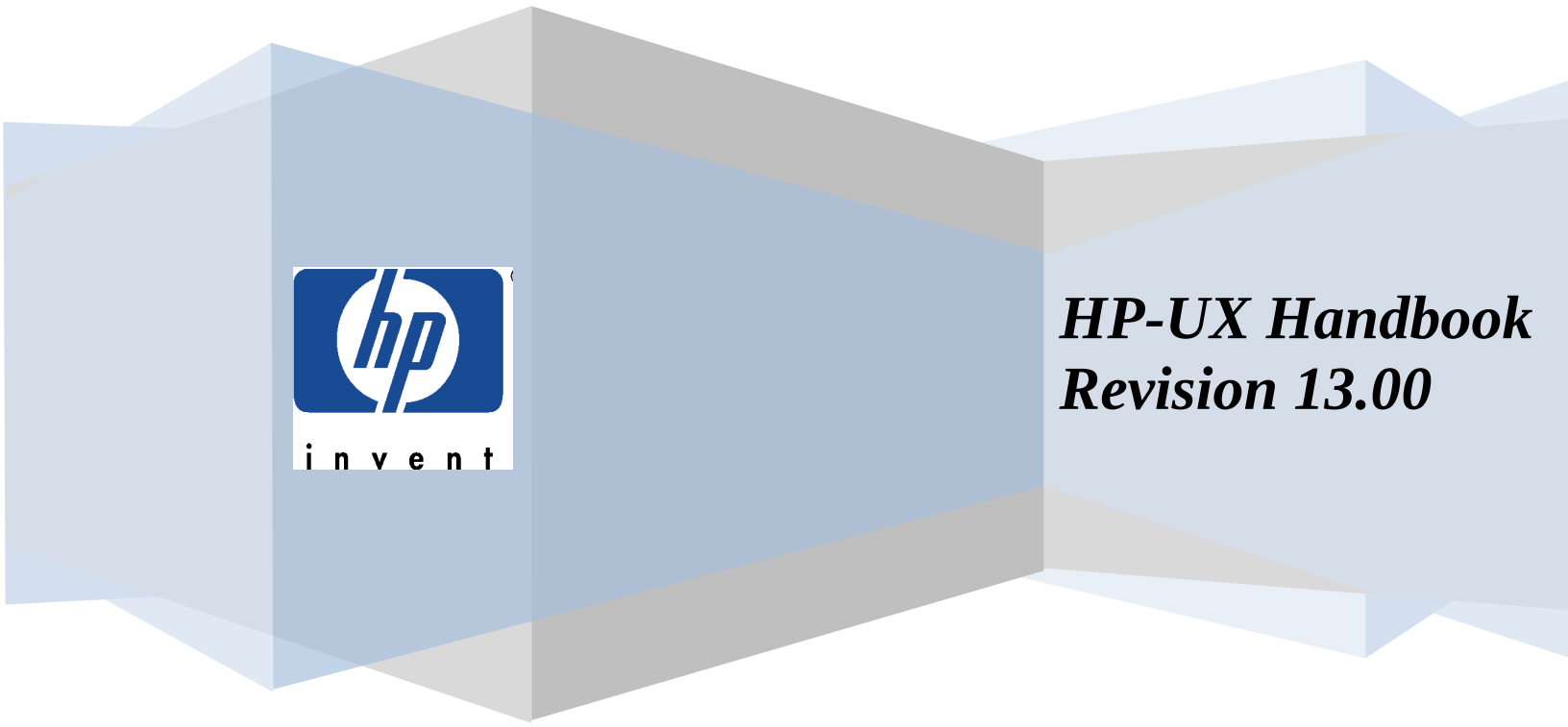
Chapter 09

Network Basics



HP-UX Handbook
Revision 13.00

TERMS OF USE AND LEGAL RESTRICTIONS FOR THE HP-UX RECOVERY HANDBOOK

ATTENTION: PLEASE READ THESE TERMS CAREFULLY BEFORE USING THE HP-UX HANDBOOK. USING THESE MATERIALS INDICATES THAT YOU ACCEPT THESE TERMS. IF YOU DO NOT ACCEPT THESE TERMS, DO NOT USE THE HP-UX HANDBOOK.

THE HP-UX HANDBOOK HAS BEEN COMPILED FROM THE NOTES OF HP ENGINEERS AND CONTAINS HP CONFIDENTIAL INFORMATION.

THE HP-UX HANDBOOK IS NOT A PRODUCT QUALITY DOCUMENT AND IS NOT NECESSARILY MAINTAINED OR UP TO DATE. THE HP-UX HANDBOOK IS HERE MADE AVAILABLE TO HP CONTRACT CUSTOMERS FOR THEIR INTERNAL USE ONLY AND ON THE CONDITION THAT NEITHER THE HP-UX HANDBOOK NOR ANY OF THE MATERIALS IT CONTAINS IS PASSED ON TO ANY THIRD PARTY.

Use of the HP-UX Handbook: Hewlett-Packard Company ("HP") authorizes you to view and download the HP-UX Handbook only for internal use by you, a valued HP Contract Customer, provided that you retain all copyright and other proprietary notices contained in the original materials on any copies of the materials. You may not modify the HP-UX Handbook in any way or publicly display, perform, or distribute or otherwise use them for any public or purpose outside your own business. The materials comprising the HP-UX Handbook are copyrighted and any unauthorized use of these materials may violate copyright, trademark, and other laws. If you breach any of these Terms, your authorization to use the HP-UX Handbook automatically terminates and you must immediately destroy any downloaded or printed materials.

Links To Other Web Sites: Links to third party Web sites provided by the HP-UX Handbook are provided solely as a convenience to you. If you use these links, you will leave this Site. HP has not reviewed all of these third party sites and does not control and is not responsible for any of these sites or their content. Thus, HP does not endorse or make any representations about them, or any information, software or other products or materials found there, or any results that may be obtained from using them. If you decide to access any of the third party sites linked to this Site, you do this entirely at your own risk.

Disclaimer: THE HP-UX HANDBOOK IS PROVIDED "AS IS" WITHOUT ANY WARRANTIES OF ANY KIND INCLUDING WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF INTELLECTUAL PROPERTY. HP further does not warrant the accuracy and completeness of the materials in the HP-UX Handbook. HP may make changes to the HP-UX Handbook at any time without notice. The HP-UX Handbook may be out of date, and HP makes no commitment to update the HP-UX Handbook. Information in the HP-UX Handbook may refer to products, programs or services that are not available in your country. Consult your local HP business contact for information regarding the products, programs and services that may be available to you.

Limitation of Liability: IN NO EVENT WILL HP, ITS SUPPLIERS, OR OTHER ANY THIRD PARTIES MENTIONED IN THE HP-UX HANDBOOK BE LIABLE FOR ANY DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, THOSE RESULTING FROM LOST PROFITS, LOST DATA OR BUSINESS INTERRUPTION) ARISING OUT OF THE USE, INABILITY TO USE, OR THE RESULTS OF USE OF THE HP-UX HANDBOOK, WHETHER BASED ON WARRANTY, CONTRACT, TORT OR ANY OTHER LEGAL THEORY AND WHETHER OR NOT ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THIS DOES NOT APPLY IN CASE OF INTENT OR IF LIABILITY IS LEGALLY STIPULATED. IF YOUR USE OF THE HP-UX HANDBOOK RESULTS IN THE NEED FOR SERVICING, REPAIR OR CORRECTION OF EQUIPMENT OR DATA, YOU ASSUME ALL COSTS THEREOF.

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

Applicable Laws: These Terms will be governed by and construed in accordance with the laws of the State of California, without giving effect to any principles of conflicts of laws.

General: HP may revise these Terms at any time by updating this posting. ***Revised Oct 2013***

FEEDBACK or QUESTIONS: please email essam.ackleh@hp.com
(please use subject syntax: *HP-UX Handbook v13.00 Chapter <YY> - <Feedback Title>*

TABLE OF CONTENTS

Introduction

This document is a short introduction to network configuration and administration on HP-UX systems. It discusses the most common networking tasks, that you might encounter during your daily work but will not cover every task in detail. For additional information please refer to
www.hp.com/go/hpux-networking-docs
Or
http://bizsupport2.austin.hp.com/bc/docs/support/SupportManual/c02492449/c02492449.pdf

A little bit of Theory

Protocols and Models

TCP/IP stands for a set of standards and protocols, the *Transmission Control Protocol* (TCP) and the *Internet Protocol* (IP) being the most important. They are organized into:

Addressing:

IP addresses, which uniquely identify every host on the network, deliver data to the correct host.

Routing:

Gateways deliver data to the correct network.

Multiplexing:

Protocol and port numbers deliver data to the correct software module within the host.

The ***OSI Reference Model*** illustrates the access of applications to the network. This implementation is independent (from operating systems and hardware) and helps a lot in heterogeneous networks. Table 1 shows the standardized terminology; the right column offers some examples for the respective layer.

7	**APPLICATION LAYER** consist of application programs that use the network	(i.e…telnet, ftp, DNS, etc)
6	**PRESENTATION LAYER** standardizes data presentation to the application	(i.e…Mime, TLS, SSL, etc)
5	**SESSION LAYER** manages sessions between applications	(i.e.. RPC Protocol)
4	**TRANSPORT LAYER** provides end-to-end error detection and correction	(TCP or UDP Protocol)
3	**NETWORK LAYER** manages connections across the network for the upper layer	(IP-Adresses, Routing)
2	**DATA LINK LAYER** provides reliable data delivery across the physical link	(MAC-Adresses, Duplex-Mode)
1	**PHYSICAL LAYER** defines the physical characteristics of the network media	(Hardware)

Table 1 - OSI Layer

A network packet (the logical form how the information is transported via a network) contains separate sections for each of these layers (as long as the layer is used in the specific communication). Layers 1 through 4 are organized on behalf of the operating system and/or the TCP/IP protocol. The layers above correspond to the applications.

While layers 1 through 3 are used by all applications, there are big differences in the upper layers. Layer 4 stands for the method to organize a connection. Ordinarily the protocols *UDP* and *TCP* are used here.

UDP stands for *User Datagram Protocol*:

> The User Datagram Protocol gives application programs direct access to a datagram delivery service, e.g. the delivery service that IP provides. This allows applications to exchange messages over the network with a **minimum of protocol overhead**.
>
> UDP is an unreliable, **connectionless** datagram protocol. As noted previously, "unreliable" merely means there are no techniques in the protocol to verify that the data reached the other end of the network correctly. Within your computer, UDP will deliver data correctly. UDP uses 16-bit Source Port and Destination Port numbers in word 1 of the message header, to deliver data to the correct applications process.

TCP stands for *Transmission Control Protocol*. This protocol goes through significantly more expense to safeguard data integrity:

> Applications requiring the transport protocol to provide reliable data delivery use TCP because it **verifies that data is delivered across the network** accurately and in the proper sequence. TCP is a reliable, **connection-oriented**, byte-stream protocol. Let's look at each of the terms – reliable, connection-oriented, and byte-stream – in more detail.
>
> TCP provides reliability with a mechanism called "Positive Acknowledgment with Re-transmission" (PAR). A system using PAR sends the data again, unless it "hears" from the remote system that the data arrived ok. The unit of data exchanged between cooperating TCP modules is called a segment. Each segment contains a checksum that the recipient uses to verify that the data is undamaged. If the data segment is received undamaged, the receiver sends a positive acknowledgment back to the sender. If the data segment is damaged, the receiver discards it. After an appropriate time-out period, the sending TCP module re-transmits any segment for which no positive acknowledgment has been received.

The correlation between the transmitted data and the applications is accomplished by using **port numbers**. Lots of applications/server programs bind themselves to dedicated ports. If this correlation is documented in a standard RFC we talk about so-called **well-known ports**.

HP-UX handles this correlation within the file /etc/services (e.g. telnetd = port 23; ftpd = port 21; ftp-data= port 20; rpcbind = port 111).

A different concept to make ports known to other programs is the "**R**emote **P**rocedure **C**all" technology. There, a server process will register itself with a central "location broker" process which listens to a well known port. The clients can ask this location broker to get the required port information for the other servers. (Examples: SUN-RPC: rpcbind = port 111; DCE-RPC: rpcd/dced = port 135)

IP Addresses and Netmasks

An IP address specifies the network the system resides in and the "number" of the *host* within this network. The IP address consists of a network part and a host part. The network part is important for routing, to find a way through the network(s) to the target. The host part identifies a specific host within its own network.
The netmask defines how many bits of the 4-byte IP address are used for the network part. Usually 3 classes of networks are distinguished using predetermined netmasks and some reserved areas of address range.

1st byte	type	network part	hosts part	netmask
< 128	class A	byte 1	byte 2,3,4	255.0.0.0
128 - 191	class B	byte 1,2	byte 3,4	255.255.0.0
192 - 223	class C	byte 1,2,3	byte 4	255.255.255.0
224 - 239	multicast	n/a	multicast group	
> 239	reserved	n/a	n/a	n/a

Official network addresses have since become a rare commodity and the philosophy of small, medium, and large networks no longer fits the bill of the current field of networking. Subnets and so-called *classless IP addresses* are required nowadays. The theory behind all this is defined in RFCs 1122 and 1878. As a result IP addresses are specified with the number of bits for the network part, e.g. 15.140.10.113/21 (the first 21 bits of this address are the network part).
The following example will show how netmasks are used:
Network administration (IT) provides an official IP address and netmask. Now we would like to determine which subnet this IP-address is located within, and which broadcast address belongs to this subnet.

Example: IP-address: 15.140.10.113 netmask: 255.255.248.0
15.140.10.113/21 (in another notation – see above)

	binary
IP Address	15.140.10.113 0000 1111 . 1000 1100 . 0000 1010 . 0111 0001
Netmask	255.255.248.0 1111 1111 . 1111 1111 . 1111 1000 . 0000 0000

Using the logical AND results in the:

Net portion of the IP address 0000 1111 . 1000 1100 . 0000 1000 . 0000 0000
in decimal: **15.140.8.0**

setting all hostbits to 1 results in the:

broadcast address 0000 1111 . 1000 1100 . 0000 1111 . 1111 1111

in decimal: **15.140.15.255**

IPv6
IPv6 is the next generation network-layer protocol designed to be the successor to the current Internet Protocol version 4 (IPv4). It provides the packet delivery service for TCP, UDP and ICMPv6.
IPv6 has significant advantages over IPv4 in terms of increased address space, simplified header format, integrated QoS support and mandatory security.
IPv6 has extended the address size from 32 bits to 128 bits and they are textually represented in hex-colon notation as x:x:x:x:x:x:x:x, where the x's are the hexadecimal values of the eight 16-bit pieces of the address. For example fedc:83ff:fef6:417a:210:83ff:fef6:3dc0.
Private IP Addresses

If someone needs to build a network with private IP-addresses, with no connection to the Internet/WAN, it is still recommended that the IP-addresses follow the RFC 1918 standard. The following address ranges are exclusively used for private networks and never for the Internet.

Class A:	10.0.0.0	...10.255.255.255
Class B:	172.16.0.0	...172.31.255.255
Class C:	192.168.0.0	...192.168.255.255

Routing

When we discuss routing, we have to differentiate between two aspects:

How does the Operating System find out, which network interface has to be used?
How does the information find its path (from source to destination) through the different subnets?

The following information will cover the first question only.

Note: Always disable gated, arpd or similar programs, if you do not know exactly that they are really needed and how to configure those programs!

You need to be familiar with the following terms, if you would like to understand routing:
Loopback Interface: virtual interface to allow locally running processes to communicate using networking methods without using any real network hardware.
Host Route: route, giving the direction to a single host
Network Route: route, giving the direction to a network
Default Route: the way to any other (not explicit listed) system or network. The default keyword indicates a wildcard route, used as a last resort if no route is specified for a particular remote host

or network.
The systems store the corresponding information within the routing table and we can have a look at this information with the netstat command:

```
# netstat -rn
```

The kernel will refer to this list, when deciding which interface to use to send out a network package.
To interpret the output you should consult the man page:

```
# man routing
```

The route command is used to (re-)configure the routing table on a running system. To make permanent changes (that survive a reboot) enter the changes into the file, **/etc/rc.config.d/netconf**, (using vi or SAM/SMH).

Good to know…
The initialization of an interface with the ifconfig command creates two entries within the routing table of the system:
One host route to the new local IP.
One network route to the subnet the IP belongs to.
The source IP of a package of an outgoing connection will be determined by the routing table (HP-UX 11.X). This might result in funny effects on systems where additional IP addresses on virtual interfaces are defined (e.g. lan0:1).
When initializing the interface, net and broadcast addresses can be specified at the same time. If this information is missing, then the system defaults will be used. The current values used by the system can be checked, using:

`# ifconfig` <interface> also shows netmask and broadcast address

`# netstat -in` also shows the net addresses

`# netstat -rnv` also shows net addresses and netmasks

Starting with HP-UX 11.00, systems use the so-called *Dead Gateway Detection* mechanism the active gateways in the routing table are pinged regularly to check whether they are still active or not. If the gateway does not answer, then the appropriate entry is deactivated (marked "DEAD"). As a result, devices like firewalls or routers, which do not answer to ICMP-ECHO-REQUESTS, cause problems. To deactivate this feature use the command:

```
# ndd -set /dev/ip ip_ire_gw_probe 0
```

Commands and Utilities:

Below is an overview of useful commands and tools.

Basic Configuration Commands

Display a HW scan of all network interfaces:

```
# ioscan -funC lan
```

Display hardware state, paths and names (how the system has recognized the NICs):

```
# lanscan
```

Configure and display software state, IP and network parameters of a NIC:

```
# ifconfig <interface>
```

Display and customize driver settings of interfaces (speed, duplex modi, pmtu):

```
# lanadmin
```

Display lots of statistics and status information:

```
# netstat -[ainrv]
```

Add and delete entries from the routing table in the kernel:

```
# route [add|delete][host|net]
```

Display and customize different kernel settings for networking. (See man page for more detail).

Get the value of the parameter for network_device and print the value to standard output.

```
# ndd -get
```

Set parameter for network_device to value.

```
# ndd -set
```

Basic Utilities

Display and customize the arp table in the kernel:

```
# arp -[ansd]
```

Test layer 2 connectivity (only for interfaces with HP-UX drivers):

```
# linkloop
```

Test layer 3 connectivity:

```
# ping [-o]
```

Test network connectivity and display route to target:

```
# traceroute
```

To specify a network interface to obtain the source IPv4 or IPv6 address for outgoing probe packets. This is normally only useful on a multi-homed host. (See the -s flag for another way to do this.):

```
# traceroute –i
```

This utility is very useful to check connectivity in Serviceguard environments. Download it from,
http://teams3.sharepoint.hp.com/teams/esssupport/InsideESSSupport/InsideWTEC/HA/Pages/Tools.aspx, (HP internal)

```
# dlpiping
```

Installation

Latest installation media and up-to-date installed systems contain most of the required drivers; therefore hardware installations often require no additional software preparation. But as this situation might change – the old rule is still valid:

Install driver software before installing new hardware!

To find out which driver software (and patches) is needed use HP-websites like,
The **HP Support Center** portal, **www.hp.com/go/hpsc**

Or

http://wtec.cup.hp.com/~netux/lanlinks/link_index.html, **(HP internal)**

NIC (network interface card) configuration requires some context information. You will need at least:

- Hostname (a specific name for the system)
- IP address
- Name belonging to this IP

For 100BaseT interfaces you will need information about the settings of the Switch/HUB (speed, duplex mode, auto-negotiation).

Network environment information you will need netmask, broadcast address, default gateway,

nameserver, domainname.

The **initial configuration** of the system should be done using the script;

```
# /sbin/set_parms
```

(man set_parms).

The most important options are:

/sbin/set_parms **hostname**	To change the hostname
/sbin/set_parms **ip_address**	To change the IP address of lan0
/sbin/set_parms **addl_netwrk**	To set up default gateway, declare DNS server, local ARPA Domain Name, NIS Domain
/sbin/set_parms **initial**	Run the entire initial boot-time dialog sequence

Hint: If you have to change the **hostname** of your system, it is easier to use this tool, because it will also change the corresponding information in databases like those of the software distributor utilities (SDU).

Additional interfaces can be configured or changed with SAM/SMH. Run:

```
# /usr/sbin/sam
```
→ Networking and Communication
→ Network Interface Cards

Mark the appropriate interface, choose the required action and add all requested information into the mask.

It is also possible to add the required information to the configuration files using a text editor (e.g. **vi**) and to activate it manually. The system will store all required information about IP addresses, subnets and routers etc. in the file:

/etc/rc.config.d/netconf

Additional information (if required), like speed, duplex settings, auto-negotiation, MTU and MAC addresses can be configured in the files **/etc/rc.config.d/hp*conf**

Unfortunately there is a different file for each class of interfaces (e.g. /etc/rc.config.d/hpietherconf for the iether driver). Use the ioscan command to find the correct driver name and description, for example:

```
# ioscan -fknC lan
Class I  H/W Path   Driver   S/W State    H/W Type     Description
=============================================================================
lan   0  0/1/1/0/4/0 igelan   CLAIMED    INTERFACE    HP A6794-60001 PCI 1000Base-T
lan   1  0/4/1/0     iether   CLAIMED    INTERFACE    HP A7012-60601 PCI/PCI-X
```

1000Base-T Dual-port Adapter

To configure the network on your system you should also check/edit the file **/etc/hosts**.
It should contain at least the line "127.0.0.1 localhost loopback" and one additional line for each local IP address. For interoperability with other name services you should use the following syntax:

<IP address> <full qualified name> <one or more aliases>

Even if it is not mandatory, it is strongly recommended to create/activate the file **`/etc/nsswitch.conf`** to prevent problems caused by the fact that different versions of libc (libc.1, libc.2) behave differently from the side effects of IPv6 enablement.

In a standard IPv4 environment the following entries in nsswitch.conf should be present:

hosts: **files [NOTFOUND=continue] dns**
ipnodes: **files**

Replace dns by nis or ldap or add some additional name resolution services if needed;

```
# man nsswitch.conf
```

If your system uses DNS, (many applications today will use name resolution) – check/create the file **/etc/resolv.conf.**

It has to contain a domain **or** search statement, which defines the local domain name(s) and up to three lines with nameserver definitions. Compare with this simple example:

```
# cat /etc/resolv.conf
search deu.hp.com emea.hpqcorp.com
nameserver 16.58.37.124
```

The init process will use this information during startup (/sbin/rc2.d/S*) to configure the NIC correctly. It will be done using the commands:

`ifconfig` - initialize interface
`route` - add routes
`lanadmin` - change speed/duplex settings

Starting with 11.31 the nwmgr command became available.
nwmgr - network interface management command for LAN and RDMA interfaces.

```
# nwmgr --get
Name/          Interface   Station        Sub-       Interface   Related
ClassInstance  State       Address        system     Type        Interface
```

lan0	UP	0x00306EF30764	igelan	1000Base-T
lan1	UP	0x001A4B066052	iether	1000Base-T
lan2	UP	0x001A4B066053	iether	1000Base-T
lan900	DOWN	0x000000000000	hp_apa	hp_apa

These commands can also be used to change the network settings on a running system. But please keep in mind that some programs/applications may have problems changing the network environment during runtime.
To check your basic installation or to test settings, it helps to configure a new interface manually. Please check the examples (following later in this chapter) on how to use those commands.
Sometimes it is required to change the default network behavior of the HP-UX kernel to interact with specific network environments. This can be done with the following commands:
ndd (HP-UX 11.X)
nettune (HP-UX 10.X)

These commands allow you to change the kernel settings without a reboot. To make the changes permanent (surviving a reboot) on HP-UX 11.X we have to write the appropriate entries into the file.
/etc/rc.config.d/nddconf

Refer to the man page ndd and/or check:
http://teams3.sharepoint.hp.com/teams/esssupport/InsideESSSupport/InsideWTEC/NETUX-XPORT/Pages/NDD%20parameters/nddparams.aspx, **(HP internal)**.

Installation Hints

Here are some descriptions and installation hints to avoid known problems.
Special HP-UX Feature: Usage of RFC 1122 – Subnet Issue
According to RFC 1122 not all possible subnet address combinations are allowed. The HP-UX operating system tests those conditions upon activation of the IP address with ifconfig. You will see ioctl error messages, if this problem occurs. Since newer RFCs impose fewer restrictions, you can turn off this test on HP-UX 11.X with the following ndd command (and corresponding ndd.conf entries):

```
# ndd -set /dev/ip ip_check_subnet_addr 0
```

Duplex Modi
Most of the currently used network interfaces and switches have configurable speed and duplex settings. Even if all involved parties use algorithms to check and set the best values for speed (via *auto-sensing*) and duplex (via *auto-negotiation*) automatically, we have often seen problems with auto-negotiation in the past. Please do not forget the following rules:
A network interface and the switch port to which it is connected **must** use the same settings, either using AUTO or using the same specific value

(10HD/100HD/100FD/1000FD/100000FD…).
If you use 100BaseT interfaces, the whole environment works more stable if you avoid the value AUTO
If you use Gigabit interfaces, it is strongly suggested to use the value AUTO only.
duplex communication system is a system composed of two connected parties or devices that can communicate with one another in both directions.
Ref: http://en.wikipedia.org/wiki/Duplex_(telecommunications)
The term **multiplexing** is used when describing communication between more than two parties or devices.
A **half-duplex** system provides communication in both directions, but only one direction at a time (not simultaneously). Typically, once a party begins receiving a signal, it must wait for the transmitter to stop transmitting, before replying
A **full-duplex** or sometimes *double-duplex* system, allows communication in both directions, and, unlike half-duplex, allows this to happen simultaneously. Land-line telephone networks are full-duplex, since they allow both callers to speak and be heard at the same time
Autonegotiation is an Ethernet procedure by which two connected devices choose common transmission parameters, such as speed, duplex mode, and flow control. In this process, the connected devices first share their capabilities regarding these parameters and then choose the highest performance transmission mode they both support.

Ref: http://en.wikipedia.org/wiki/Autonegotiation

Auto-sensing is and Ethernet procedure by which two connected devices can match the line speed of its link partner but the adapter cannot sense the duplex mode. Auto detection is only a speed sensing mechanism; it does not support the additional features Auto Negotiation provides.
Flow Control - A sending station (computer or network switch) may be transmitting data faster than the other end of the link can accept it. Flow control is used to halt or slow down the transmission of data, (using PAUSE frames) until the receiving side can clear the existing buffers.
MTU - Maximum transmission unit is the size of the largest packet that a network protocol can transmit. Most Ethernet traffic is transmitted using a MTU size of 1500 bytes. Jumbo frames can be anywhere in the range of 1500 – 9000 depending on the implementation in the network.
Path MTU Discovery a technique for determining the path MTU between two IP hosts. The path MTU is the largest packet size that can traverse this path without suffering fragmentation.
In Ethernet, a "**duplex mismatch"** is a condition where two connected devices operate in different duplex modes, that is, one operates in half duplex while the other one operates in full duplex. The effect of a duplex mismatch is a network that works but is often much slower than its nominal speed. Duplex mismatch may be caused by manually setting two connected network interfaces at different duplex modes or by connecting a device that performs Autonegotiation to one that is manually set to a full duplex mode.
Ref: http://en.wikipedia.org/wiki/Duplex_mismatch
More than one physical interfaces
IP addresses of different active physical interfaces **must** belong to different logical subnets. Even if some experienced administrators are capable of configuring their environment differently, we cannot support this setup. Instead, if there is a need for different physical interfaces to handle

communication within the same logical subnet, you should use "HP AUTO-PORT Aggregation" software (aka APA) as a supported solution. Additionally, APA can provide load balancing and high availability.

Dead Gateway Detection

Starting with HP-UX 11.00, systems use the so-called *Dead Gateway Detection* mechanism, i.e. the active gateways in the routing table are pinged regularly to see if they are still active. If there is no answer from the gateway, then the appropriate entry is deactivated (marked DEAD). As a result, devices like Firewalls or Routers, which do not answer to ICMP-ECHO-REQUESTS, cause problems. To deactivate this feature use the command:

```
# ndd -set /dev/ip ip_ire_gw_probe 0
```

Private IP addresses

If you have to build a private network, it is still recommended that the IP addresses follow the RFC 1918 standard. The following address ranges are exclusively used for private networks and should never be used within the public internet:

1	Class A network number:	10.0.0.0	
2	Class B network numbers:	172.16.0.0	...172.31.0.0
256	Class C network numbers:	192.168.0.0	...192.168.255.0

Manual configuration – an example

The following steps could be used as a guideline for testing and configuring NICs manually. First, check whether the interface is recognized by the kernel with ioscan –funC lan. It should show you the hardware path and driver information. Then, run

```
# lanscan |grep lan1
0/1/2/0  0x00306EF372E5 1   UP    lan1 snap1     2   ETHER     Yes     119
```

This command returns the name and the "card instance number" = "Crd In #" of the NIC required for further configuration and tests. It will also show the "Station Address", (MAC address), of the card.

Now, we check what is configured on this interface:

```
# netstat -in |grep lan1
```

No output means, the interface is recognized by the system, but it is not configured yet. Getting no response is what we expect. We can configure an IP address on this interface:

```
# ifconfig lan1 10.10.10.111 netmask 255.255.255.0 up
```

After that, all commands should recognize the interface

```
# ifconfig lan1
```
lan1: flags=1843<UP,BROADCAST,RUNNING,MULTICAST,CKO>
inet 10.10.10.111 netmask ffffff00 broadcast 10.10.10.255

```
# netstat -in |grep lan1
```
lan1 1500 10.10.10.0 10.10.10.111 0 0 0 0 0

```
# netstat -rnv |grep lan1
```
10.10.10.111/255.255.255.255 10.10.10.111 UH 0 lan1 4136
10.10.10.0/255.255.255.0 10.10.10.111 U 2 lan1 1500

The kernel defines two routes for each physical and virtual interface automatically; one to organize communication with the IP address locally and another to define the way to local subnets.
If we need more than one IP address on this physical interface, we can simply add it with the ifconfig command:

```
# ifconfig lan1:1 10.10.10.111 netmask 255.255.255.0 up
```

We have to check, if anything is reachable within out new network. We send packages to the broadcast address of this network:

```
# ping 10.10.10.255
```
PING 10.226.91.255: 64 byte packets
64 bytes from 10.10.10.30: icmp_seq=0. time=0. ms
64 bytes from 10.10.10.14: icmp_seq=0. time=0. ms
64 bytes from 10.10.10.38: icmp_seq=0. time=0. ms

At least our own system has to answer. If you get lots of answers, then the network is up and running. The above simple test requires no additional duplex settings, but for data transfer it is important.
Check the speed/duplex/Autonegotiation settings with:

```
# lanadmin –x 1
```
Speed = 1000 Full-Duplex.
Autonegotiation = On.

Use the card instance number from the `lanscan` output to address the appropriate interface. The same command used with option "-x" allows us to make changes. Some older drivers do not recognize the "-x" option, instead use "-s"/"-S" for those interfaces.
To transmit data outside of your local subnet you should be able to ping the gateway:

```
# netstat -rn
```
Routing tables

Destination	Gateway	Flags	Refs	Interface	Pmtu
127.0.0.1	127.0.0.1	UH	0	lo0	32808
10.10.10.30	10.10.10.30	UH	0	lan0	32808
10.10.10.0	10.10.10.30	U	2	lan0	1500
127.0.0.0	127.0.0.1	U	0	lo0	32808

default 10.10.10.254 UG 0 lan0 1500 ←

```
# ping 10.10.10.254
```

Checking network connectivity

After installation or whenever somebody suspects a network problem, we have to check whether the network is up and running. The following table lists some examples on how to gather the needed information. (There might be other ways to collect the same data. Please feel free to use own methods!)

Check	Expected Results	Conclusion elsewhere
ioscan -funC lan	appropriate "Driver" information and "S/W State" == CLAIMED	Install correct driver and check hardware
Lanscan	It shows us name, MAC-address and Card-Instance number of interfaces (== Crd In#)	If there is no MAC check HW
netstat –in	For each network interface an own line which shows name, MTU, network- and IP-address and statistics if there was some traffic at the line	If there is no info for interface, configuration with ifconfig is required
ifconfig <name >	It shows IP- and broadcast address, netmask and state flags "UP" and "RUNNING"	Run "ifconfig … up" with appropriate options again and check line
netstat –rn	at least for each configured interface one host route (Flags "UH") with its own IP as destination and gateway and one network route to local subnet with its own IP as gateway	Run "ifconfig … up" with appropriate options again
Now test connection ping <broadc.-addr> -n 100	If there are some other systems in the discussed network we have to see answers of different systems	Check network information – netmask and broadcast address - and configuration at hub/switch

lanadmin –g "Crd In#"	Look for speed settings and statistics	"Operation Status: down" == no link ; "FCS/Aligment" errors == bad line / hardware; some Collisions == no full duplex; no inbound or outbound packages == check which interface is used by traffic
nettl –status	Size >0 and location of nettl-logfiles	Run "nettl -stop;nettl -start"
ll /var/adm/nettl.LOG00*	No current change at this files	Format this log file with netfmt to check which event was logged
Dmesg	Last entry of dmesg should be memory information	If there are add. messages check what happened
ipfstat –io	Check if IPFilter is installed and if we have to discuss limitations by rules ; "ipfstat: not found." == not installed == no problem by this SW	Configure IPFilter in an appropriate way
/usr/sbin/ipsec_admin –status	Check if IPSec product is installed and if we have to discuss limitations by rules; "not found" == not installed == no problem	Configure IPSec in an appropriate way
arp –an	Some arp-table entries but no error messages, correct MAC/IP mapping	If you get there some errors messages check it and react in an appropriate way
/etc/rc.log	Check it for error messages	Do the required steps to avoid the messages

Discussing some known problems

Problem: After a reboot the network and the network interfaces are not configured
Before you start "the big troubleshooting procedure", check /etc/rc.log first for any relevant information **AND** check and ensure that the file systems /tmp and /are not full (message: file system full). Our system requires some space for temporary files in /tmp to initialize the network stack. If either /tmp or / is full during the boot process, clean it up and restart the system.

Problem: We send a ping request to the broadcast address and get an answer even if the cable is not plugged in.
Please be aware that if we activate an IP at a NIC we get an answer from this interface to any ping request to this IP and to the broadcast address of the subnet, even if there is no cable connected. We need no cable to reach our local interface. We can check the connectivity to the subnet only through communication to any other system.
When you ping the broadcast check for responses from IP addresses other then the local interface you are configuring.

Problem: If we check our routing with the traceroute command, we do not see the expected behavior
Please be aware that all tools have limitations. We have to check our tool first and then the environment.
Example: If the traceroute command tells you "multiple interfaces found", when it starts, then you know that this specific binary version does not automatically use the routing table of the system. We have to specify the interface using the "-i" option.
Otherwise traceroute will use the first "visible" interface and we cannot test the routing situation with this tool. If there are only "*" signs in the output instead of time stamps, either your routers or target system do not send ICMP messages or somewhere in your network a firewall blocks the ICMP messages.

Kernel configuration with ndd

HP-UX gives us lots of opportunities to tune the network stack for performance improvement and adaptation to different environments. The behavior of the kernel can be changed with the ndd command.

```
# ndd -h
```

This will show you the available kernel parameters. You will get additional information for each parameter, if you run the command with the option "-h" and the specific parameter, e.g.

```
# ndd -h ip_ire_gw_probe
```

Enable dead gateway probes. This option should only be disabled on networks containing gateways which do not respond to ICMP echo requests (ping).

[0-1] Default: 1 (probe for dead gateways)

The command:

```
# ndd -get /dev/<device> <parameter>
```

will show you the current value of the parameter and you can change it with:

```
# ndd -set /dev/<device> <parameter> <value>
```

We can deduce the device that we have to use, by looking at the first part of the parameter names. For example:

```
# ndd -get /dev/ip ip_ire_gw_probe
1

# ndd -get /dev/tcp tcp_fin_wait_2_timeout
0
```

Please be aware, that changes done with ndd, will only be done in the running kernel (in memory) and will not survive a reboot. To make those changes permanent, you should use the file **/etc/rc.config.d/nddconf**. For each parameter you have to add three lines – like this:

```
TRANSPORT_NAME[0]=<device>
```

```
NDD_NAME[0]=<parameter name>
NDD_VALUE[0]=<value>
```

Set a unique ordinal (sequential number) in brackets. Starting with "0", you have to increase this number by one, for each new entry. If you run:

```
# ndd -c
```

the ndd program will re-read the configuration file and activate the changes within the running kernel. During startup, the system runs this script in /sbin/rc2.d/S340net (and in some other scripts).

Common changes done with ndd

Disable router ability of the system – IP forwarding
If there is more than one network interface in your HP-UX system, it will work as a router. To disable this feature, run

```
# ndd -set /dev/ip  ip_forwarding 0
```

Disable the gateway probe mechanism
If you have to use a gateway/router in your network, which does not answer to ICMP ECHO requests (e.g. a firewall), you will see that your HP-UX system automatically disables the routing definitions using this gateway after a short period of time. The system usually checks the existence of gateways with these ICMP requests and if it does not answer, the system "assumes" that the gateway is not available. The routing entry to this gateway is then marked as "dead". To disable this mechanism use:

```
# ndd -set /dev/ip ip_ire_gw_probe 0
```

Disable the test for compliance with RFC 1122
The HP-UX kernel will automatically check a subnet mask for its RFC 1122 compliance. Because newer RFCs are less restrictive, you can turn off this test with:

```
# ndd -set /dev/ip ip_check_subnet_addr 0
```

Define a FIN_WAIT_2 timer
The TCP protocol specification explains that we have to wait for the last ACK in communication before closing a TCP session. That is why there is no timeout definition per default for the FIN_WAIT_2 status of a port which was used for TCP communication.
Unfortunately, some systems do not send such a last ACK package, or sometimes it gets lost in the network. If you see lots of ports with status FIN_WAIT_2 in the output of the netstat -an command, it makes sense to define a timeout (e.g. waiting 10 minutes and then closing the port without receiving the last ACK):

```
# ndd -set /dev/tcp tcp_fin_wait_2_timeout 600000
```

There is a very nice web page with additional information at:
http://teams3.sharepoint.hp.com/teams/esssupport/InsideESSSupport/InsideWTEC/NETUX-

XPORT/Pages/NDD%20parameters/nddparams.aspx, **(HP-Internal).**

Trace tools

If there are any problems or unexpected effects in the network it is often really helpful to check the network communication directly. There are some easy to use programs available for this purpose. This is a short introduction to the most common tools:

nettl

The nettl command is a tool to monitor problems within the network and to trace the complete network communication. It is included in each HP-UX installation; it is the "ready to use" tool for administrators to do first checks. Unfortunately, there are some limitations. First, there are different versions available. Newer versions have more options to adapt it to the needs of current hardware generations. Please check the man page. Second, it is sometimes too slow to capture all traffic on 1Gbit or faster networks. Nevertheless, here is a short introduction to this tool. It is usually started by the `init` process with the script:

```
/sbin/rc2.d/S200nettl
```

When started by this default procedure, it logs problematic events in network only. You can configure it, if required, in /etc/rc.config.d/nettl.
The output of the command:

```
# nettl -status
```

shows if it is running, location and size of the log files, and which subsystems are observed for "ERROR" and "DISASTER" events. The line "No Subsystems Active" in the output means that currently only problematic events are observed, and the tracing part of nettl is not active.
To capture network packages, use the following steps:
Check, if nettl is running:

```
# nettl -status
```

if not yet started, start it with:

```
# nettl -start
```

Start tracing:

```
# nettl -tn pduin pduout -e ns_ls_ip -tm 99999 -f /tmp/my-trc
```

Run the command/process which should be observed and stop tracing directly afterwards, with the command:

```
# nettl -tf -e all
```

The trace tool writes a binary output in one or two files /tmp/my-trc.TRC0*. It uses a circular output file method, which means, as soon as one file fills up, another file is used. As soon as the second file fills up, the first file gets overwritten. The size of both output files together is limited

to the value of the "-tm" option.
You can use the –n option to specify the number of output files. The default is 2.

```
# nettl -tn pduin pduout -e ns_ls_ip -n 8 -tm 99999 -f /tmp/my-trc
```

We have to format it to get a readable trace output or to view the binary output with some viewer tools like ethereal (see below). We can format the binary output files from nettl (both – trace and log files) with the netfmt command. The simplest way to do this is the following command line:

```
# netfmt -nNl -f /tmp/my-trc.TRC00 >  /tmp/my-trc.TRC00.txt
```

If we have to check for specific traffic, it helps to use some filters. A filter is a text file containing filter conditions. That is to see all traffic from or to a specific interface we can use the condiction "ip_saddr" (= IP source address) and "ip_daddr" (= IP destination address).
We create a text file called "filter" with the following two lines:

```
filter ip_saddr 10.10.10.111
filter ip_daddr 10.10.10.111
```

and start the netfmt command with this filter as an additional option:

```
# netfmt -c filter -nNl -f /tmp/my-trc.TRC00 > /tmp/my-trc.TRC00a.txt
```

The output of this command contains only those network packages which have the named IP addresses as source or destination.

For available filter options, please refer to the man page of the netfmt command.
Some options which we can use for nettl:

Option		Explanation
-tn		== "trace on" ; it starts tracing
-tf		== "trace off", it stops tracing
-e <entity>		It defines from which part of network stack the information is taken. "–e ns_ls_ip" takes it from IP-layer which is enough for most issues. If you use "–e all" you will see your packages in each part of the network stack. To find the available entities to trace use the option –ss, i.e. # nettl -ss
-c <interface>		nettl by default takes all packages of all interfaces. This option allows you to restrict it to one specific interface.
-m <# byte>		Normally nettl catches all bytes of the network packets but often it is enough to see just header information. This option allows you to specify the number of bytes to be captured in the trace packet. The maximum value for bytes is 2000.
-tm <maxsize>		It defines the maximal size of output of trace. The default value for combined file sizes (maxsize) is 1000 KB. The possible range for maxsize is 100 to 99999.

There are a lot of additional options for nettl and netfmt, please read the man pages to get the

appropriate information.

Additional hints:

To get a first impression of what is going on within the network, we can combine nettl and netfmt, even though UNIX pipes are to slow to trace a really busy network.

```
# /usr/sbin/nettl -tn pduin pduout -e ns_ls_ip | netfmt -F1TN | tee file
```

This shows you what happened on the network and writes it to file. You have to use the kill command to stop this pipe and to run:

```
# /usr/sbin/nettl -tf -e all
```

to stop tracing afterwards.

Older documentations use ns_ls_driver to get more informative traces. Unfortunately, the current driver generation does not use this part of the network stack anymore. Therefore, you will have to use more specific entities, such as "BTLAN" or something similar, to catch packages at the driver level.

If you like graphical interfaces, you can use the command /opt/nettladm/bin/nettladm. It starts a GUI for nettl and netfmt.

tcpdump

A very useful trace tool for networks is tcpdump. It is a small, open-source, easy to use program. You can find the latest version and background information at: http://www.tcpdump.org.

You can find software depots with binary versions for HP-UX available at:

http://software.hp.com,
under "Security and manageability", HPUX Internet Express for HP-UX.
(Install libpcap as well which is available on the same server).

Here are some examples on how to use tcpdump (which is located at `/usr/sbin/tcpdump or /usr/contrib/bin/tcpdump`).

To will catch everything:

```
# tcpdump -nnn -v -x -s 256 -w /tmp/tcp.out
```

Check what happens at interface lan1, show each incoming package, even if we have no appropriate IP address:

```
# tcpdump -n -i lan1
```

Show the DNS traffic (port 53) running via lan1 to the name server with IP address 15.137.16.1

```
# tcpdump -p -n -i lan1   host  15.137.16.1 and port 53
```

These commands write (-w option) only one line of information per network package. Often those one-liners do not contain enough information. Therefore we better write a raw trace to disk (using the "-s" option to specify how much data should be stored – "0" means all).
Write all data of the network packages of the DNS traffic to disk:

```
# tcpdump -p -s 0 -i lan1 -w /tmp/dns-trc.out host 15.137.16.1 and port 53
```

The output file (/tmp/dns-trc.out) contains raw (binary) data. We have to format the file, before we can read it. This can be done with the tcpdump command, too:

```
# tcpdump -n -r /tmp/dns-trc.out -vvv
```

Analysis of those trace files is much more effective and a lot easier, if you install a graphical tool like Ethereal or Wireshark and open the trace file within this tool. Those tools also provide a lot of online help and make reading and understating the trace easier.

Useful options:

Option		Explanation
-p		Disables promiscuous mode. **ATTENTION**: Some network drivers dislike changes to the promiscuous mode, therefore ALWAYS use this option on productive systems!!!
-n		Useful only for display. It disables name resolution within the output.
`-s` <number>		Defines the number of bytes, which will be saved to disk from each package. The default value is 64. Use 0 if all bytes of the packages should be saved.
`-i` <name>		Name of interface where we like to trace
`-w` <file>		Output file for trace data
`-r` <file>		Read data from file and do not trace
-v[vv]		Display more information
host <IP>		Filter statement, collect only packages from and to this IP
port <port#>		Filter statement, collect only packages from and to this port number

The big advantage of tcpdump is the flexibility of the filter expressions. You can use "AND" and "OR" statements to combine different filters. So it will ensure to save only the needed data to disk (as you can see in the example). Please check the man pages and documentation for additional information.

ethereal and wireshark

Another very usefully tool from the open-source community is the program ethereal/wireshark. Please don't be confused about those two names. It is really the same software, only in different versions. Formerly, the program was called "ethereal", but due to some legal restrictions it now uses the name "wireshark". Full documentation, the latest version and more is available at http://www.wireshark.org. If you try to use it on an HP-UX system, you should use the depots from http://software.hp.com, under "Security and manageability", HPUX Internet Express for HP-UX.

You can use ethereal/wireshark as a trace tool. It is a very strong tool for formatting, analyzing and interpreting network traces. Using ethereal/wireshark is the simplest way to analyze a network trace.

The tool is always capable of opening the network trace files created with nettl or tcpdump, regardless of the platform it is installed on (HP-UX, Windows, Linux…). It is a strong protocol analyzer, contains a lot of easy to use filter function, and also gives you statistical information for performance discussions. As you could expect from such a graphical tool, you will find a voluminous online help.