.CFG. Furnished by Novell (use versions more recent than 1.0). |
| **winpkt.com** | A special "virtual" packet driver translator needed by Windows. |
| **vlm.com** | The Netware workstation shell, for establishing connection with Netware file servers. Replaces the older NETX.COM. Furnished by Novell. |

For more information on ODI installations, please consult the Novell Netware User Manual.

## USING ODI WITH TOKEN RING

The instructions and principles given in the previous section apply to Token Ring environments as well as Ethernet, with one or two minor differences.
Token Ring networks use Token-Ring 802.5 frame types (as opposed to 802.3 for Ethernet); TCP/IP over Token Ring uses Token-Ring_Snap frame types.

### NET.CFG (Token Ring)

The NET.CFG file for a Token Ring installation needs to modified in order to take into account the name of the Token Ring ODI driver:

```
LINK SUPPORT
      buffers 6 1600
      Max boards 4
LINK DRIVER LANSUP
      Frame TOKEN-RING
      Frame TOKEN-RING_SNAP
      Protocol IPX E0 TOKEN-RING
      Protocol ODIPKT E0 TOKEN-RING_SNAP
```

### ODI STARTUP FILES (TOKEN RING)

The AUTOTCP.BAT file is similar to that given in the previous section; however, two lines need to be changed:

```
lsl
lansup (or ibmtoken.com)
ipxodi
odipkt 1 98
winpkt 0x62
```

**vlm** or **netx**

| | |
|---|---|
| **lansup(.com)** | ODI driver for an IBM Token Ring LAN card; used in conjunction with DXMA0MOD.SYS and DXMC0MOD.SYS (described below). Furnished by card manufacturer. In the NET.CFG, the line **LINK DRIVER LANSUP** refers to lansup.com. |

### CONFIG.SYS (TOKEN RING)

As mentioned earlier, the **Tun KERNEL** Supervisor program creates and updates two system files: ESTCPIP.SYS, ESTCPIF.SYS. These files contain IP address information (including Hostname, Gateway Address, Name server, etc.).
In an ODI installation on Token Ring, the **CONFIG.SYS file** also needs to include special Token Ring drivers. In addition, we recommend using the DOS **stacks** command.

For example:

```
shell=c:\command.com /e:1024 /p
files=30
buffers=30
device=c:\xxx\DXMA0MOD.SYS
device=c:\xxx\DXMC0MOD.SYS
stacks=9,512
```

**Note**: The drivers DXMA0MOD.SYS and DXMC0MOD.SYS are furnished by the LAN board manufacturer, and must be present when using LANSUP.COM to access a Token Ring card.

You may also use the ODI driver IBMTOKEN.COM, which does **not** require the use of DXMA0MOD.SYS and DXMC0MOD.SYS to activate a Token Ring card.

## Tun KERNEL, NETWARE & WFWG 3.11

Running **Tun KERNEL**, Novell Netware and Windows for Workgroups at the same time requires a configuration with three network protocols: TCP/IP, IPX/SPX and NDIS.

Before installing **Tun KERNEL**, please make sure that WFWG and Novell are installed and running correctly. If this is the case, then adding a third protocol is very simple. Usually it is best to install Netware first, and then install Windows for Workgroups - declaring Netware (IPX) as an already-existing protocol in the PC.

After installing **Tun KERNEL**, use the Supervisor to select a card configuration with **NDIS Coexistence** in order not to create an appropriate AUTOTCP.BAT file. Usually, the network programs are configured by WFWG to load automatically in the AUTOEXEC.BAT file.

### NET.CFG (Triple protocol)

If Netware and WFWG are running correctly, add the two lines shown in bold type below to the existing NET.CFG file:

```
LINK SUPPORT
      buffers 6 1600
      Max boards 4
LINK DRIVER SMC8000
      int 5
      port 300
      Frame ETHERNET_802.3
      Frame ETHERNET_II
      Protocol IPX 0 ETHERNET_802.3
      Protocol ODIPKT 8137 ETHERNET_II
      Frame Ethernet_802.2
      Frame Ethernet_S NAP
```

### Startup files (Triple protocol)

The order of program execution should then be as follows:

```
c:\windows\net start
cd \windows
lsl
smc8000(ODI driver for an SMC card)
c:\tun\kernelw\drivers\odipkt 1 98
c:\tun\kernelw\winpkt 0x62
c:\windows\odihlp.exe
ipxodi
vlm(or netx)
```

**Note**: Do not run a packet driver from the **TUN\KERNELW\CLARKSON** directory when using this type of configuration.

## USING NDIS DRIVERS (LAN MANAGER)

This section describes a configuration with **Tun KERNEL** and Microsoft LAN Manager using NDIS drivers.

The following lines should be in the CONFIG.SYS file

```
device=c:\lanman\protman.sys /i:c:\lanman
device=c:\lanman\elnkii.sys
device=c:\tun\kernelw\drivers\convert.dos /i:c:\lanman
```

Update the AUTOEXEC.BAT as follows:

```
...
REM == LANMAN == do not change these lines
PATH=C:\LANMAN.DOS\NETPROG;%PATH%
NET START WORKSTATION
LOAD NETBEUI
C:\TUN\KERNELW\WINPKT 0x62
REM == LANMAN == do not change these lines
...
```

Update **C:\LANMAN\PROTOCOL.INI**to include the lines shown in bold type:

```
;MACS
;3Com Etherlink II
[ETHERLINKII]
   drivername = ELNKII$
   interrupt  = 5
   ioaddress  = 0x300
[PROTMAN]
   DRIVERNAME = PROTMAN$
[NETBEUI_XIF]
   Drivername = netbeui$
   SESSIONS   = 6
   NCBS       = 12
   BINDINGS   = ETHERLNKII
   LANABASE   = 0

[PKTDRV]
   Drivername = PKTDRV
   BINDINGS   = ETHERLINKII
   INTVEC     = 0x62
   CHAINVEC   = 0x65
```

# APPENDIX B - WINDOWS FOR WORKGROUPS

**Tun KERNEL** provides automatic installation and configuration for **Windows for Workgroups** (WFWG). This section describes the additions that are made to a standard **Tun KERNEL** setup so that it functions correctly with WFWG.

1. The packet driver protocol is added to the end of the **PROTOCOL.INI** file (read by **NET START**):

```
[network.setup]
...
[protman]
...
[XIRCMAC]
...
[MS$NDISHLP]
...
[NETBEUI]
...
[PKTDRV]
DRIVERNAME=pktdrv$
BINDINGS=XIRCMAC
intvec=0x62
chainvec=0x65
```

2. A packet driver converter is added to the system.ini file ( on a single line):

```
transport=ndishlp.sys,*netbeui,   C:\TUN\KERNELW\DRIVERS\CONVERT.DOS
```

> **Note**: When NET START is run, you should see a message on the screen saying that this driver has been loaded.

As mentioned in the configuration section of this manual, it is highly advisable to install and test Windows for Workgroups before installing **Tun KERNEL** for Windows. Most problems relating to LAN card configuration can be avoided if WFWG is set up correctly to start with.

# APPENDIX C - IP ADDRESSES

## OVERVIEW

IP addresses are logical numbers used to identify each host in a TCP/IP network: every IP address must therefore be unique.

IP addresses may be numbers up to 32 bits long, separated into four (4) 8-bit fields by periods (.). Many millions of possible combinations exist: 128.127.126.10, 60.0.0.1, 200.45.30.1 are examples of valid IP addresses.

In simple networks, selection of IP addresses can be made almost at random (as long as no two nodes are the same). However, there are rules and conventions for organizing larger and more complex networks, including those that will be connected to an "outside world" network such as the Internet or the ARPAnet. Careful assignment of IP address can help divide large networks into smaller segments, thereby reducing total network traffic by keeping data local to each segment most of the time.

## COMPOSITION OF AN IP ADDRESS

There are two parts to an IP address: the **Host portion** that identifies the machine itself, and the **Network portion** that must be the same for all nodes in the same network.

For example:

| | |
|---|---|
| 60.0.0.1<br>60.0.0.2<br>60.0.0.10 | These addresses identify hosts **0.0.1**, **0.0.2**, and **0.0.10**<br>on network **60**. |
| 128.127.0.1<br>128.127.0.2<br>128.127.0.10 | These addresses identify hosts **0.1**, **0.2**, and **0.10** on network **128.127**. |
| 200.50.50.1<br>200.50.50.2<br>200.50.50.10 | These addresses identify hosts **1**, **2**, and **10** on network **200.50.50**. |

The **network portion** of the IP address can take 1, 2, or 3 bytes of the total 4byte address, with the **host portion** taking the remainder. Potential network size (the total number of nodes) depends on how many bytes are available to identify the host.

The number in the first field on the left (specifically the first three bits of that number) indicates the **network class**, and therefore how many fields of the IP address are considered to make up the **network portion**. The table below summarizes the network classes and the number of hosts that may be assigned for each:

| First field | Network Class | Example | Network Portion | Host Portion | Number of Hosts |
|---|---|---|---|---|---|
| 1-126 | A | 40.1.2.3 | 40 | 1.2.3 | 254*254*254<br>(app. 16 million) |
| 128-191 | B | 129.1.2.3 | 129.1 | 2.3 | 254*254<br>(app. 64,000) |
| 192-223 | C | 210.1.2.3 | 210.1.2 | 3 | 254 |

## SUBNET BITS AND SUBNET MASK

To further isolate network segments, it is possible to divide large networks into **subnetworks**. A subnetwork address uses part of the host portion of the IP address; the network portion stays the same. The number of **subnet bits** determines how much of the host portion will be used to identify a subnetwork, and therefore what possible addresses remain for the host portion.

The advantage to using subnetworks is that to the "outside world", the whole network is considered as a single physical network, but within the structure of the network, there are several different segments. This can help organization, and may be used to reduce overall network traffic.

# INDEX