IPv6/VSE IPv4

Installation Guide

Current Build

Table of Contents

Preface

About this Publication

This is the **IPv6/VSE IPv4 Installation Guide**. The manual will introduce you to IPv6/VSE and provide you with the information necessary to install and startup the IPv6/VSE IPv4 TCP/IP stack.

Trademarks

The following are lists of the trademark and products referenced in this manual. Symbols for trademarks and registered trademarks do not appear in subsequent references.

Barnard Software, Inc.

TCP/IP-TOOLS is a registered trademark of Barnard Software, Inc.
IPv6/VSE is a registered trademark of Barnard Software, Inc.

International Business Machines Corporation

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at **www.ibm.com/legal/copytrade.shtml**.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Copyrights

This software and documentation is covered by the following copyright:

Portions of this software are covered by the following copyright:

Redistribution and use in source and binary forms are permitted provided that this notice is preserved and that due credit is given to the copyright holders. The names of the copyright holders may not be used to endorse or promote products derived from this software without specific prior written permission. This software is provided 'as is' without express or implied warranty. The authors assume no liability for damages incidental or consequential, nor is the software warranted for correctness or suitability for any purpose.

Portions of this software are documented in the book:

```
Comer, D. E. [2006], "Internetworking with TCP/IP Vol 2:
       Design, Implementation, and Internals," Prentice-Hall,
       Upper Saddle River, New Jersey.
```

This software may not be sold or published in printed form without written permission from the copyright holders.

zlib-1.2.5 Compression Library

(C) 1995-2010 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

Technical Support

IBM Customers

IBM IPv6/VSE customers should contact IBM for support.

BSI Customers

Technical Support is available from Barnard Software, Inc. by phone, mail or email:

Barnard Software, Inc.
806 Silk Oak Terrace
Lake Mary, FL 32746

Phone: 1-407-323-4773

Support: bsiopti@bsiopti.com

Sales: bsisales@bsiopti.com

Support is available from 9:00 a.m. through 5:00 p.m. EST, Monday through Friday.

If a TSR (Technical Support Representative) is not available at the time of your call, please leave a message and a TSR will return your call as soon as possible. Please provide the following information: name, company, phone number, product name, product release level, and a short description of the problem.

BSIUsers Announcement List Server

When new releases of IPv6/VSE are available BSI will post an announcement on its BSIUsers announcement list.

To subscribe to the BSIUsers announcement list send an email to this email address

BSIUsers-subscribe@yahoogroups.com

To unsubscribe to the BSIUsers announcement list send an email to this email address

BSIUsers-unsubscribe@yahoogroups.com

Problem Determination

If you have a problem using an IPv6/VSE application always check the SYSLST output for additional information and messages. Most messages are written to SYSLST and not to the VSE/ESA system console.

When contacting BSI for technical support always have the applications JCL/commands, console and SYSLST output available for problem determination. The SYSLST output is very important.

While an IPv6/VSE application is running, you can issue the **AR CANCEL XX,PARTDUMP** command to terminate the IPv6/VSE application and dump the partition to SYSLST. Using the VSE/POWER Flush (F) command cancels the IPv6/VSE application partition without a dump.

If the IPv6/VSE application partition stops responding to its console interface, use the **AR DUMP XX** command to obtain a dump of the partition.

Chapter 1

Installation

System Requirements

Operating System

- BSI supports IPv6/VSE on all versions and releases of VSE/ESA and z/VSE including ...
  VSE/ESA 1.3, 1.4 and VSE/ESA 2.1 through 2.7
  z/VSE 3.1, 4.1, 4.2, 4.3, 5.1, 5.2 or higher
- IBM IPv6/VSE 1.1 is supported by IBM on z/VSE 5.1 and 5.2.

VSE/ESA 1.3/1.4 Restrictions

- Only a single BSTTINET TCP/IP stack partition can be run
- BSTTINET must hook the IBM $IJBSEOT routine in the SVA
- BSTTINET must be started after all other vendor software
- BSTTVNET is limited to 100 sessions per server

System GETVIS

IPv6/VSE applications do not allocate any VSE system GETVIS. All data transfers and access to the TCP/IP stack (BSTTINET) partition is done using access registers. No 24-bit or 31-bit System GETVIS is allocated by any IPv6/VSE application.

SVA Phases

No IPv6/VSE phases should be placed in the SVA.

Partition Execution

IPv6/VSE can run in any partition, including dynamic partitions.

Partition Priority

The priority of the IPv6/VSE partition should be set higher than partitions processing batch jobs but may be set lower than the TCP/IP partition's priority.

Virtual Partition Size

Partition size is dependent on the application being used. Formulas are provided in the chapter describing the application. If a formula is not provided a 8MB or larger partition is required.

Installation of the INSTTOOL.JOB File

The IPv6/VSE download is a .zip file. When the .zip file is unzipped it will contain a file called INSTTOOL.JOB. The INSTTOOL.JOB file is a BINARY file with 80 bytes fixed length records. This job will catalog the IPv6/VSE phases and library members.

Using TCP/IP

FTP the INSTTOOL.JOB file to the VSE/POWER RDR queue using a BINARY transfer. If you are using the CSI/IBM TCP/IP for VSE product will will want to rename the INSTTOOL.JOB file to INSTTOOL.BJB prior to any FTP.

Use the VSE/POWER R RDR,INSTTOOL command to release the INSTTOOL job for execution. The INSTTOOL job will pause to allow you to set various SETPARM statements.

Using IND$FILE

Transfer the INSTTOOL.JOB file to the VSE/POWER RDR queue. Specify BINARY and NOUC (NO UpperCase) for the transfer.

Customize the IPv6/VSE BSTTPARM.A Member

The BSTTPARM.A member of the installation library contains the COMPANY and LICENSE parameters provided by Barnard Software, Inc. and required to run the IPv6/VSE products.

After customizing the BSTTPARM.A member please copy the BSTTPARM.A member to a configuration library. We recommend using PRD2.CONFIG to hold this member.

After IPv6/VSE starts executing it reads two files. The first is the BSTTPARM.A member and the second is from SYSIPT. The first file contains the IPv6/VSE COMPANY and LICENSE command statements. The second file contains IPv6/VSE command statements to be processed. IPv6/VSE considers any statement with an asterisk (*) in column one to be a comment.

Startup Commands to Customize

COMPANY

Enter your company name in the name field of the COMPANY command statement. The company name specified on this statement is passed to the LICENSE command statement processing routines to validate the verification code provided by Barnard Software, Inc.

CPUID

Enter your system's CPU serial number and the model number on the CPUID command statement. The CPU serial and model number specified on this statement are passed to the LICENSE command statement processing routines to validate the verification code provided by Barnard Software, Inc.

LICENSE

Enter the product expiration date in the EXPDATE field on the LICENSE command statement. The format of the EXPDATE field is full-year Julian (for example, 1994365). IPv6/VSE will begin issuing messages warning of product expiration 45 days prior to the actual expiration date.

Enter the verification code in the VCODE field on the LICENSE command statement. The format of this field is seven numeric digits (for example, 1234567).

The parameters on the COMPANY, CPUID and LICENSE statements are combined to validate the verification code. If an error is detected in processing these statements an error message is issued and the product continues to initialize. However, unless the verification code is validated, IPv6/VSE cannot be enabled by the TCP/IP-TOOLS or IPV6/VSE ENABLE command.

After initialization completes, you can reissue the COMPANY, CPUID and LICENSE commands through the console interface, if necessary.

TCP/IP-TOOLS ENABLE

The TCP/IP-TOOLS ENABLE command should be the last startup command to be processed. Without this command TCP/IP-TOOLS will not initialize within the partition.

Trial Mode

When the IPv6/VSE TCP/IP stack is started without customizing the BSTTPARM.A member the product enters 'Trial Mode'. When trial mode is active the product will function for 30 days in a limited performance mode. Once you have received your official LICENSE information and customize the BSTTPARM.A member the product will function at full performance for the duration of the license.

Trial mode is indicated by messages appearing on the console

```
R1 0045 // JOB INETOSA6
        DATE 02/22/2010, CLOCK 15/24/59
R1 0045 1T20I  SYS000 HAS BEEN ASSIGNED TO X'FEE' (TEMP)
R1 0045 BSTT000I INITIATED  BSTT6NET Build248 02/15/10 16.24    EP=00520078
R1 0045 BSTT003I COPYRIGHT (C) 1998-2010 BARNARD SOFTWARE, INC.
R1 0045 BSTT700I IPv6/VSE BUILD 248PRE23
R1 0045 BSTT004I CB=TTLA A=0052E000 L=000013FC
R1 0045 BSTT019I VSE 8.20 MODE 31-BIT
R1 0045 BSTT004I CB=COMR A=002E44F0 L=00000108
R1 0499 BSTT000I INITIATED  BSTTX6PC Build248 01/27/10 18.36    EP=00831000
R1 0499 BSTT025W LICENSE WILL EXPIRE IN     31 DAYS
R1 0499 BSTT710I * - - - - - - - - - - - - - - - - - - - - *
R1 0499 BSTT709I TRIAL MODE DETECTED, LIMITING PERFORMANCE
R1 0499 BSTT710I * - - - - - - - - - - - - - - - - - - - - *
R1 0499 BSTT045I TCP/IP ID SET TO 66
R1 0497 BSTT000I INITIATED  BSTT6SRV Build248 02/20/10 07.20    EP=0084EB80
R1 0497 BSTT000I INITIATED  BSTT21EP Ver 2.46 04/01/09 11.27    EP=803279D0
R1 0497 BSTT600I     0 INITIATED  ipboot   Ver 2.48 10/12/09 08.02
```

LIBDEF's used by IPv6/VSE Programs

All of our TCP/IP applications run with our TCP/IP stack and CSI/IBM's TCP/IP stack. Basically, you must run a TCP/IP stack. You must libdef the IPv6/VSE lib.slib and the stack lib.slib. If they are the same then only one library needs to be in the search chain.

```
// LIBDEF PHASE,SEARCH=(toollib.slib,tcplib.slib)
// LIBDEF SOURCE,SEARCH=(parmlib.slib,toollib.slib)
```

Where …

Toollib.slib is the IPv6/VSE library.sublibrary.

Tcplib.slib is the TCP/IP library.sublibrary.

Parmlib.slib is the library.sublibrary containing the BSTTPARM.A member.

Upper Case SYSLOG/SYSLST Output

IPv6/VSE normally uses mixed case SYSLOG (console) and SYSLST (printer) output. This output can be converted to all upper case by using the UCMSG SETPARM. Using the SETPARM to set the variable UCMSG to 'YES' will cause output to be converted to upper case. If the SETPARM is not specified or UCMSG is set to a value other than 'YES' no conversion will occur.

```
// SETPARM UCMSG='YES'
```

Quiet Console Mode

IPv6/VSE normally writes various message to the VSE/ESA system console. This output can be disabled and all message written to SYSLST using the MSGMODE SETPARM.

```
// SETPARM MSGMODE='QUIET'
```

OS390 Execution Mode

Some of IPv6/VSE programs take advantage of the ',OS390' parameter on the // EXEC JCL statement. The ',OS390' parameter is valid and should be used on VSE/ESA 2.4 and newer systems. This parameter should be omitted from VSE/ESA 1.4, 2.1, 2.2 and 2.3 systems.

Commands Read From SYSIPT

All commands and parameters read from SYSIPT by IPv6/VSE applications are full 80 column card images. If you use an editor that uses columns 72-80 for sequence or line numbers you must disable this feature. Sequence or line numbers in column 72-80 will be treated by IPv6/VSE applications as part of the command or parameter.

Chapter 2

IPv6/VSE Features

IPv6/VSE provides a high performance IPv4 TCP/IP stack and a full suite of TCP/IP applications.

The TCP/IP stack support a number of network interfaces including LCS3172, OSA, OSA/2, IUCV, CLAW, CTCA, OSA Express and Hipersocket interfaces.

The application suite includes many useful applications including a batch FTP client, FTP server, TN3270E server, batch LPR, Remote Execution client, GZIP utility, and much more. See the IP Users Guide for details on IPv6/VSE application suite.

Chapter 3

TCP/IP Communications Stack

BSI's IPv6/VSE includes a TCP/IP communication stack. The BSTTINET TCP/IP stack runs in a stand-alone partition (static or dynamic). No TCP/IP applications (FTP, TN3270E, etc) run in the TCP/IP stack partition with the exception of the Simple Network Management Protocol (SNMP) server. The SNMP server allows external SNMP client applications monitor the activity in the TCP/IP stack. Although the SNMP server could have been written to run outside of the TCP/IP stack partition it was easier to implement within the TCP/IP stack partition.

The BSTTINET TCP/IP communications stack runs in 31-bit mode and makes extensive use of access registers. The BSTTINET application will only run on VSE/ESA systems running in ESA mode. 370 mode VSE/ESA systems are not supported.

The BSTTINET TCP/IP stack is optimized for dynamic partition usage. For the best possible performance run the BSTTINET TCP/IP stack in a dynamic partition and run all TCP/IP applications in dynamic partitions also.

Usage of the BSTTINET TCP/IP stack requires a verification code in the BSTTPARM.A member with feature code I.

Compatibility

The BSI TCP/IP stack was designed to be as compatible as possible with the IBM/CSI TCP/IP stack. Applications written using the IBM/CSI API should run using the BSI TCP/IP stack without any changes or re-assemblies. Programs using the IBM/CSI API and the BSI TCP/IP stack must LIBDEF the IPv6/VSE installation library prior to any library containing IBM/CSI phases.

Programming

The best information on coding TCP/IP from a mainframe CICS COBOL program can be found in some of the OS/390 manuals as OS/390 has had this functionality for several years.

A "MUST" book is the Redbook "A Beginner's Guide to OS/390 TCP/IP Socket Programming", GG24-2561-00. It walks you though both batch and CICS Socket programming using the "EZASOKET" interface. Of the 350 pages, 200 pages are sample code. We have actually copied the CICS Cobol Server example (using"Cut and Paste") and after removing the RACF call, ran it in VSE using the BSI TCP/IP stack.

For a technical reference, the manual "IBM TCP/IP for OS/390 Application Programming Interface Reference, Version 3 Release 2", SC31-7187-03 is the best guide. BSI has implemented 99% of the EZASOKET HLL interface. The only areas not implemented are a few items in the SETSOCKOPT area that were intended for use only by IBM subsystems.

An additional technical reference for the EZASOKET HLL interface can be found in the new VSE/ESA manual "z/VSE TCP/IP Support". Although VSE/ESA folks wrote this manual, the OS/390 manual is better. The EZASOKET interface delivered with VSE/ESA 2.5 is only a subset of the OS/390 API. There are several limitations to the VSE/ESA 2.5 interface that are not limitations in the BSI

implementation. The BSI implementation does not violate the VSE/ESA 2.5 implementation, so if you code to the VSE/ESA 2.5 manual, you will be compatible with the BSI TCP/IP stack.

The BSTTEZA.C member of the IPv6/VSE installation library is a COBOL copybook containing definitions of all the parameter that might be used by the EZASOKET HLL interface and for which a length has been defined within the EZASOKET specification.

Sample EZASOKET Programs

We provide both CICS and batch sample EZASOKET programs. The CICS EZASOKET sample programs provide a generic listener that runs under any version of CICS (CICS/VSE 2.3 and CICS T/S). The CICS program DFHEIQSK must be defined to CICS. An AEY9 abend results if this program is not available.

We provide a listener startup transaction, listener shutdown transaction, the listener transaction itself and a sample processing transaction. All you really have to modify is what the processing transaction is going to do.

There are 4 transactions

- EZ10 starts the listener (EZ11)
- EZ11 waits for a connection
  When a connection is made EZ11 starts the processor transaction (EZ12)
  and passes the socket to it.
- EZ12 processes the input, does work and sends the response.
- EZ13 stops the listener

All the programs are written in COBOL using the EZASOKETs API. Simple to use, easy to understand and very high performance/low overhead. Everything runs under any version CICS. Batch versions of these programs are provided too.

BSTTEZ10.C, BSTTEZ11.C, BSTTEZ12.C and BSTTEZ13.C are the CICS source code members. BSTTEZ01.C, BSTTEZ02.C and BSTTEZ03.C are the batch source code members.

Manuals

```
These manuals are included in your TCP/IO-TOOLS download.
EZAP7003.BOO TCP/IP for OS/390 Application Programming Interface Reference
EZ306700.BOO A Beginner's Guide to OS/390 TCP/IP Socket Programming
```

Tracing

EZASOKETs Trace flag settings

```
// SETPARM IPTRACE='xxxxxxx'
'Y.......'  produce call parms trace information
'.Y......'  produce iprpl trace information
'..Y.....'  produce console messages on entry and exit
'...Y....'  produce trace information on SYSLST, not direct to LST
queue
'....Y...'  produce waitlist trace information
'.....Y..'  produce one line entry and exit messages trace
information
'......Y.'  Force 'Y.......' for any call which results in an error
'.......Y'  Trace full SEND/RECEIVE buffer
All flags can be used in combinations to create multiple output
entries.

Normally, all trace information (except console traces '..Y....') are
written directly the VSE/POWER LST queue with a job name of
'EZALOGxx' (xx is the partition identifier). With heavy processing,
this XPCC communication can slow the IP processing to the point that
time based failures occur. The '...Y...' flag can help as it
instructs the trace facility to use a faster write to partition
SYSLST. But, by writing to SYSLST, the trace will be interspersed
with normal job output messages and may actually cause an overprint
of the program output. (The tracing function uses "write before
advancing one line'.)
```

External Packet Capture Facility

The CAPTURE command is used to enable and disable the external packet capture facility. When the packet capture facility is in use, packets are written to the BSTTCAP VSAM ESDS file. The DLBL for the BSTTCAP file can be added to System Standard Labels or to the JCL used to run the stack. When the CAPTURE ON command is used, the BSTTCAP file is automatically open when the stack processed the next packet. When the CAPTURE OFF command is used, the BSTTCAP file is automatically closed after the next packet is processed by the stack. The BSTTCAP file is reset each time it is opened.

After trace data has be captured in the BSTTCAP file, a data can be FTP'ed to a PC in BINARY mode and viewed using Ethereal/Wireshark.

Allocating the BSTTCAP File

The following is a sample job stream for allocating the BSTTCAP file

```
// DLBL IJSYSUC,'VSESP.USER.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=IDCAMS
   DELETE IPV6.VSE.PACKET.CAPTURE.FILE CLUSTER PURGE
   DEFINE CLUSTER -
   (NAME(IPV6.VSE.PACKET.CAPTURE.FILE) -
    SPEED -
    REUSE -
    SHR(2 3) -
    NONINDEXED -
    RECSZ(1500 18425) -
    VOL(SYSWK1))-
   DATA(NAME(IPV6.VSE.PACKET.CAPTURE.FILE.DATA) CISZ(18432) CYL(10 0))
   LISTC ENT(IPV6.VSE.PACKET.CAPTURE.FILE) ALL
/*
```

The highlighted fields in the above sample JCL may be changed. The VSAM ESDS cluster name, volume and number of cylinders to allocate are user modifiable. The CISZ should not be changed. Each cylinder allocated will hold slightly less than 1MB of trace data. Therefore, a cylinder allocation of 100 cylinders will hold a little less than 100MB of trace data.

Add BSTTCAP to Standard Labels

This sample JCL will add a label to System Standard labels. You should also add the label for the BSTTCAP file to your normal system IPL procedure.

```
// JOB STDLABEL
// OPTION STDLABEL=DEL
BSTTCAP
/*
/&
// JOB STDLABEL
// OPTION STDLABEL=ADD
// DLBL BSTTCAP,'IPV6.VSE.PACKET.CAPTURE.FILE',,VSAM,CAT=VSESPUC
/*
/&
```

Capturing Data

The CAPTURE command is used to enable (ON) or disable (OFF) the capture of data.

To enable the external packet capture facility ...

```
MSG BSTTINET,D=CAPTURE ON
```

The BSTTCAP file will automatically open when the next packet is processed by the stack.

To disable the external packet capture facility ...

```
MSG BSTTINET,D=CAPTURE OFF
```

The BSTTCAP file will automatically close after the next packet is processed by the stack.

Transferring BSTTCAP Data for Viewing

The following sample BSTTFTPC JCL can be used to transfer the BSTTCAP file to a remote host for viewing by the ethereal/wireshark packet viewer.

```
// EXEC BSTTFTPC,SIZE=BSTTFTPC
ID 66
OPEN 192.168.1.60
USER userid
PASS password
*
CWD directory
INPUT VSAM BSTTCAP
TYPE I
PASV
STOR file.name.pcap
*
QUIT
/*
```

The hilighted fields in the above sample JCL can be modified by you for your installation.

You can now use wireshark/ethereal to open the packet capture file.

Firewall

IPv6/VSE contains a basic firewall security facility. The firewall examines IPv4 and IPv6 Ethernet IP packets for basic types of information. The source IP address, packet protocol, TCP or UDP port numbers and ICMP message type and code can be verified and processing accepted or denied.

The firewall is controlled by library member named BSTTFWRT.T and this member contains the firewall rules table. The stack accesses this member through the LIBDEF SOURCE,SEARCH chain. By default, the firewall is enabled. However, the default firewall rules allow all packets to be processed by the stack.

When a packet is denied processing by the stack, it is dropped. This is usually referred to as 'stealth' mode or firewall stealth operation. No response to the packet is generated.

To disable the firewall simply delete the library member BSTTFWRT.T. Deleting the BSTTFWRT.T member will result in an error message written to the stack's SYSLST log during startup.

Firewall commands specifying IP6 are ignored by the BSTTINET IPv4 stack. Commands specifying IP are ignored by the BSTT6NET IPv6 stack. TCP, UDP and ICMP rules are common to both stacks.

The firewall in the BSTT6NET IPv6 stack always allows various ICMPv6 traffic. This traffic is required for the functioning of an IPv6 stack. This traffic includes packets to the Link Local IPv6 address and various ICMPv6 traffic required by Neighbor Discovery and Router/Gateway processing.

For IP PROT, IP6 PROT, TCP, UDP and ICMP a value of zero indicates 'match any value'.

Typical firewall rules should follow the sequence of commands shown in this manual.

PROT values are 6 (TCP), 17 (UDP), ICMP (1) and ICMPv6 (58).

Currently only Inbound (IN) rules are processed. The syntax for Outbound (OUT) rules is defined but not currently implemented.

There are two methods for defining firewall rules. Both work. The first method is to deny those types of packets you do not want and allow all the remaining traffic. And, the second method (which we recommend) is to allow only the types of traffic you want and deny everything else. The Sample Firewall Rules shown in this section contains examples of both types.

Firewall rules are processed in the following order. IP/IP6 source IPv4/IPv6 address and protocol are validated. Then TCP, UDP and ICMP/ICMPv6 rules are validated. If at any time during rules processing a deny is detected the packet will be dropped. If an allow is detected processing continues with the next type of rule. Only 64 rules of each type are currently permitted.

When a packet is dropped a message (BSTT738W) is written to the stack's SYSLST log.

```
22-Oct-2014 08:08:29 E4 BSTT738W DROPPED  34649    21     6 fd00:806:1::3
22-Oct-2014 08:08:32 E4 BSTT738W DROPPED  34649    21     6 fd00:806:1::3
22-Oct-2014 08:08:38 E4 BSTT738W DROPPED  34649    21     6 fd00:806:1::3
```

In the BSTT738W message after the 'DROPPED' literal appears the source port number, destination port number, IP protocol number and IPv4 or IPv6 source address. In the case of ICMP/ICMPv6 message traffic the port numbers are replaced with the ICMP/ICMPv6 type and code values.

FIREWALL Commands

MSG XX,D=FIREWALL RELOAD
causes the BSTTFWRT.T member to be reloaded.

MSG XX,D=FIREWALL LIST
causes the firewall table to be display on the console and SYSLST log.

MSG xx,D=FIREWALL LOGLEVEL n
causes the firewall message log level to be displayed or changed. The default value is 4 and results in a single line written to the SYSLST log when a packet is dropped. The value 0 results in no logging. The value 8 causes the packet to be dumped in addition to the single line written to the SYSLST log.

```
msg inetosa,d=firewall reload
AR 0015 1I40I  READY
R6 0050 BSTT010I FIREWALL RELOAD
R6 0504 BSTT011I COMMAND PROCESSING COMPLETE
msg inetosa,d=firewall list
AR 0015 1I40I  READY
R6 0050 BSTT010I FIREWALL LIST
R6 0504 BSTT737I IN  IP   A IP 0.0.0.0 0.0.0.0
R6 0504 BSTT735I IN  IP   A PROT     0
R6 0504 BSTT735I IN  TCP  A PORT     0
R6 0504 BSTT735I IN  UDP  A PORT     0
R6 0504 BSTT736I IN  ICMP A TYPE     0 CODE     0
R6 0504 BSTT011I COMMAND PROCESSING COMPLETE
msg inetosa,d=firewall loglevel
AR 0015 1I40I  READY
R6 0050 BSTT010I FIREWALL LOGLEVEL
R6 0504 BSTT060I LOGLEVEL SET TO     4
R6 0504 BSTT011I COMMAND PROCESSING COMPLETE
```

Firewall Rule Syntax

```
*
* IPV6/VSE FIREWALL RULE TABLE
*
* IN|OUT IP   ALLOW|DENY 'IP' IPV4ADDR MASK
* IN|OUT IP   ALLOW|DENY 'PROT' 0
* IN|OUT IP6  ALLOW|DENY 'IP' IPV6ADDR MASK
* IN|OUT IP6  ALLOW|DENY 'PROT' 0
* IN|OUT TCP  ALLOW|DENY 'PORT' 0
* IN|OUT UDP  ALLOW|DENY 'PORT' 0
* IN|OUT ICMP ALLOW|DENY 'TYPE' 0 'CODE' 0
*
* IN|OUT IP  COMMANDS ARE IGNORED BY THE IPV6 STACK
* IN|OUT IP6 COMMANDS ARE IGNORED BY THE IPV4 STACK
*
```

Default Firewall Rules

These firewall rules are used in the default BSTTFWRT.T member.

```
*
* IPV6/VSE FIREWALL RULE TABLE
*
* IN|OUT IP   ALLOW|DENY 'IP' IPV4ADDR MASK
* IN|OUT IP   ALLOW|DENY 'PROT' 0
* IN|OUT IP6  ALLOW|DENY 'IP' IPV6ADDR MASK
* IN|OUT IP6  ALLOW|DENY 'PROT' 0
* IN|OUT TCP  ALLOW|DENY 'PORT' 0
* IN|OUT UDP  ALLOW|DENY 'PORT' 0
* IN|OUT ICMP ALLOW|DENY 'TYPE' 0 'CODE' 0
*
IN IP  ALLOW IP 0.0.0.0 0.0.0.0
IN IP6 ALLOW IP ::0     ::0
*
IN IP  ALLOW PROT 0
IN IP6 ALLOW PROT 0
*
IN TCP  ALLOW PORT 0
IN UDP  ALLOW PORT 0
*
IN ICMP ALLOW TYPE 0 CODE 0
```

Sample Firewall Rules

```
*
IN IP ALLOW IP 127.0.0.1   255.255.255.255
IN IP ALLOW IP 192.168.1.0 255.255.255.0
IN IP ALLOW IP 10.1.1.0    255.255.255.0
IN IP DENY  IP 0.0.0.0     0.0.0.0
*
IN IP6 ALLOW IP FD00:806:1::0  /48
IN IP6 ALLOW IP FD00:806:2::0  /48
IN IP6 ALLOW IP FD00:806:3::0  /48
IN IP6 ALLOW IP FD00:806:4::0  /48
IN IP6 DENY  IP ::0        ::0
*
* IN IP  ALLOW IP 0.0.0.0 0.0.0.0
* IN IP6 ALLOW IP ::0     ::0
*
IN IP  ALLOW PROT 6
IN IP  ALLOW PROT 17
IN IP  ALLOW PROT 1
IN IP  DENY  PROT 0
*
IN IP6 ALLOW PROT 6
IN IP6 ALLOW PROT 17
IN IP6 ALLOW PROT 58
IN IP6 DENY  PROT 0
*
IN TCP  ALLOW PORT 0
*
*  UDP PORT 1900  SSDP UPNP PACKETS
*  UDP PORT 17500 DROPBOX LAN SYNC PACKETST
IN UDP  DENY  PORT 1900
IN UDP  DENY  PORT 17500
IN UDP  ALLOW PORT 0
*
IN ICMP ALLOW TYPE 0 CODE 0
```

Converting CSI/IBM EXEC TCP Applications to EZASOKET

BSTTPREP.PROC is a migration program used to convert COBOL source from the CSI/IBM EXEC TCP API to the EZASOKET socket interface. BSTTPREP is supplied as a VSE REXX PROC. BSTTPREP is intended only as a one-time conversion program.

BSTTPREP can be run in VSE using the following JCL:

```
// EXEC REXX=BSTTPREP,SIZE=256K
<COBOL source goes here>
/*
```

VSE output is directed to SYSPCH.

BSTTPREP can also be run under VM (after transporting to VM via a LIBR PUNCH job) using the following:

BSTTPREP filename filetype filemode

VM output is a new CMS file on the “A” disk with the same filename.
‘BSI’ will be appended to the filetype.

BSTTPREP will convert all “EXEC TCP” statements found in the PROCEDURE SECTION to calls to the EZASOKET call interface. BSTTPREP will also partially convert all “EXEC TELNET” statements to a combination of EZASOKET calls and calls to the data translation routines “EZACIC04” and “EZACIC05”. With “EXEC TELNET” statements, the programmer should review the code as data translation is only one of the differences between the “EXEC TELNET” and the standard “EXEC TCP” calls.

BSTTPREP will NOT convert any CSI “EXEC UDP”, “EXEC FTP”, “EXEC CLIENT”, “EXEC CONTROL”, or “EXEC RAW” statements. As a note, the EZASOKET interface does support, but in a very different manner, most of the functions for which the “EXEC CONTROL” API was used. The EZASOKET interface also supports UDP functions, but due to the complexities of using UDP connections, it was considered outside the scope of BSTTPREP.

Upon processing the program source, BSTTPREP will place the literal “BSI” in columns 1-3 of any source line that was modified. Additionally, BSTTPREP will place “BSI-W’ or “BSI-E” in columns 1-5 of any statements that are warnings or error messages. In those cases, the message will be in columns 8-72. Warning messages will contain a “*” in column 7 so that the program will compile, but error messages will not so as to cause a compile error.

When the program being migrated contains multiple instances of “EXEC TCP OPEN” and “EXEC TCP CLOSE” commands, the code should be reviewed in these areas. A single program should only do one “INITAPI” and “TERMAPI” call to the EZASOKET interface. BSTTPREP will insert multiples of these function calls, as it does not understand the program flow enough to know which open would be executed first and which close would be executed last.

Using the LE/C API ($EDCTCPV.PHASE)

CICS TS, MQSeries, VSE Connector Server and DB2 use the LE/C Sockets API. The LE/C API requires using the BSI $EDCTCPV.PHASE instead of the default (IBM supplied) $EDCTCPV.PHASE. IBM IPv6/VSE customers must rename the BSTTTCPV.PHASE (supports IPv4 only) or BSTTTCP6.PHASE (supports IPv4 and IPv6) to $EDCTCPV.PHASE in the BSI lib.slib.

Users of DB2/VSE should use the EZA or ASM SOCKET APIs provided by IBM. BSI strongly recommends using the ASM SOCKET API for performance reasons.

Activate the BSI $EDCTCPV.PHASE using these instructions:

```
1) Download and install the latest update.
   Check with BSI for information on the latest update.

2) Remove IBM EZASOHxx phases from the SETSDL.PROC
   EZASOH00.PHASE, EZASOH03.PHASE, ESASOHTR.PHASE

3) Change your JCL
   // LIBDEF PHASE,SEARCH=(bsilib.slib,...)   BSI first
   // SETPARM IPTRACE='NNNNNNNN'              Disable all traces
   // OPTION SYSPARM='nn'                     Set stack ID

4) Add // EXEC BSTTWAIT,SIZE=BSTTWAIT
   before // EXEC for you application
   BSTTWAIT will wait for BSTTINET to startup

5) If you are using DB2, set parameter TCPDISPB to 50 to 100

6) DB2 and the VSE Connector Server require
   // SETPARM SENDALL=YES to be added to the JCL.

   TCPDISPB info can be found at
   http://www-3.ibm.com/software/data/db2/vse-vm/v72T3/TCPIPauto/index12.htm
```

TCP/IP EZA Application Considerations

Compiling and Assembling TCP/IP EZA Applications

Compiling or Assembling requirements are dependent on the level of VSE.

For Pre-VSE 2.5, the BSI sublibrary must be available during the Compiling or Linking process.

For later systems, but prior to z/VSE 4.2 plus APAR DY47077 (IBM IPv6 Support), if the program is using any IPv6 functions, the BSI sublibrary must be available during any Assemblies using the EZASMI macro.

Running TCP/IP EZA Applications

For Pre-VSE 2.5, the BSI sublibrary must be available during program execution.

For later systems, but prior to z/VSE 4.1, the BSI sublibrary must be LIBDEFed prior to any IBM libraries during program execution and the EZASOH00 phase must be removed from the SET SDL list.

For z/VSE 4.1 and z/VSE 4.2 prior to APAR DY47077, it is strongly recommended that the BSI sublibrary be LIBDEFed prior to any IBM libraries during program execution and that the EZASOH00 phase be removed from the SET SDL list. Alternatively, the IBM Multiple-Vendor EZA Interface may be used. This interface requires that the IBM version of EZASOH00 be LIBDEFed prior to the BSI version of EZASOH00. Please see the information below about the IBM Multiple-Vendor EZA Interface for setup instructions.

For z/VSE 4.2 plus APAR DY47077 (IJBOSA IPv6 Support), it is recommended that the BSI sublibrary be LIBDEFed prior to any IBM libraries during program execution and that the EZASOH00 phase be removed from the SET SDL list. Alternatively, the IBM Multiple-Vendor EZA Interface may be used. The IBM Multiple-Vendor EZA Interface is required for those partitions executing programs that utilize both the IPv6/VSE and TCP/IP for VSE products. This interface requires that the IBM version of EZASOH00 be LIBDEFed prior to the BSI version of EZASOH00. Please see the information below about the IBM Multiple-Vendor EZA Interface for setup instructions.

Known users of the EZA API include: CICS TS and VSE/POWER.

IBM Multiple-Vendor EZA Interface

With z/VSE 4.1, IBM introduced a new EZA Interface to better support multiple TCP/IP products on a single z/VSE partition. The use of this interface does require either additional JCL statements or the passing of additional information during the INITAPI EZA call. The following information summarizes what may be found in 'z/VSE TCP/IP Support’ manual. More details may be found in that manual. In

addition, for CICS environments, this interface also requires the activation of the EZA 'TASK-RELATED-USER-EXIT' (TRUE). Please see the chapter ‘CICS Considerations for the EZA Interfaces’ in the above manual for details.

BSI added support for this interface with Build 2.48. Do not use this interface with prior Builds.

To use this interface, in addition to providing the Stack-ID, the name of the applicable Vendor Interface Routine must also be passed to the EZA Interface. This is accomplished by using one of the following methods:

1. Pass the name of the Vendor Interface Routine during the INITAPI call by specifying the name in the ADSNAME parameter.
2. Using a setparm: // SETPARM EZA$PHA='routine-name'

When these are required varies depending on the z/VSE environment.

In the case where all stacks are licensed from BSI, the use of a system SETPARM during IPL is recommended: “// SETPARM SYSTEM EZA$PHA='BSTTIPS1' ”. This removes any requirement that the vendor routine be specified within each partition or via an INITAPI call.

In the case where stacks from different vendors exists within a z/VSE image, how the stacks are being used must be considered. If a specific partition is only accessing a stack(s) from one vendor, they will need to specify the correct SETPARM as required by the stack vendor. This removes any requirement that the routine name be specified during the INITAPI call.

The most complicated situation is where a program(s) within a single partition access multiple stacks from multiple vendors. Because there is only one SETPARM EZA$PHA setting within a single partition, and because it is an override, not a default, the SETPARM EZA$PHA must **NOT** be used. All programs **MUST** provide the ADSNAME values by using an INITAPI call as the first EZA call. In addition, any program not using the default stack as indicated by the SYSPARM value, MUST provide the correct TCPNAME value. To address some of the maintenance and support issues with actually coding a vendor phase name within every program, BSI provides an optional BSI TCP/IP EZA API Multiplexer described below.

BSI TCP/IP EZA API Multiplexer

The BSI TCP/IP EZA API Multiplexer is an optional use feature. By using the Multiplexer, the programmer is no longer required to know or code the name of the vendor routine normally required by the IBM Multiple-Vendor EZA Interfaces as delivered with z/VSE 4.1. Additionally, it allows a use of site specific stack TCPNAMEs that do not need to be of the form “SOCKETnn” when specified during an INITAPI call. The BSI EZA Multiplexer is "vendor independent" and will properly handle requests for non-BSI stacks.

The Multiplexer is controlled by a system wide table that is generated as an SVA eligible phase that is named BSTTCPMC. The Multiplexer Control Table is designed to also be used at some point in the future by LE/C and BSD Multiplexers and thus includes some information that is not currently used. A

sample job is delivered in the BSI sublibrary under the name BSTTCPMC.Z. The following is an excerpt from that sample:

```
BSTTCPMC CSECT
BSTTCPMC RMODE ANY
BSTTCPMC AMODE ANY
* SAMPLE BSTTCPMC MULTIPLEXER CONTROL TABLE
         BSTTCPME SYSID='00',TCPNAME='$DEFAULT',                      X
               LEPHASE='XXXXXXXX',EZAPHASE='EZASOH00',                X
               BSDPHASE='YYYYYYYY'
         EZATCPME SYSID='33',TCPNAME='LFP',                           X
               LEPHASE='XXXXXXXX',EZAPHASE='IJBLFPEZ',                X
               BSDPHASE='YYYYYYYY'
         BSTTCPME SYSID='66',TCPNAME='IPV6',                          X
               LEPHASE='XXXXXXXX',EZAPHASE='BSTTIPS1',                X
               BSDPHASE='YYYYYYYY'
         END
```

Each invocation of the BSTTCPME macro provides the information related to a specific stack. Multiple entries can indicate the same stack SYSID allowing multiple TCPNAMEs to reference the same TCP/IP stack. A special entry for TCPNAME='$DEFAULT' is available and will be used with the programmer provides neither a TCPNAME or SYSID. All operand values must be specified within single quotes as shown in the example.

SYSID=

The SYSID specifies the two character system ID specified during the TCP/IP stack startup.

TCPNAME=

The TCPNAME operand specifies a name that can be used by a program as passed in the INITAPI TCPNAME field value. A special TCPNAME of '$DEFAULT' may be used to indicate a default stack when one is not specified by the programmer or if the program passes a TCPNAME that does not exist in the table.

EZAPHASE=

The name of the EZA Vendor Interface Routine as specified by the specific stack vendor. For BSI stacks, the correct name is 'BSTTIPS1'. At the time of this writing, the correct name for the CSI TCP/IP for VSE product is 'EZASOH99'. Please refer to the documentation provide by each vendor for this value.

LEPHASE=

The name of the LE/C Vendor Interface Routine as specified by the specific stack vendor. This field is currently not used, but must be specified. Any value may be used.

BSDPHASE=

The name of the BSD/C Vendor Interface Routine as specified by the specific stack vendor. This field is currently not used, but must be specified. Any value may be used.

The generated table phase must be placed in the SVA or available via LIBDEFs for any partition using

the Multiplexer. If the table is not located as execution time, the program will receive an ERRNO value of 20114. Once the table is loaded by the first INITAPI call within a partition, it will not be refreshed until end of step even if a replacement phase is cataloged.

The actual multiplexer phase name is BSTTCPMX. It must be specified in one of two ways:

1. Using a system wide setting via “// SETPARM SYSTEM EZA$PHA='BSTTEZMX' ”
2. Specifying it's name in each partition using “// SETPARM EZA$PHA='BSTTEZMX' ”

If a SETPARM is not located, the IBM Multiple-Vendor EZA Interface will use “EZASOH99” which will produce undesired results. The phase BSTTCPMX must be available via the LIBDEF for the partition and must not be loaded into the SVA. If the phase is not available at execution time, the program will receive an ERRNO value of 20113.

LE/C TCP/IP Application Considerations

Note: This interface, as defined by IBM, does not currently support IPv6 connections.

Running LE/C TCP/IP Applications

This interface is not available prior to VSE/ESA 2.3.

For Pre-VSE 2.3, the BSI sublibrary must be available during program execution.

For later systems, but prior to z/VSE 4.3, the BSI sublibrary must be LIBDEFed prior to any IBM libraries during program execution and the customer must rename the BSTTTCPV.PHASE (supports IPv4 only) or BSTTTCP6.PHASE (supports IPv4 and IPv6) to $EDCTCPV.PHASE in the BSI lib.slib.

1. If using IPv6/VSE Build 2.47 or earlier, the BSI LE/C TCP/IP Application Interface requires that the EZASOH00 phase be removed from the SET SDL list. Please review the EZA TCP/IP Application Considerations elsewhere in this manual as the EZA interface may still require that the EZASOH00 phase be removed from the SET SDL list. the BSI phase and the EZASOH00 phase must be removed from the SET SDL list.

For z/VSE 4.3 and later, it is recommended that the BSI sublibrary be LIBDEFed prior to any IBM libraries during program execution and the customer must rename the BSTTTCPV.PHASE (supports IPv4 only) or BSTTTCP6.PHASE (supports IPv4 and IPv6) to $EDCTCPV.PHASE in the BSI lib.slib. Alternatively, the IBM LE/C TCP/IP Multiplexer may be used. Please see the information below about the IBM LE/C TCP/IP Multiplexer for setup instructions.

Independent of the version of z/VSE, when using IPv6/VSE Build 2.47 or earlier, the BSI LE/C TCP/IP Application Interface requires that the EZASOH00 phase be removed from the SET SDL list. Please review the EZA TCP/IP Application Considerations elsewhere in this manual as the EZA interface may still require that the EZASOH00 phase be removed from the SET SDL list. the BSI phase

and the EZASOH00 phase must be removed from the SET SDL list.

Known users of the LE/C API include: CICS TS, MQSeries, VSE Connector Server, DB2 and the SNMP Agent for z/VSE.

IBM LE/C TCP/IP Multiplexer

Information on setup and use of the IBM LE/C TCP/IP Multiplexer can be found in the following manual:

> IBM Language Environment for z/VSE – C Run-Time Library Reference; SG33-6689-05
> Section 3.2 Configuring the LE/C TCP/IP Socket API Multiplexer
> http://publibz.boulder.ibm.com/cgi-bin/bookmgr/BOOKS/fl2cle05/3.2

The name of the TCP/IP interface phase supplied by IPv6/VSE is BSTTTCPV.PHASE (supports IPv4 only) or BSTTTCP6.PHASE (supports IPv4 and IPv6).

Using the IBM VTAPE Facility

The IBM VTAPE facility uses SVA Socket buffers. You must modify the $VTMAIN JCL as follows:

```
// OPTION SYSPARM='nn'
// LIBDEF *,SEARCH=(bsilib.slib,…)
// SETPARM SVABUF=YES
// SETPARM SENDALL=YES
// SETPARM IPTRACE='NNNNNNNN'
```

Warning: Double check the IPv4 address specified in your VTAPE JCL. If you specify an IP address that is invalid the $VTMAIN server may appear to 'hang' waiting for a connection. After the TCP/IP retransmit timeout message appears on the console you can simply flush the partition, correct the JCL and resubmit the job. We strongly recommend using DNS resolved names instead of hard coded IPv4 addresses. DNS resolved names are easier to remember and less prone to problems.

Using ASM SOCKET Applications Under CICS TS

The ASM SOCKET API is a commonly used socket programming API used by both users and vendors. CICS TS programs using the ASM SOCKET macro API must be defined to CICS with an EXECKey of CICS.

Additionally, see Step 3 and Step 4 of the Using the LE/C API ($EDCTCPV.PHASE) too.

Note: The ASM SOCKET API CICSKEY requirement was removed in Build 240 GA

BSTTPREP messages

BSI-W **CSI EXEC TELNET API REQUIRES REVIEW**
The TELNET interface is not fully supported so the generated code will need to be reviewed.

BSI-W **KEYWORD xxxx IS IGNORED**
The keyword xxxx has been ignored, as it is not supported within the EZASOKET interface specification.

BSI-E *KEYWORD LOCAL IS RESTRICTED**
The keyword “LOCAL” is not supported within the EZASOKET interface specification.

BSI-E *KEYWORD xxxx IS UNKNOWN**
The keyword xxxx is unknown to the BSTTPREP program. If the keyword is not a valid CSI keyword, correct the code. Otherwise, contact BSI support.

BSI-E *NO PERIOD AFTER END-EXEC**
The original “EXEC TCP” code did not have a period after the “END-EXEC”. As BSTTPREP inserts multiple “IF” statements, the original “END-EXEC” is treated as though a period was found. The generated code must be reviewed to determine if the logic flow has been modified.

BSI-E *OPTION WAIT(NO) IS RESTRICTED**
The “WAIT(NO)” is not supported within the EZASOKET interface specifications in a manner similar to the CSI TCP/IP API.

BSI-E *UNSUPORTED CSI TCP API IGNORED**
Only the “EXEC TCP” and “EXEC TELNET” CSI TCP/IP API calls are supported. The code will require manual conversion.

EZASOKET Error Codes

ERRNO_00013	The other application (listener) did not give the socket to your application.
ERRNO_00035	The socket is in nonblocking mode and read data is not available. This is not an error condition.
ERRNO_00056	The socket is already connected.
ERRNO_00057	The socket is not connected.
ERRNO_10101	A storage acquire failed.
ERRNO_10104	Invalid function was requested.
ERRNO_10108	The first call was not initapi, Gethostid, Gethostname, Getpeername, Getsockname, Getclientid, Takesocket, Getibmopt, Gethostbyname, or Gethostbyaddr.
ERRNO_10124	A zero or negative IOVCNT was specified for a READV, RECVMSG, SENDMSG, or SENDV call.
ERRNO_10125	The value for IOVCNT was greater than 120 for a READV, RECVMSG, SENDMSG, or SENDV call.
ERRNO_10142	Errors were found in the parameter list for an ACCEPT call.
ERRNO_10143	Errors were found in the parameter list for a BIND call.
ERRNO_10144	Errors were found in the parameter list for a CLOSE call.
ERRNO_10145	Errors were found in the parameter list for a CONNECT call.
ERRNO_10146	Errors were found in the parameter list for an FCNTL call.
ERRNO_10147	Errors were found in the parameter list for a GETCLIENTID call.
ERRNO_10148	Errors were found in the parameter list for a GETHOSTID call.
ERRNO_10149	Errors were found in the parameter list for a GETHOSTNAME call.
ERRNO_10150	Errors were found in the parameter list for a GETPEERNAME call.
ERRNO_10151	Errors were found in the parameter list for a GETSOCKNAME call.
ERRNO_10152	Errors were found in the parameter list for a GETSOCKOPT call.
ERRNO_10153	Errors were found in the parameter list for a GIVESOCKET call.
ERRNO_10154	Errors were found in the parameter list for an IOCTL call.
ERRNO_10158	The parameter list for a LISTEN call is incorrect.
ERRNO_10159	A zero or negative data lent was specified for a READ or READV call.
ERRNO_10162	The parameter list for a READ call is incorrect.
ERRNO_10163	A zero or negative data lent was specified for a RECV, RECVFROM, or RECMSG call.
ERRNO_10166	The parameter list for a RECV, RECVFROM, or RECVMSG call is incorrect.
ERRNO_10167	The discriptor set size for a SELECT or SELECTEX call is less than or equal to zero.
ERRNO_10169	The parameter list for a SELECT or SELECTEX call is incorrect.
ERRNO_10170	A zero or negitive datga length was found for a SEND or SENDMSG call.
ERRNO_10173	The parameter list for a SEND call is incorrect.
ERRNO_10174	A zero or negative data length was found for a SENDTO call.
ERRNO_10177	The parameter list for a SENDTO call is incorrect.
ERRNO_10178	The SETSOCKOPT option length is less than the minimum length.
ERRNO_10179	The SETSOCKOPT option length is greater than the maximum length.
ERRNO_10180	The parameter list for a SETSOCKOPT call is incorrect.
ERRNO_10181	The parameter list for a SHUTDOWN call is incorrect.
ERRNO_10182	The parameter list for a SOCKET call is incorrect.
ERRNO_10183	The parameter list for a TAKESOCKET call is incorrect.
ERRNO_10186	A negative data length was sspecified for a WRITE or WRITEV call.
ERRNO_10188	Errors were found in the parameter list for a WRITE call.

ERRNO_10190	The GETHOSTNAME option length is less than 24 or greater than the maximum length.
ERRNO_10193	The GETSOCKOPT option length is less than the minimum or greater than the maximum length.
ERRNO_10197	The application issued an INITAPI call after the connection was already established.
ERRNO_10198	The maximum number of sockets spcified for an INITAPI exceeds 2000.
ERRNO_10203	The requested socket number is a negitive value.
ERRNO_10205	The requested socket number is a duplicate.
ERRNO_10208	The NAMELEN parameter for a GETHOSTBYNAME call was not specified.
ERRNO_10209	The NAME parameter on a GETHOSTBYNAME call was not specified.
ERRNO_10210	The HOSTENT parameter on a GETHOSTBYNAME or GETHOSTBYADDR call was not specified.
ERRNO_10211	The HOSTADDR parameter on a GETHOSTBYNAME or GETHOSTBYADDR call is incorrect.
ERRNO_10215	The APITYPE parameter on an INITAPI call instruction was not 2.
ERRNO_10218	The application programming interface API) cannot locate the specified TCP/IP.
ERRNO_10219	The NS parameter is greater than the maximum socket for this connection.
ERRNO_10220	Trying to close socket that has not been allocated.
ERRNO_10221	The AF parameter of a SOCKET call was not AF_INET.
ERRNO_10222	The SOCTYPE parameter of a SOCKET call must be stream or datagram (1 or 2).
ERRNO_10223	No ASYNC parameter specified for INITAPI for APITYPE=3.
ERRNO_10226	Invalid COMMAND parameter specified for a GETIBMOPT call.
ERRNO_10228	The CANCEL call was issued on an non-asynchronus connection.
ERRNO_10229	A call was issued on an APITYPE=2 connection without an ECB parameter.
ERRNO_10330	A SELECT or SELECTEX call was issued without a MAXSOC value oand a TIMEOUT parameter.
ERRNO_10332	A SELECT or SELECTEX call wass invoked wit a MAXSOC value greater than that which was specified on the INITAPI.
ERRNO_10333	Error in asynchronous exit routine
ERRNO_20001	Invalid FUNCTION used with EZASOKET call.
ERRNO_20100	Error loading phase EZASOH00
ERRNO_20101	Error loading phase BSTTIPS3.
ERRNO_20106	Error loading phase BSTTIPS1.
ERRNO_30301	EZHP storage exhausted.
ERRNO_30302	Incorrect call to BSTTIPS3.
ERRNO_30303	Protocol does not match protocol specified on SOCKET call.
ERRNO_30304	No free sockets.
ERRNO_30305	The parameter list for a CANCEL call is incorrect.
ERRNO_30306	ACCEPT issued against an un-opened socket.
ERRNO_30307	An attempt was made to close a socket while it was waiting for a TAKESOCKET.
ERRNO_30308	Storage release failed.
ERRNO_30309	Call specified an ECB, but INITAPI did not specify ASYNC=ECB.
ERRNO_30310	Error loading phase BSTTIPAM.
ERRNO_30311	Requested socket has not been allocated.
ERRNO_30312	Synchronous close failed.
ERRNO_30313	Error loading phase BSTTSTMR
ERRNO_30314	BSTTSTMR processing error
ERRNO 30315	Invalid value for ASYNC
ERRNO 30316	EZTSK Out of Task Storage (Batch)
ERRNO 30317	EZTSK Out of Task Storage (CICS)

ERRNO 30318	Not Auth for EXEC CICS INQ TASK (Heap STG Recovery) (MAXSOC)
ERRNO_30350	IPAM return R15 + 30350 (max 30378)
-1	Socket not connected
-4	Send request timeout (re-transmit limit exceeded)
-5	Socket connection reset or BSTTINET terminated
-6	Request cancelled

Additional Error Codes

EAGAIN	112
EBADF	113
EBUSY	114
EFAULT	118
EINTR	120
EINVAL	121
EIO	122
ENOSYS	134
EPERM	139
EPIPE	140
EMVSINITIAL	156
EWOULDBLOCK	1102
EALREADY	1104
ENOTSOCK	1105
EOPNOTSUPP	1112
ENETDOWN	1117
ECONNRESET	1121
ENOBUFS	1122
ENOTCONN	1124
ECANCELLED	1152

Network Interface Device Support

The BSTTINET TCP/IP stack currently supports several network interface devices.

- 3172 LAN Channel Station (LCS) devices
- OSA and OSA/2 devices
- OSA Express (OSAX) devices (requires VSE/ESA 2.6 or higher)
- CLAW devices (RS/6000 and 3172 CLAW mode)
- CTCA and Virtual CTCA devices
- IUCV linkage to VM TCP/IP and LINUX/390

9221 CETI Ethernet and Token Ring adapters are currently not supported.

Partition Size

The BSTTINET TCP/IP stack requires a ***minimum 20MB partition***. This amount of storage will allow for stack startup and storage for a few users. Running the BSTTINET TCP/IP stack in a 20MB partition will allow for basic testing but 20MB is to small for production usage.

The basic formula is 20MB + 108K per TCP connection. For example, on a system with 100 TCP (TN3270E) connections plus 10 FTP (client and server) connections would require 20MB + 108 x 110 (11880K) = ~32MB.

The maximum number of TCP connections currently supported is 4095. To support the maximum number of connections the stack would need to run in a 460MB partition.

Partition Priority

Ideally a production VSE/ESA system should have two network interfaces. A high priority BSTTINET TCP/IP stack for interactive traffic would use the 1st network interface. The 2nd network interface would be used by lower priority BSTTINET TCP/IP stack for bulk data transfer (FTP) traffic.

The BSTTINET TCP/IP stack may be balanced or even run lower priority than the TCP/IP applications it is servicing. On VSE/ESA systems with a single network interface this allows for greater flexibility in priority management.

Remember that TCP/IP is a timer driven protocol. The BSTTINET TCP/IP stack can run at lower priorities but still must have access to the CPU when timer processes need to run.

Network Traffic Filtering

Any network interface used by VSE/ESA should be attached to a filtering switch.

- Filter all Ethernet packets except types x'0800' (IP), x'0806' (ARP) and x'8035' (RARP).
- Filter all IP Ethernet packets except IP protocol 1 (ICMP), 6 (TCP), 17 (UDP) and 89 (OSPF).
- Filter all IP UDP Ethernet packets by port number to limit broadcast traffic.

Port Number Selection

RFC 1340 Assigned Numbers states ...

"The Well Known Ports numbers are controlled and assigned by the IANA and on most systems can only be used by system (or root) processes or by programs executed by privileged users. The assigned ports use a small portion of the possible port numbers. For many years the assigned ports were in the range 0-255. Recently, the range for assigned ports managed by the IANA has been expanded to the range 0-1023."

Please keep this in mind when selecting port numbers to use with IPv6/VSE applications.

Waiting for the TCP/IP stack to come up

You can use the BSTTWAIT program to cause a job stream to wait until the BSTTINET TCP/IP stack is ready. The BSTTWAIT program will wait for the BSTTINET TCP/IP stack to come up or a message from the operator. Use the MSG xx (xx is the partition id) command to terminate the BSTTWAIT program early.

```
// LIBDEF PHASE,SEARCH=(ttlib.slib)
// OPTION SYSPARM='nn'
// EXEC BSTTWAIT,SIZE=BSTTWAIT
/*
```

ttlib.slib is the IPv6/VSE lib.slib

nn is the BSTTINET stack ID

Using the Services Table

The TCP/IP stack supports a services table. The services table is in the BSTTSERV.T member of the BSI lib.slib. The services table defines services by name and port. Applications can access the services table to determine the port number they should be using.

ASM SOCKET API application can use the GETPORTBYNAME Control request to query the services table. EZA API applications can use the GETHOSTINFO to query the services table. At this time, only the BSI TCP/IP stacks support a services table.

BSI applications support using the services table. When an OPEN command is processed, the IP address is resolved. If the port number is specified on the OPEN command, this is the port number used. If the port number is not specified, the services table will be queried to determine port number to be used. If the services table does not contain an entry for that application or the port number is zero, the default port number (hard-coded in the application) is used.

The advantage of using the BSTTSERV.T services table to specify a port number is simple. Changing the port number used for a service can be done without any changes to JCL or applications.

Sample BSTTSERV.T Services Member

```
CATALOG BSTTSERV.T                              REPLACE=YES
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                                         *
*   COMMON SERVICES USED                                                  *
*                                                                         *
*   THIS TABLE PROVIDES A SMALL LIST OF SERVICES AVAILABLE                *
*                                                                         *
*   THE TABLE SHOWS THE SERVICE NAME AND THE DEFAULT PORT                 *
*   NUMBER. THE PROTOCOL IS TREATED AS A COMMENT BUT MAY BE               *
*   USED IN THE FUTURE.                                                   *
*                                                                         *
*   ANY LINE WITH AN ASTERISH (*) IN COLUMN 1 IS A COMMENT.               *
*                                                                         *
*   APPLICATION CAN USE THE ASM SOCKET API OR THE EZA API                 *
*   TO QUERY THE DEFAULT PORT NUMBER FOR THE SERVICE THEY ARE             *
*   PROVIDING. FOR EXAMPLE, AN FTP CLIENT OR SERVER CAN                   *
*   QUERY THE FTP SERVICE FOR THE DEFAULT PORT NUMBER TO USE.             *
*                                                                         *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*
FTP              21 TCP
TELNET           23 TCP
SMTP             25 TCP
DOMAIN           53 UDP
HTTP             80 TCP
NTP             123 TCP
HTTPS           443 TCP
EXEC            512 TCP
SYSLOG          514 UDP
PRINTER         515 TCP
/+
```

Give and Take Socket Performance

The BSI TCP/IP stack supports both global and local Givesocket and Takesocket requests. The default operating mode is Global Givesocket and Takesocket. When global Givesocket and Takesocket mode is active, the stack handles all Give/Take requests. When Local operation mode is active, the API within the application partition, handles all Give/Take requests. Local Give/Take operation mode reduces CPU usage by about 12% (Your mileage will vary!).

The following SETPARM, in the application JCL, activates Local Give/Take operating mode ...

```
// SETPARM LOCALGT=YES
```

Large TCP Window Support

The BSI TCP/IP stack supports large TCP receive windows. The LRGBUF SETPARM provides an external method of enabling or disabling usage of this support by an application.

As a starting point you might find this helpful ...
http://bsitcpip.blogspot.com/2010/09/care-and-feeding-of-your-zvse-tcpip.html

To enable 64-bit storage usage on z/VSE 5.1+ please refer to the SYSDEF MEMOBJ command in the z/VSE System Control Statement manual.

E.g., `SYSDEF MEMOBJ,MEMLIMIT=MAX,SHRLIMIT=50M,LFAREA=20M`

Not all applications can benefit from using large TCP windows. The CICS TS CWS (HTTP daemon) or a TN3270E server are examples of interactive applications. Applications that benefit from large TCP windows are applications reading large amounts of data to/from a remote host.

What applications do this? BSTTFTPC (batch FTP) doing a RETR operation. The IBM VTAPE sever partition reading a remote virtual tape file, etc.

So, the TCP/IP stack will not allocate large TCP window buffers unless all the conditions are correct.

The conditions are ...

```
1) z/VSE 4.2+ with LTWBUF specified -or-
   z/VSE 5.1+ with 64-bit available.
2) SHIFT 4|5|6|7 specified in the stack startup.
   Default is SHIFT 4 for z/VSE 4.2/4.2 (1M) and
   SHIFT 6 (4M) for z/VSE 5.1+
3) The application must requeste large TCP windows buffers.
    I) // SETPARM LRGBUF=YES in the JCL -or-
   II) 'L' specified in the SRBLOK+1 during SOCKET OPEN,TCP request
4) The remote host TCP/IP stack must also agree to use large TCP
windows. The shift value used for a socket is negotiated during the
connection process. The SYN packets contain an options header for
this purpose and both the local and remote TCP/IP stacks must agree
to use 'shifted windows' (large TCP windows) before they can be used.
```

Currently the only application requesting large TCP window buffers is BSTTFTPC when an RETR operation is being done.

Note: No programming changes are required when // SETPARM LRGBUF=YES is specified in the applications JCL. However, when this option is specified in the JCL, all sockets created by the application will use large TCP window buffers.

The following SETPARM, in the application JCL, enables or disables large TCP receive window support for an application by the stack.

```
// SETPARM LRGBUF=YES|NO
```

Default Maximum Sockets

The default MAXSOC value is 64 sockets on an EZA or LE/C initapi() request. Some applications, including CICS Web Services, do not specify a MAXSOC value resulting in the default maximum sockets allocated. For applications with very high transaction rates the default value of 64 sockets may not be enough. The maximum value is 1024.

The following SETPARM, in the application JCL, overrides the default value ...

```
// SETPARM MAXSOC=nnnn
```

Posting All Partition Subtasks

Normally, the BSI TCP/IP stack will only post the z/VSE subtask that issued a request upon completion of the request. If you are writing an application that issues a socket request from one z/VSE subtask and waits on the request from another z/VSE subtask, you need to inform the TCP/IP stack of this. This SETPARM is not required if you are running z/VSE 4.2 (or higher).

The following SETPARM, in the application JCL, specifies partition level posting ...

```
// SETPARM POST=PARTITION
```

CICS TS Socket Domain Performance

Applications using the CICS TS Socket Domain can have poor performance. This is because CICS TS does not directly wait on the ECB used by the Socket Domain. Instead CICS TS checks these ECBs periodically based on the CICS TS ICV value. BSI provides a performance enhancement to assist application using the Socket Domain interface.

The program BSTTPLT0 runs in the CICS TS PLT StartUp and PLT Shutdown. This program issues a CICS START for the transaction that runs the BSTTPLT1 program. The BSTTPLT1 program will run until CICS termination. If the BSTTINET TCP/IP stack is recycled without CICS TS being recycled the BSTTPLT0 can be used to terminate and restart the BSTTPLT1 program. Place the BSTTPLT0 DFHPLT entry after the DFHDELIM on start up (PI) and before the DFHDELIM on shutdown (SD).

These transactions and program must be defined to CICS TS. Any group and transaction names can be used. The G(BSTT) and transactions BST0/BST1 are examples.

```
         DFHPLT TYPE=ENTRY,PROGRAM=BSTTPLT0

E G(BSTT)
ENTER COMMANDS
 NAME     TYPE         GROUP
 BSTTPLT0 PROGRAM      BSTT
 BSTTPLT1 PROGRAM      BSTT
 BST0     TRANSACTION  BSTT
 BST1     TRANSACTION  BSTT

 CEDA  View PROGram( BSTTPLT0 )
  PROGram        : BSTTPLT0
  Group          : BSTT
  DEscription    : PLT PROGRAM
  Language       : Assembler
  RELoad         : No
  RESident       : No
  USAge          : Normal
  USEsvacopy     : No
  Status         : Enabled
  RSl            : 00
  Cedf           : Yes
  DAtalocation   : Any
  EXECKey        : Cics

 CEDA  View PROGram( BSTTPLT1 )
  PROGram        : BSTTPLT1
  Group          : BSTT
  DEscription    : PLT PROGRAM
  Language       : Assembler
  RELoad         : No
  RESident       : No
  USAge          : Normal
  USEsvacopy     : No
  Status         : Enabled
  RSl            : 00
  Cedf           : Yes
  DAtalocation   : Any
  EXECKey        : Cics
```

```
CEDA  View TRANSaction( BST0 )
 TRANSaction    : BST0
 Group          : BSTT
 DEscription    : START BSTT PLT PROGRAM
 PROGram        : BSTTPLT0
 TWasize        : 00000             0-32767
 PROFile        : DFHCICST
 PArtitionset   :
 STAtus         : Enabled           Enabled | Disabled
 PRIMedsize     : 00000             0-65520
 TASKDATALoc    : Any               Below | Any
 TASKDATAKey    : Cics              User | Cics

CEDA  View TRANSaction( BST1 )
 TRANSaction    : BST1
 Group          : BSTT
 DEscription    : LONG RUNNING ECB WATCHER
 PROGram        : BSTTPLT1
 TWasize        : 00000             0-32767
 PROFile        : DFHCICST
 PArtitionset   :
 STAtus         : Enabled           Enabled | Disabled
 PRIMedsize     : 00000             0-65520
 TASKDATALoc    : Any               Below | Any
 TASKDATAKey    : Cics              User | Cics
```

BSTTINET commands

ATTACH

```
ATTACH TCP/IP
```

The ATTACH command is used to start the TCP/IP stack subtask. This command must be present in the BSTTINET startup command and must also be the last command.

BACKLOG

```
BACKLOG nn
```

The BACKLOG command defines the TCP/IP stack's default connection backlog queue depth. Values from 5 to 20 are valid. The default is 10.

CAPTURE

```
CAPTURE ON|OFF
```

The CAPTURE command enables (ON) or disables (OFF) the stack's external packet capture facility. The BSTTCAP VSAM ESDS capture file is used by this command. When the CATPTURE ON command is issued, the capture facility automatically opens the BSTTCAP file when the next packet is processed by the stack. When the CAPTURE OFF command is issued, the BSTTCAP file is automatically closed after the next packet is processed by the stack.

CLOSETOV

```
CLOSETOV nnn
```

The CLOSETOV command specifies the CLOSE Time Out Value. ***nnn*** is specified in minutes and the default is 3 minutes.

DEVICE

```
DEVICE device_name CTCA cuu
DEVICE device_name IUCV 0 0 vm_id
DEVICE device_name LCS cuu ETHERNET|TOKENRING
DEVICE device_name OSA cuu ETHERNET|TOKENRING
DEVICE device_name OSAX cuu portname cuu3 LAYER2 macaddr
DEVICE VIRTUALx    OSAX cuu dev_name cuu3 LAYER2 macaddr
DEVICE device_name HIPR cuu portname cuu3
DEVICE device_name CLAW cuu claw_host_name claw_name
```

The DEVICE command is used to define a network adapter to the BSTTINET TCP/IP stack. This command must precede any LINK commands that reference the specified *device_name*. Up to 31 network devices can be specified. For LCS or OSA devices ETHERNET is assumed unless TOKENRING specified. OASX indicates an OSA Express adapter. HIPR indicates a Hipersocket device. OSAX or HIPR cuu must specify the lowest cuu address of the required 3 OSAX/HIPR device addresses. If the control cuu is not in sequence with the OSAX/HIPR read/write ports cuu3 can be used to specify this cuu address. When OSA Express 3 adapters are used, the port number being used is specified in the adapter_number field of the LINK statement.

```
OSAX devices are only supported under VSE/ESA 2.6 + DY46036.
OSAX usage with z/VM Guest Lan requires DY46036.
HIPR devices are only supported under VSE/ESA 2.7. DY46197 is
recommended.
OSAX/HIPR devices require SETPFIX LIMIT=(256K,1100K) per DEVICE
command.
Cuu is the unit address of the device.
```

Vm_id is the virtual machine name of the VM TCP/IP or LINUX/390 machine.

portname is the OSAX or HIPR device portname.DEVICE

OSAX devices requiring layer 2 support instead of the default layer 3 network interface must specify the literal LAYER2. When exploiting Layer 2 support with a z/VSE system that is not connected to a z/VM VSwitch, the user must manually configure a unique MAC address (which must be different from the default MAC address of the port) for the z/VSE network interface. The MAC address is specified as 12 hexadecimal characters. The first byte of the actual MAC address used will be forced to x'02' indicating a user defined MAC address. Layer 2 support is available only on z/VSE 5.1 (or higher) systems.

The DEVICE called VIRTUALx (x=0-9,A-Z) is a virtual OSAX (OSA Express) device. The portname field is used to reference the real OSA Express device. Virtual device specifications must precede the DEVICE statement for the real OSA Express device. This allows you to have multiple IP/VMAC addresses for each physical OSA Express adapter. Each device (real or virtual) can be on a different VLAN too.

Virtual devices allow you to define an additional virtual NIC for a physical device. The virtual device will have a different IP address. The additional IP address can be on the same or a different subnet. This allows you to define multiple IP addresses for a single physical device. This also implies your network

folks are going to setup the necessary network environment for this to work.

Claw_host_name and claw_name are the names specified during the CLAW device configuration. Claw_host_name is equivalent to the CSI/IBM HOSTNAME parameter. Claw_name is equivalent to the CSI/IBM WSNAME parameter.

Sample VSE ADD statements

```
ADD 600:601,CTCA,EML    Sample CTC/OSA/LCS/CLAW device ADD
ADD 600:602,OSAX        Sample OSA Express ADD
ADD 600:602,OSAX,01     Sample Hipersocket ADD
```

CTC Connections

CTC connections under VM normally use cross-coupled virtual CTC definitions. The CTC read address of system A is coupled with the CTC write address of SYSTEM B and the CTC write address of SYSTEM A is coupled with the CTC read address of SYSTEM B. The read address in the even numbered address. The write address is the odd address. When using real CTC connections between systems, one system must specify a LINK command-using adapter 0 while the other system must specify a LINK command-using adapter 1.

DNS

```
DNS a.b.c.d PRI|SEC
```

The DNS command specifies a dotted decimal format IP address of a DNS server. If the PRI option is specified the primary DNS IP address is set or updated. If the SEC option is specified the secondary DNS IP address is set or updated. If the DNS command is issued without an option (PRI/SEC) the first DNS command sets the primary DNS IP address and the second DNS command sets the secondary DNS IP address.

DOMAIN

```
DOMAIN .name
```

The DOMAIN command specifies a domain name suffix to be used by the DNS look up routines. The name must begin with a leading period. The length is limited to 55 characters. Up to 4 DOMAIN commands may be specified. If the name is defined with a HOST command this value is used. Look ups are done on the qualified names before the unqualified name look up is done.

DOS

```
DOS ADD IP-address subnet-mask
DOS DEL IP-address subnet-mask
DOS PRINT
```

The DOS command is used to have the BSTTINET TCP/IP stack deny service to an IP address or range of IP addresses. **Warning:** Be sure you specify the correct subnet mask or you can deny service to a broad range of IP addresses. The DOS PRINT command will print the DOS table.

FILTER

```
FILTER ON|OFF
```

The FILTER command is used to enable or disable packet filtering. The default is OFF. When FILTER ON is specified echo packet and packet sent to the wrong network interface are discarded before processing.

GARBAGE

```
GARBAGE ON|OFF
```

The GARBAGE command adjusts the TCP/IP packet type used in TCP Keep Alive processing. GARBAGE OFF (default) sets NULL packet mode. GARBAGE ON set garbage packet mode (single byte of out-of-sequence data).

HOST

```
HOST name ip_address
```

The HOST command specifies a host file entry. The host file associates a name with a dotted decimal format IP address. When performing DNS name-to-IP or IP-to-name lookups the host file is searched first. Multiple HOST commands can be specified in the BSTTINET startup commands. At least one HOST command is required for each network adapter used by the BSTTINET TCP/IP stack. This HOST statement identifies the local IP address and its associated name. When the IP address specified in the HOST command specifies 0.0.0.0 the HOST entry will be deleted. If the name specified in the HOST command already exists the ip address of the name is replaced. If the host name contains a plus sign (+), the plus sign is translated to a period (.). This allows you to enter fully qualified names in a HOST command. A HOST command that contains a colon (:) (specifying an IPv6 address) will be ignored. See the installation library member HOSTS.T for instructions on setting up a common HOST command member for use with multiple TCP/IP stacks. Host names defined with the HOST command are limited to 56 characters.

HOTSWAP

```
If your primary DEVICE command is ...
DEVICE OSAX610 OSAX 610 portname 612

Use this HOTSWAP command ...
HOTSWAP OSAX610 0710 0712 0
```

The HOTSWAP command specifies a spare OSA Express adapter and port that can be used by the BSTTINET stack in the event the primary OSA Express device fails. The HOTSWAP command specifies the device name and cuu addresses of the read/write pair and the control cuu address. The port number to use is also specified. The cuu address must be specified as 4 character hexadecimal values.

ID

```
ID nn
```

The ID command specifies the TCP/IP stack identifier. The TCP/IP stack identified is a 2 digit number in the range 00 to 99. This command must be specified for proper stack startup processing.

INTERVAL

```
INTERVAL nnn [EXCEPT lport lport lport …]
```

The INTERVAL specifies the number of seconds the TCP/IP stack (BSTTINET) waits after receiving a packet for a connection before it sends a keep-alive packet for that connection. This command must be specified in the startup commands. Not specifying this command in the BSTTINET startup results in INTERVAL 0 which disables Keep-Alive processing. If specified in the BSTTINET startup the minimum value is 60 and should be a multiple of 60 seconds. The EXCEPT clause identifies up to 8 local ports (lport) to be excluded from keep-alive processing. The EXCEPT clause is optional.

IP ARPTAB

```
IP ARPTAB
```

The IP ARPTAB command displays the contents of the ARP table.

Sample Output

```
BSTT010I IP ARPTAB
BSTT613I Intf.   IP address     Physical Address   HW  Proto   State
BSTT613I ----- ---------------  -----------------  --  -----  ------
BSTT613I   1    192.168.1.1     00:a0:cc:22:db:7c   1   0800  RESOLV
BSTT011I COMMAND PROCESSING COMPLETE
```

IP BPOOL

```
IP BPOOL
```

The IP BPOOL command displays the buffer pool statistics.

Sample Output

```
BSTT010I IP BPOOL
BSTT613I pool= 0 count  100 bsize=  32, sem=8187, inuse=   7 (  7%)
BSTT613I pool= 1 count 1024 bsize=1524, sem=8183, inuse=   5 (  0%)
BSTT613I pool= 2 count   32 bsize=65567, sem=8182, inuse=   0 (  0%)
BSTT011I COMMAND PROCESSING COMPLETE
```

IP CLOSE

```
IP CLOSE nnn
```

The IP CLOSE command closes all connections on the specified local port.

IP IP2NAME

```
IP IP2NAME a.b.c.d
```

The IP IP2NAME command translates a dotted decimal format IP address to a name. Reverse DNS lookups are supported only for host names defined using a HOST command to the stack.

Sample Output

```
BSTT010I IP IP2NAME 192.168.1.1
BSTT612I DNS RESPONSE JCB
BSTT011I COMMAND PROCESSING COMPLETE
```

IP NAME2IP

```
IP NAME2IP name
```

The IP NAME2IP command translates a name into a dotted decimal format IP address.

Sample Output

```
BSTT010I IP NAME2IP JCB
BSTT612I DNS RESPONSE 192.168.1.1
BSTT011I COMMAND PROCESSING COMPLETE
```

IP NETINFO

```
IP NETINFO
```

The IP NETINFO command displays information about the network devices in use. The primary and secondary DNS server IP addresses are also shown.

Sample Output

```
BSTT010I IP NETINFO
BSTT608I DEVICE 01 NAME LCS612   MAC 00A0C9FC8AF3 MTU   1500
BSTT609I IP 192.168.1.13     NA 192.168.1.0      SA 192.168.1.0
BSTT610I MS 255.255.255.0    BR 192.168.1.255    NB 192.168.1.255
BSTT611I DNS SERVER 192.168.1.2
BSTT611I DNS SERVER 192.168.1.2
BSTT011I COMMAND PROCESSING COMPLETE
```

IP NETSTATS

```
IP NETSTATS
IP LOGSTATS
```

The IP NETSTATS display network statistics on Console/SYSLST.

The IP LOGSTATS command displays network statistics on SYSLST.

Sample Output

```
BSTT010I IP NETSTATS
BSTT011I COMMAND PROCESSING COMPLETE
```

IP PING

```
IP PING ip_address
IP PING name
```

The IP PING command pings the specified IP address. The ip_address can be specified as a dotted decimal IP address or as a name. If a name is specified the NAME2IP routine is used to convert the name to an IP address.

Sample Output

```
BSTT010I IP PING 192.168.1.1
BSTT093I REPLY FROM 192.168.1.1      TIME     5
BSTT093I REPLY FROM 192.168.1.1      TIME     6
BSTT093I REPLY FROM 192.168.1.1      TIME     6
BSTT011I COMMAND PROCESSING COMPLETE
```

IP RAWINFO

```
IP RAWINFO
```

The IP RAWINFO command is used to display information about RAW Socket connections.

Sample Output

```
msg s1,d=ip rawinfo
AR 0015 1I40I  READY
S1 0048 BSTT010I IP RAWINFO
S1 0102 BSTT613I Raw        Remote IP Proto Queue ID
S1 0102 BSTT613I --- ---------------- ----- ----- --
S1 0102 BSTT613I   0    192.168.1.104     1    -1 P2
S1 0102 BSTT011I COMMAND PROCESSING COMPLETE
```

IP ROUTE

```
IP ROUTE ADD device_name network_address network_mask gateway metric
IP ROUTE DEL device_name network_address network_mask gateway
IP ROUTE PRINT
```

The IP ROUTE command will add, delete or print the current routing table. The metric value will default to zero and may be omitted unless the route defines a non-local network.

Sample Output

```
BSTT010I IP ROUTE PRINT                                          \
BSTT607I NETWORK         MASK            GATEWAY         IFN MTR
BSTT606I 0.0.0.0         0.0.0.0         192.168.1.100    01 00
BSTT606I 127.0.0.1       255.255.255.255 127.0.0.1        00 00
BSTT606I 192.168.1.12    255.255.255.255 192.168.1.4      01 01
BSTT606I 192.168.1.255   255.255.255.255 192.168.1.13     00 00
BSTT606I 192.168.1.0     255.255.255.255 192.168.1.13     00 00
BSTT606I 192.168.1.13    255.255.255.255 192.168.1.13     00 00
BSTT606I 192.168.1.0     255.255.255.0   192.168.1.100    01 00
BSTT011I COMMAND PROCESSING COMPLETE
```

IP TRACE

```
IP TRACE ON
IP TRACE OFF
```

The IP TRACE enables or disables tracing of the stack. Trace data is printed on SYSLST.

Sample Output

```
BSTT010I IP TRACE ON
BSTT011I COMMAND PROCESSING COMPLETE
```

IP TRACERT

```
IP TRACERT ip_address
IP TRACERT name
```

The IP TRACERT command traces the network path taken to reach the specified destination.

Sample Output

```
M1 0045 BSTT010I IP TRACERT CFL.RR.COM
M1 0100 BSTT618I TRACING ROUTE TO 24.95.227.33
M1 0100 BSTT619I     1    24    18    19 10.165.248.1
M1 0100 BSTT619I     2    21    17    19 24.95.225.105
M1 0100 BSTT619I     3    19    53    19 24.95.225.113
M1 0100 BSTT619I     4    23    18    17 24.95.227.33
M1 0100 BSTT011I COMMAND PROCESSING COMPLETE
```

IP UDPINFO

```
IP UDPINFO
```

The IP UDPINFO command displays information about UDP sockets.

Sample Output

```
msg s1,d=ip udpinfo
AR 0015 1I40I  READY
S1 0048 BSTT010I IP UDPINFO
S1 0102 BSTT613I  DG LPort        Remote IP RPort Flg Queue ID
S1 0102 BSTT613I --- ----- --------------- ----- --- ----- --
S1 0102 BSTT613I   0   123          0.0.0.0     0 012    -1 Z3
S1 0102 BSTT613I   1   161          0.0.0.0     0 011    -1 S1
S1 0102 BSTT011I COMMAND PROCESSING COMPLETE
```

LINK

```
LINK name adapter_no ip_addr net_mask mtu VLAN number prty GLOBAL
```

The LINK command identifies a link from the network to the network device specified. This command must follow the DEVICE command for the device_name specified.

Adapter_number is normally 0 but may be up to 3 for devices with multiple adapters. When OSA Express 3 adapters are used, the port number being used is specified in the adapter_number field.

Ip_address is the dotted decimal format ip address of this network interface. E.g., 192.168.1.1.

Network_mask is the network mask of the network. E.g., 255.255.255.0.

Mtu_size specifies the MTU size to be used. Generally this value is in the range or 576 to 1500. CTCA and OSAX devices support an MTU size of up to 9000. HIPR devices support an MTU of up to 57216. Specifying a value larger than the maximum supported for the device will result in the maximum being used.

CTC connections under VM normally use cross-coupled virtual CTC definitions. The CTC read address of system A is coupled with the CTC write address of SYSTEM B and the CTC write address of SYSTEM A is coupled with the CTC read address of SYSTEM B. The read address in the even numbered address. The write address is the odd address. When using real CTC connections between systems, one system must specify a LINK command-using adapter 0 while the other system must specify a LINK command-using adapter 1.

If the network interface is to be connected to a VLAN trunc port the literal VLAN must be specified along with the VLAN number and priority. If a VLAN is not being used or the adapter is connected to an access port these parameters may be omitted

The GLOBAL parameter applies to OSA Express and Hipersocket interfaces using Layer 3 only. Layer 2 is not supported. Usage of this parameter requires z/VSE 5.1 and IJBOSA at APAR level DY47264/UD53715/UD53716.

LTWBUF

```
LTWBUF nnn
```

The LTWBUF command is used to specify the amount of 31-bit partition GETVIS to reserve during TCP/IP stack startup for large TCP window buffers. This command is valid under z/VSE 4.2 and higher only. The amount of storage that needs to be specified is related to the SHIFT command. The SHIFT 4/5/6/7 command indicates 1/2/4/8 MB buffers respectfully. To allocate 20 buffers, reserve 20 * buffer_size. Under z/VSE 5.1 (or higher) 64-bit virtual will be used. Usage of 64-bit virtual also requires 64-bit virtiual storage be made available with the SYSDEF SYSTEM command at IPL time.

PORT

```
PORT nnn SUSPEND
PORT nnn RESUME
PORT
```

The PORT command is used to SUSPEND or RESUME connections on a specified port. To disable connections on a port use the SUSPEND command. To re-enable connections use the RESUME command. To query suspended ports enter the PORT command with no options.

PROXYARP

```
PROXYARP ip_address

Example:
PROXYARP 192.168.1.26
```

The PROXYARP command specifies an IP address. When another host ARPs this IP address the BSTTINET stack will reply with its hardware mac address. PROXYARP allows a VSE system to act as a geteway and answer ARP requests for hidden hosts. Hidden host will be connected to the VSE gateway system via a CTC or IUCV connection. The hidden hosts can have IP addresses on the same subnet as the VSE gateway system.

ROUTE

```
ROUTE device_name network_address network_mask gateway metric

Examples:
ROUTE CTC600 192.168.1.0 255.255.255.0 0.0.0.0 0
ROUTE CTC600 0.0.0.0 0.0.0.0 192.168.1.100 1
```

The ROUTE command specifies a static route using the specified device_name to the network. Normally two route statements are required. The first specifies the route used by the local subnet and the second specifies the default route. The default route identifies the gateway to be used to deliver a packet to an outside network. The gateway IP address must be specified even if it is null (0.0.0.0).

The route metric is used to tell the stack the number of hops required to reach the specified outside network. A metric of zero (0) indicates that the network is local subnet and the IP addresses on this network can be ARP'ed to determine their MAC addresses. A metric of one (1) or more indicates that the route is to an outside network and the IP addresses on this network can only be accessed via the specified gateway IP address. No ARP'ing is done for these network IP addresses. The default metric is zero (0).

In the example above, the first ROUTE command specifies a local subnet route. The IP addresses on subnet 192.168.1.0 are ARP'able. This is indicated by the metric of zero (0). The second ROUTE command specified a default route with a gateway IP address of 192.168.1.100. The metric of one (1) indicates the IP addresses using this route are not ARP'able.

REDIRECT

```
REDIRECT ON|OFF
```

The REDIRECT command enables (ON) or disables (OFF) ICMP Redirect processing. The default is ON.

SEGMENT

```
SEGMENT * $$ LST …
```

The SEGMENT command is used to tell the BSTTINET command processor to issue a VSE/POWER SEGMENT request.

SHIFT

```
SHIFT 0|4|5|6|7
```

The SHIFT command is used to enable large TCP receive window support. The command SHIFT 0 is used to disable the support. This command must be part of the TCP/IP stack startup commands and once enabled the SHIFT value cannot be changed except to disable the support. The SHIFT 4|5|6|7 command specified 1|2|4|8 MB buffer usage respectfully. The default is 0 for z/VSE 4.1 (and before), 4 for z/VSE 4.2 (and higher) and 6 for z/VSE 5.1 (and higher).

TRACEIP

```
TRACEIP iii.iii.iii.iii mmm.mmm.mmm.mmm
```

The TRACEIP command is used to limit the output from an TRACE2 ON packet trace. The iii.iii.iii.iii value is the IP address or subnet address. The mmm.mmm.mmm.mmm is the subnet mask. If an IP address is specified use a subnet mask of 255.255.255.255. The subnet mask must be specified.

TRACEID

```
TRACEID xx
```

The TRACEID command is used to limit output from a TRACE ON API trace. Trace output is limited to a specific partition.

TRACEEZ

```
TRACEEZ xx yyyyyyyy
```

The TRACEEZ command is used to enable and disable EZASOKET API traces in a partition. The xx value is the partition ID. The yyyyyyy value indicates Y/N values for each possible trace.

EZASOKETs Trace flag settings

```
// SETPARM IPTRACE='xxxxxxx'
'Y.......'  produce call parms trace information
'.Y......'  produce iprpl trace information
'..Y.....'  produce console messages on entry and exit
'...Y....'  produce trace information on SYSLST, not direct to LST
queue
'....Y...'  produce waitlist trace information
'.....Y..'  produce one line entry and exit messages trace
information
'......Y.'  Force 'Y.......' for any call which results in an error
```

USERMSS

```
USERMSS 1460
```

The USERMSS command is used to override the RFC 1122 Host Requirements minimum value for the Maximum Segment Size (MSS) used. The default is 536. Values over 1460 should not be used without contacting BSI support.

Simple LCS and CTC Network Configuration

The following example shows a simple 2 VSE/ESA system. The VSEPROD system is connected to the network and is acting as a gateway for the VSETEST system using a CTCA connection.

```
    VSEPROD (System A)          VSETEST (System B)
                          ^
                          |
    LCS 192.168.1.3 <- --+
    CTC 192.168.1.5 <- ----- -> 192.168.1.4

CP DEFINE 600 CTC               CP DEFINE 600 CTC
CP DEFINE 601 CTC               CP DEFINE 601 CTC
CP COUPLE 600 VSETEST 601       CP COUPLE 600 VSEPROD 601
CP COUPLE 601 VSETEST 600       CP COUPLE 601 VSEPROD 600

ADD 440:441,CTCA,EML
ADD 600:601,CTCA,EML            ADD 600:601,CTCA,EML

*  VSEPROD System A
// SETPFIX LIMIT=256K
// EXEC BSTTINET,SIZE=BSTTINET,OS390
ID 00
INTERVAL 120
*
DEVICE CTC600 CTCA 600
LINK   CTC600 0 192.168.1.5 255.255.255.0 1480
ROUTE  CTC600   192.168.1.4 255.255.255.255 192.168.1.5 1
*
DEVICE LCS440 LCS 440 ETHERNET
LINK   LCS440 1 192.168.1.3 255.255.255.0 1480
ROUTE  LCS440   192.168.1.0 255.255.255.0 0.0.0.0         0
*
ROUTE  LCS440   0.0.0.0       0.0.0.0       192.168.1.101 1
*
PROXYARP 192.168.1.4
HOST VSEPROD 192.168.1.3
HOST VSETEST 192.168.1.4
*
ATTACH TCP/IP
/*

*  VSETEST System B
// SETPFIX LIMIT=128K
// EXEC BSTTINET,SIZE=BSTTINET,OS390
ID 00
INTERVAL 120
*
DEVICE CTC600 CTCA 600
LINK   CTC600 0 192.168.1.4 255.255.255.0 1480
ROUTE  CTC600   192.168.1.0 255.255.255.0 0.0.0.0 0
*
ROUTE  CTC600   0.0.0.0       0.0.0.0       192.168.1.3 1
*
HOST VSEPROD 192.168.1.3
HOST VSETEST 192.168.1.4
*
ATTACH TCP/IP
/*
```

BSTTINET TCP/IP Stack JCL

In general the BSTTINET JCL follows the following format.

```
// OPTION LOG,PARTDUMP,NOSYSDUMP,SADUMP=1
// LIBDEF PHASE,SEARCH=(toollib.slib)
// LIBDEF SOURCE,SEARCH=(cfglib.slib,toollib.slib)
// SETPFIX LIMIT=128K
// EXEC BSTTINET,SIZE=BSTTINET,OS390
ID nn
INTERVAL nnn
DEVICE devname …
LINK   devname …
ROUTE  devname …
ROUTE  devname …
HOST … (1 required for each LINK statement)
ATTACH TCP/IP
/*
```

toollib.slib is the IPv6/VSE installation lib.slib

cfglib.slib is the lib.slib where the BSTTPARM.A member is stored. Normally PRD2.CONFIG.

The // EXEC statement includes the '*,OS390'* parameter. This is valid for VSE/ESA 2.4 or newer systems. This parameter can be omitted for VSE/ESA 1.4, 2.1, 2.2 and 2.3 systems.

The // SETPFIX LIMIT should be set to 128K for each DEVICE command. OSAX and HIPR DEVICE commands require LIMIT=(256K,1100) per DEVICE command.

For more information about the ROUTE command see the ROUTE command documentation.

The INTERVAL should be specified. This command enables TCP/IP Keep Alive processing and socket cleanup within BSTTINET. The recommended INTERVAL value is 120.

Sample 3172 LCS JCL

In VSE ADD 612:613,CTCA,EML

```
// OPTION LOG,PARTDUMP,NOSYSDUMP,SADUMP=1
// LIBDEF PHASE,SEARCH=(BSILIB.TTDEV)
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,BSILIB.TTDEV)
// SETPFIX LIMIT=128K
// EXEC BSTTINET,SIZE=BSTTINET,OS390
*
*  SPECIFY THE STACK ID
*
ID 10
INTERVAL 120
*
* DEVICE DEVICE_NAME TYPE CUU NETTYPE
* LINK   DEVICE_NAME ADAPTER_NUMBER IP_ADDR MTU_SIZE
* ROUTE  DEVICE_NAME NETWORK_ADDR NET_MASK IP_GATEWAY METRIC
*
DEVICE LCS612 LCS 612 ETHERNET
LINK   LCS612 0 192.168.1.13 255.255.255.0 1500
ROUTE  LCS612 192.168.1.0 255.255.255.0 0.0.0.0 0
*
* THE FOLLOWING ROUTE IS THE DEFAULT ROUTE.
* IF ALL ELSE FAILS SEND THE PACKET TO THIS DEVICE
* AND TO THE SPECIFIED GATEWAY IP ADDRESS.
*
ROUTE  LCS612 0.0.0.0 0.0.0.0 192.168.1.100  1
*
* DEFINE THE DSN SERVERS IP ADDRESSES
* THE 1ST DNS SPECIFIES PRIMARY, 2ND SECONDARY
*
DNS 192.168.1.2
DNS 192.168.1.3
*
* HOSTS FILE DEFINITIONS
*
HOST JCB      192.168.1.1
HOST SERVER   192.168.1.2
HOST ROUTER   192.168.1.100
*
*  STARTUP THE TCP/IP STACK ...
*
ATTACH TCP/IP
/*
```

Sample OSA and OSA/2 JCL

In VSE ADD 612:613,CTCA,EML

```
// OPTION LOG,PARTDUMP,NOSYSDUMP,SADUMP=1
// LIBDEF PHASE,SEARCH=(BSILIB.TTDEV)
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,BSILIB.TTDEV)
// SETPFIX LIMIT=128K
// EXEC BSTTINET,SIZE=BSTTINET,OS390
*
*  SPECIFY THE STACK ID
*
ID 10
INTERVAL 120
*
* DEVICE DEVICE_NAME TYPE CUU NETTYPE
* LINK   DEVICE_NAME ADAPTER_NUMBER IP_ADDR MTU_SIZE
* ROUTE  DEVICE_NAME NETWORK_ADDR NET_MASK IP_GATEWAY METRIC
*
DEVICE OSA612 OSA 612 TOKENRING
LINK   OSA612 0 192.168.1.13 255.255.255.0 1500
ROUTE  OSA612 192.168.1.0 255.255.255.0 0.0.0.0
*
* THE FOLLOWING ROUTE IS THE DEFAULT ROUTE.
* IF ALL ELSE FAILS SEND THE PACKET TO THIS DEVICE
* AND TO THE SPECIFIED GATEWAY IP ADDRESS.
*
ROUTE  OSA612 0.0.0.0 0.0.0.0 192.168.1.100 1
*
* DEFINE THE DSN SERVERS IP ADDRESSES
* THE 1ST DNS SPECIFIES PRIMARY, 2ND SECONDARY
*
DNS 192.168.1.2
DNS 192.168.1.3
*
* HOSTS FILE DEFINITIONS
*
HOST JCB      192.168.1.1
HOST SERVER   192.168.1.2
HOST ROUTER   192.168.1.100
*
*  STARTUP THE TCP/IP STACK ...
*
ATTACH TCP/IP
/*
```

Sample IUCV JCL

```
// OPTION LOG,PARTDUMP,NOSYSDUMP,SADUMP=1
// LIBDEF PHASE,SEARCH=(BSILIB.TTDEV)
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,BSILIB.TTDEV)
// SETPFIX LIMIT=128K
// EXEC BSTTINET,SIZE=BSTTINET,OS390
*
*  SPECIFY THE STACK ID
*
ID 13
INTERVAL 120
*
* DEVICE DEVICE_NAME TYPE 0 0 VM_ID
* LINK   DEVICE_NAME ADAPTER_NUMBER IP_ADDR MTU_SIZE
* ROUTE  DEVICE_NAME NETWORK_ADDR NET_MASK IP_GATEWAY METRIC
*
DEVICE IUCVLINK IUCV 0 0 TCPIP
LINK   IUCVLINK 0 192.168.1.15 255.255.255.0 1500
ROUTE  IUCVLINK 192.168.1.0 255.255.255.0 192.168.1.9
ROUTE  IUCVLINK 0.0.0.0 0.0.0.0 192.168.1.9 1
*
* DEFINE THE DSN SERVERS IP ADDRESSES
* THE 1ST DNS SPECIFIES PRIMARY, 2ND SECONDARY
*
DNS 192.168.1.2
DNS 192.168.1.3
*
* HOSTS FILE DEFINITIONS
*
HOST JCB      192.168.1.1
HOST SERVER   192.168.1.2
HOST ROUTER   192.168.1.100
*
*  STARTUP THE TCP/IP STACK ...
*
ATTACH TCP/IP
/*
```

Sample CLAW JCL

In VSE ADD 620:621,CTCA,EML

```
// OPTION LOG,PARTDUMP,NOSYSDUMP,SADUMP=1
// LIBDEF PHASE,SEARCH=(BSILIB.TTDEV)
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,BSILIB.TTDEV)
// SETPFIX LIMIT=128K
// EXEC BSTTINET,SIZE=BSTTINET,OS390
*
*  SPECIFY THE STACK ID
*
ID 12
INTERVAL 120
*
* DEVICE DEVICE_NAME TYPE CUU CLAW_HOST CLAW_NAME
* LINK   DEVICE_NAME ADAPTER_NUMBER IP_ADDR MTU_SIZE
* ROUTE  DEVICE_NAME NETWORK_ADDR NET_MASK IP_GATEWAY METRIC
*
DEVICE CTC620 CLAW 620 VSE 3172CLAW
LINK   CTC620 0 192.168.1.14 255.255.255.0 1500
ROUTE  CTC620 192.168.1.14 255.255.255.0 192.168.1.9 1
*
* THE FOLLOWING ROUTE IS THE DEFAULT ROUTE.
* IF ALL ELSE FAILS SEND THE PACKET TO THIS DEVICE
* AND TO THE SPECIFIED GATEWAY IP ADDRESS.
*
ROUTE  CTC620 0.0.0.0 0.0.0.0 192.168.1.9 1
*
* DEFINE THE DSN SERVERS IP ADDRESSES
* THE 1ST DNS SPECIFIES PRIMARY, 2ND SECONDARY
*
DNS 192.168.1.2
DNS 192.168.1.3
*
* HOSTS FILE DEFINITIONS
*
HOST JCB      192.168.1.1
HOST SERVER   192.168.1.2
HOST ROUTER   192.168.1.100
*
*  STARTUP THE TCP/IP STACK ...
*
ATTACH TCP/IP
/*
```

Sample CTCA JCL

In VSE ADD 620:621,CTCA,EML

```
// OPTION LOG,PARTDUMP,NOSYSDUMP,SADUMP=1
// LIBDEF PHASE,SEARCH=(BSILIB.TTDEV)
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,BSILIB.TTDEV)
// SETPFIX LIMIT=128K
// EXEC BSTTINET,SIZE=BSTTINET,OS390
*
*  SPECIFY THE STACK ID
*
ID 11
INTERVAL 120
*
* DEVICE DEVICE_NAME TYPE CUU
* LINK   DEVICE_NAME ADAPTER_NUMBER IP_ADDR MTU_SIZE
* ROUTE  DEVICE_NAME NETWORK_ADDR NET_MASK IP_GATEWAY METRIC
*
DEVICE CTC620 CTCA 620
LINK   CTC620 0 192.168.1.14 255.255.255.0 1500
ROUTE  CTC620 192.168.1.0 255.255.255.0 192.168.1.9
*
* THE FOLLOWING ROUTE IS THE DEFAULT ROUTE.
* IF ALL ELSE FAILS SEND THE PACKET TO THIS DEVICE
* AND TO THE SPECIFIED GATEWAY IP ADDRESS.
*
*ROUTE  CTC620 0.0.0.0 0.0.0.0 192.168.1.9 1
*
* DEFINE THE DSN SERVERS IP ADDRESSES
* THE 1ST DNS SPECIFIES PRIMARY, 2ND SECONDARY
*
DNS 192.168.1.2
DNS 192.168.1.3
*
* HOSTS FILE DEFINITIONS
*
HOST JCB      192.168.1.1
HOST SERVER   192.168.1.2
HOST ROUTER   192.168.1.100
*
*  STARTUP THE TCP/IP STACK ...
*
ATTACH TCP/IP
/*
```

Sample OSAX JCL

In VSE ADD 710:712,OSAX

```
// OPTION LOG,PARTDUMP,NOSYSDUMP,SADUMP=1
// ASSGN SYS000,SYSLST
// LIBDEF PHASE,SEARCH=(BSILIB.TTDEV)
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,BSILIB.TTDEV)
// SETPFIX LIMIT=(256K,1100K)
// EXEC BSTTINET,SIZE=BSTTINET,OS390
ID 30
INTERVAL 120
*
DEVICE OSAX710 OSAX 710 VMVSE27
LINK   OSAX710 0 192.168.8.2  255.255.255.0 1500
ROUTE  OSAX710   192.168.8.0  255.255.255.0 0.0.0.0       0
*
ROUTE  OSAX710   0.0.0.0      0.0.0.0       192.168.8.1   1
*
ATTACH TCP/IP
/*
```

Sample Hipersocket JCL

In VSE ADD 700:702,OSAX,01

```
// OPTION LOG,PARTDUMP,NOSYSDUMP,SADUMP=1
// ASSGN SYS000,SYSLST
// LIBDEF PHASE,SEARCH=(BSILIB.TTDEV)
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,BSILIB.TTDEV)
// SETPFIX LIMIT=(256K,1100K)
// EXEC BSTTINET,SIZE=BSTTINET,OS390
ID 30
INTERVAL 120
*
DEVICE HIPR700 HIPR 700 VMVSE26
LINK   HIPR700 0 192.168.9.2  255.255.255.0 1500
ROUTE  HIPR700   192.168.9.0  255.255.255.0 0.0.0.0       0
*
ROUTE  HIPR700   0.0.0.0      0.0.0.0       192.168.9.1   1
*
ATTACH TCP/IP
/*
```

Sample BSTTINET Startup Output

```
BSTT000I INITIATED  BSTTINET Ver 1.81 05/07/00 11.31    EP=00420078
BSTT003I COPYRIGHT (C) 1998-2000 BARNARD SOFTWARE, INC.
BSTT002I TCP/IP-TOOLS BUILD 1.81
BSTT004I CB=TTLA A=0042C000 L=00000B38
BSTT019I VSE 6.30 MODE 31-BIT
BSTT004I CB=COMR A=002BC520 L=00000108
BSTT000I INITIATED  BSTTXIPC Ver 1.81 05/07/00 03.38    EP=006B2000
BSTT045I TCP/IP ID SET TO 10
BSTT000I INITIATED  BSTTISRV Ver 1.81 05/07/00 03.19    EP=006CE680
BSTT000I INITIATED  BSTT21EP Ver 1.81 05/04/00 09.10    EP=80286258
BSTT600I     0 INITIATED  ipboot   Ver 1.81 04/23/00 12.22
BSTT600I     1 INITIATED  COMMAND  Ver 1.81 04/16/00 21.53
BSTT600I     2 INITIATED  netstart Ver 1.81 04/19/00 16.25
BSTT600I     3 INITIATED  slowtmr  Ver 1.79 03/07/00 21.44
BSTT601I     0 TERMINATED ipboot
BSTT600I     4 INITIATED  ipproc   Ver 1.81 04/14/00 18.07
BSTT600I     5 INITIATED  tcptmr   Ver 1.80 04/05/00 15.37
BSTT600I     6 INITIATED  tcpinp   Ver 1.81 05/05/00 21.09
BSTT600I     7 INITIATED  tcpout   Ver 1.81 05/03/00 15.38
BSTT600I     8 INITIATED  XPCTASK  Ver 1.81 05/05/00 13.22
BSTT600I     9 INITIATED  XPCTASK  Ver 1.81 05/05/00 13.22
BSTT600I    10 INITIATED  EOTTASK  Ver 1.81 05/04/00 08.46
BSTT004I CB=IPBH A=00892000 L=00011000
BSTT600I    11 INITIATED  DEVLCS   Ver 1.81 05/07/00 11.26
BSTT605I    11 DEVICE 01 LCS612   TYPE LCS      UNIT 0612
BSTT600I    12 INITIATED  snmpd    Ver 1.81 04/16/00 14.24
BSTT601I     2 TERMINATED netstart
```