

# **XNET (Cross System Net) Configuration and Usage Guide**

V1.2.0

07 October 2018

## **XNET Summary**

XNET connects together two or more S/370-based VM systems for the purposes of allowing remote logon. XNET allows 3270 terminal users to log on to userids on other similar VM system(s) that are currently online to XNET. Likewise, a user on one of the other VM system(s) would be able to log on to a userid on your system. You still retain access control for your own systems via the system directory and passwords.

XNET provides the management and transport of 3270 data between systems in order to facilitate remote logons and connections to other VM systems. XNET can support multiple users and multiple systems, and supports two types of data transport mechanisms. XNET is a fully asynchronous mini-operating system that runs in a virtual machine.

## **Getting Started**

Below is a general list of considerations and steps to perform to set up XNET for use. Each consideration or step is detailed further on within this documentation.

1. Decide on the transport mechanism: Channel-to-Channel Adapter (CTCA), or Binary Synchronous (BSC) lines.
2. Ensure that your Hercules emulator is at the right level.
3. Ensure that the VM host systems being connected together meet the minimum requirements.
4. Define real CTCA or BSC line devices to CP in DMKRIO.
5. To Hercules, define a CTCE device matching the CTCA device address, or if using BSC lines define a pair of 2703 line devices matching the line addresses you defined to CP. In either case, you will need a local port number, destination IP of the remote Hercules VM host, and remote port number.
6. Define a virtual machine to host XNET.
7. Load the tape image file, xnetv120.aws to the XNET 191 disk.
8. Adjust the supplied XNET CONFIG to match the network configuration you have in mind.
9. Start XNET on two or more nodes.

## **Detailed steps and additional information**

### **Deciding the transport mechanism: Whether to use Channel-to-Channel Adapter, or binary synchronous lines.**

XNET requires an emulated channel-to-channel adapter (CTCA) or a pair of emulated binary synchronous (BSC) lines to communicate with each remote VM system. The Hercules emulator can provide emulated CTCA devices via the CTCE device definition

in the Hercules configuration file. The emulator can also provide emulated BSC lines via 2703 device definitions in the Hercules configuration file.

Because of the vagaries of BSC communications protocols, the CTCE device is generally a faster transport medium. However, depending on what version of the Hercules emulator you are using you may not have the CTCE device capability or have the right level of CTCE support. If you are comfortable with compiling and building your own Hercules emulator, then using CTCE devices are recommended. If you just want to get XNET up and running with whatever Hercules emulator you have, use BSC lines.

**Ensure that your Hercules emulator is at the right level.**

The minimum recommended release of the Hercules emulator for the purpose of running XNET is Hercules 3.12. That will operate with BSC lines. CTCE support is available in Hercules 3.13 but it does not have necessary fixes to the CTCE device driver. For CTCE use, the minimum requirement is that the device driver code (ctcaadpt.c) in Hyperion 4.0 be at the 01 April 2018 level or later, if you are comfortable compiling and building your own emulator. Using the Software Development Labs (SDL) version of Hyperion (Fish's version) is recommended. SDL Hyperion is ready to go with either CTCE or BSC lines.

The pre-built version of SDL Hyperion as well as links to the source code repository can be found at <http://www.softdevlabs.com/hyperion.html>

**Ensure that the VM host systems being connected together meet the minimum requirements.**

The minimum requirement for VM host systems that can properly operate XNET is VM/370 Release 6, with the addition of the DIAG58 V3.1.0 enhancement installed.

Many users have the VM/370 Sixpack 1.2 or the VM 5-Pack distributions. These will work fine if the DIAG58 V3.1.0 enhancement is installed.

The VM/370 Sixpack 1.3 Beta 3 distribution is also available. This version of Sixpack already has the DIAG58 V3.1.0 enhancement installed as well as other enhancements that can be particularly useful with XNET, notably the "SYSID" system node name identifier string displayed next to the "RUNNING" status on 3270 displays.

The VM/370 Sixpack Beta 1.3 distribution can be found at <http://www.smrcc.org.uk/members/g4ugm/SixPack-1.3.Beta.htm>

**Define real CTCA or BSC line devices to CP in DMKRIO.**

Edit your VM system's DMKRIO file to add one or more CTCA devices, or to add one or more pairs of BSC lines, or both. Add one CTCA device per remote host that you wish to directly connect with, or one BSC line pair per remote host connection.

**For a CTCA connection:** CTCA device addresses must end in zero (e.g., 500, 610, 720, etc). Only one CTCA device per channel is recommended. Below provides an example

definition for two CTCA devices, one at device address 500 and one at 600. Note that in VM/370 you must specify the minimum range to be 8; e.g. (500,8).

```
RDEVICE ADDRESS=(500,8),DEVTYPE=CTCA
RDEVICE ADDRESS=(600,8),DEVTYPE=CTCA
*
RCTLUNIT ADDRESS=500,CUTYPE=3088
RCTLUNIT ADDRESS=600,CUTYPE=3088
*
RCHANNEL ADDRESS=5,CHTYPE=BLKMPXR
RCHANNEL ADDRESS=6,CHTYPE=BLKMPXR
```

Note: VM/SP users may have to specify FEATURE=32-DEVICE on the RCTLUNIT macro.

Reassemble DMKRIO and rebuild your nucleus and re-ipl.

**For a BSC line connection:** Add BSC line devices to DMKRIO on the system(s) that will host XNET. There are no restrictions on device addresses but channel 0 is recommended. Note that a XNET connection via BSC lines requires two lines, which function as a pair in full duplex mode. This is an example definition for 16 lines (8 pairs):

```
RDEVICE ADDRESS=(060,16),DEVTYPE=2703,ADAPTER=BSCA
*
RCTLUNIT ADDRESS=060,CUTYPE=2703,FEATURE=16-DEVICE
*
RCHANNEL ADDRESS=0,CHTYPE=MULTIPLEXOR
```

Reassemble DMKRIO and rebuild your nucleus and re-ipl.

### **Define the device(s) to the Hercules emulator.**

Add the Hercules configuration file device descriptions as appropriate for whether you are using CTCA or BSC lines for XNET.

**For a CTCA connection:** define the CTCE device(s) in Hercules, either in the configuration file or at the command line via the Hercules Attach command. The format is:

```
cuu CTCE lport ipnumber rport
```

where *cuu* is the device address of the CTCA device you added to DMKRIO; *lport* is the designated port number on the local XNET side of the connection; *ipnumber* is the ip address or ip number of the remote Hercules running VM and XNET (the remote side of the connection); and *rport* is the designated port number on the remote side of the connection.

Note that the remote port number specified on one side of the connection is the local port number on the other side, and vice versa. It is recommended that you keep the port numbers on both sides identical to reduce confusion and instead only use different port numbers per CTCE defined.

For example:

```
# SYSTEM-A definitions (SYSTEM-A is 192.168.1.2)
# connection to SYSTEM-B:
0500 CTCE 30880 192.168.1.17 30880
# connection to SYSTEM-C:
0600 CTCE 30890 192.168.1.6 30890
```

The SYSTEM-B host on the other side of the connection would have this definition:

```
# connection to SYSTEM-A:
0500 CTCE 30880 192.168.1.2 30880
```

The SYSTEM-C host on the other side of the connection would have this definition:

```
# connection to SYSTEM-A:
0600 CTCE 30890 192.168.1.2 30890
```

Note: there is no requirement that device addresses or port numbers be the same on both sides, but making them the same can help to reduce confusion.

***For a BSC line connection:*** define the line pair in Hercules, either in the configuration file or at the command line via the Hercules Attach command. The format is:

```
cuu 2703 dial=no lport=num      rport=num      rhost=ipaddr
cuu 2703 dial=no lport=num+1    rport=num+1    rhost=ipaddr
```

where *cuu* is the device addresses of at least two line devices you added to DMKRIO; *lport* is the designated port number on the local XNET side of the connection; *rport* is the designated port number on the remote XNET side of the connection; and *rhost* is the ip address or ip number of the remote Hercules running VM and XNET (the remote side of the connection).

For example:

```
0066 2703 dial=no lport=31080 rport=31080 rhost=192.168.1.17
0067 2703 dial=no lport=31081 rport=31081 rhost=192.168.1.17
```

The host on the other side of the connection would have this definition:

```
0066 2703 dial=no lport=31080 rport=31080 rhost=192.168.1.2
0067 2703 dial=no lport=31081 rport=31081 rhost=192.168.1.2
```

Note: there is no requirement that device addresses or port numbers be the same on both sides, but making them the same can help to reduce confusion. BSC line pairs need not have a consecutively numbered *cuu* designation but keeping them together can help reduce confusion.

***For either CTCE or BSC line connections:*** the *real* addresses of the devices defined in DMKRIO and Hercules can be whatever you choose within the normal rules for defining devices. When the devices are attached to XNET, the *virtual* addresses must conform to a few restrictions. For BSC lines, they must be attached on virtual channel 0. For CTCA devices, they must be attached on virtual channels 1 through F (one CTCA device per virtual channel).

Recommendation: to reduce confusion and where ever possible make the device addresses virtual=real, that is, the virtual address of the device as known to XNET is the same as the real address in DMKRIO. Therefore using the examples above, CTCA 500 would be attached to XNET as 500, or BSC lines 066 and 067 would be attached to XNET as 066 and 067.

**Define a virtual machine to host XNET.**

This machine needs no special command privileges; however being able to issue ATTACH or DETACH commands can be handy for CTCA devices and BSC lines. The machine requires ECMODE.

Use the supplied XNET DIRECTORY file as a sample template for the XNET user directory entry.

The sample directory entry is also shown below. Note that the entry specifies IPL 190 instead of the usual IPL CMS; this is required. IPL 190 allows CMS to initialize without the CMS segment in low storage, which would otherwise interfere with XNET. A large storage size is highly recommended. The DEDICATE statements are used to assign either BSC lines (at 066 and 067, commented out in the example) or a CTCA devices (at 500). Add as many more line pairs or CTCA devices as are required for your host connections. You also need to define some GRAF devices to allow users to dial into this machine (they must be addressed between 000-0FF); eight such devices are defined at 0C8-0CF and more can be added if needed. Finally, the MDISK statement defines a small minidisk to hold the XNET executable, its configuration files, and a PROFILE EXEC.

```

USER XNET    XNET    16M 16M  G
OPTION ECMODE
IPL 190
CONSOLE 009 3215
SPOOL 00C 2540 READER *
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 A
LINK MAINT 190 190 RR
LINK MAINT 19D 19D RR
LINK MAINT 19E 19E RR
*DEDICATE 066 066
*DEDICATE 067 067
DEDICATE 500 500
SPECIAL 0C8 3270
SPECIAL 0C9 3270
SPECIAL 0CA 3270
SPECIAL 0CB 3270
SPECIAL 0CC 3270
SPECIAL 0CD 3270
SPECIAL 0CE 3270
SPECIAL 0CF 3270
* userid    cuu    type    adr num volser    mode readpw    writepw    multpw
MDISK      191    3350    078 002 VMPK81    MR    ALL      WRITE      MULT

```

Sample XNET user directory entry

Save the directory and issue the DIRECT command. Then logon to the XNET virtual machine and format the 191 disk.

### **Load the XNET tape image file to the XNET 191 disk.**

This will load the XNET executable program and example configurations to the A-disk.

Steps:

1. Attach a tape drive to the XNET machine as virtual address 181:  
CP ATTACH 480 XNET 181
2. Mount the 'tape' to Hercules:  
devinit 480 *c:/pathname/xnetv120.aws*
3. Load the tape materials to the XNET 191 disk:  
VMFPLC2 LOAD \* \* A
4. Detach the tape drive and tape file:  
CP DETACH 181

### **Set up your XNET CONFIG file.**

The XNET configuration file defines to XNET all of the host systems (nodes) and the devices that will be used to connect them together. It also defines the routing to other nodes that may pass through one or more directly connected nodes. A node is a VM system that is hosting XNET.

The configuration file must have a file type of 'CONFIG'. The filename can be of your choosing. The filename is specified when launching XNET, or if not specified the default filename of XNET is used.

The minimum XNET configuration in its simplest form is two hosts. For the purposes of example, let them be named HOST1 and HOST2. Let them be assigned network id (NETID) numbers of 1 and 2, respectively. The two hosts are connected via a CTCA at address 500 on each host.

Here is the configuration for HOST1:

```
*      nodename  netid
*      -----  ----
LOCAL  HOST1      1
*
*      nodename  netid  netid      via
*      -----  ----  ----
*      NODE  HOST2      2      2
*
*      to
*      cuu  netid
*      ---  ----
CTCA  500      2
*
*      PFKey to exit from session
*      ----
EXITPFK  PF24
*
```

Example configuration for HOST1

Here is the configuration for HOST2:

```
*      nodename  netid
*      -----  ----
LOCAL  HOST2      2
*
*      nodename  netid  netid      via
*      -----  ----  ----
*      NODE  HOST1      1      1
*
*      to
*      cuu  netid
*      ---  ----
CTCA  500      1
*
*      PFKey to exit from session
*      ----
EXITPFK  PF24
*
```

Example configuration for HOST2

In all examples, it is important to understand the concept of "*adjacent node*". This is the node that data must pass through in order to get to other more distant nodes. Since in this example above the CTCA only connects between HOST1-HOST2, there are no other nodes that are more distant. However, HOST1 and HOST2 are still adjacent nodes to each other.

However, assume a third host was added beyond HOST2, e.g., HOST1-HOST2-HOST3. An additional NODE statement would be required in HOST1's configuration to define HOST3 via HOST2. In order for data to get from HOST1 to HOST3, it must pass through (or 'via') HOST2. HOST2 is adjacent to HOST1, but HOST3 is not adjacent to HOST1. It is adjacent to HOST2 however. Adjacent nodes are always the nodes directly connected via a CTCA or LINE pair.

```
*      nodename  netid
*      -----  -----
LOCAL  HOST1      1
*
*      nodename  netid  via
*      -----  -----  -----
NODE   HOST2      2      2
NODE   HOST3      3      2
*
*      to
*      cuu      netid
*      ---      -----
CTCA   500      2
*
*      PFKey to exit from session
*      ----
EXITPFK  PF24
```

Example of HOST1's configuration showing the addition of HOST3 beyond HOST2. Data moving between Host1 and Host3 must travel 'via' Host2.

Likewise in HOST2, being in the middle of the connections between HOST1 and HOST3. Being in the middle, HOST1 and HOST3 are adjacent to HOST2.

```
*      nodename  netid
*      -----  -----
LOCAL  HOST2      2
*
*      nodename  netid  via
*      -----  -----  -----
NODE   HOST1      1      1
NODE   HOST3      3      3
*
*      to
*      cuu      netid
*      ---      -----
CTCA   500      1
CTCA   600      3
*
*      PFKey to exit from session
*      ----
EXITPFK  PF24
```

Example of HOST2's configuration showing the addition of HOST3 adjacent to HOST2.



And finally, the full configuration for newly added HOST3. Only HOST2 is adjacent to HOST3, and data going back to HOST1 must pass through (or via) HOST2.

```
*      nodename  netid
*      -----  -----
LOCAL  HOST3      3
*
*              via
*      nodename  netid  netid
*      -----  -----  -----
NODE   HOST1      1      2
NODE   HOST2      2      2
*
*              to
*      cuu      netid
*      ---      -----
CTCA   600      2
*
*              PFKey to exit from session
*              -----
EXITPFK  PF24
```

Example configuration file for HOST3

Finally, within the XNET configuration file, the EXITPFK allows you to assign the PFkey you can use to exit from a session and back to the XNET hosts selection menu. Press that PFkey again to drop the dialed connection, if desired.

Other notes about the configuration file:

LINEs are defined similarly to CTCA. Using the examples above for the connections HOST1-HOST2-HOST3, below shows the HOST2 configuration from above, but using LINEs between HOST1-HOST2 instead of a CTCA, and continues to use CTCA between HOST2-HOST3. LINEs are defined using the two *cuu* values defining the pair of lines.

```
*      nodename  netid
*      -----  -----
LOCAL  HOST2      2
*
*              via
*      nodename  netid  netid
*      -----  -----  -----
NODE   HOST1      1      1
NODE   HOST3      3      3
*
*              to
*      cuu/cuu  netid
*      --- ---  -----
LINE   066 067    1
*
*              to
*      cuu      netid
*      ---      -----
CTCA   600      3
*
*              PFKey to exit from session
*              -----
EXITPFK  PF24
```

Example for Host2 showing LINE and CTCA definitions to HOST1 and HOST3, respectively.

You can mix or match LINES or CTCA definitions as needed. You could have LINES going to one node and CTCA to another, for example.

Both CTCA and LINE definitions have an optional START/NOSTART parameter in the sample configuration files, which are not shown in the examples above to reduce clutter. This allows the devices to be defined but not automatically started when XNET is started. It can be varied active later with an XNET console command (discussed later). If this parameter is not coded, START is the default.

## Launching XNET

All you need on the XNET 191 disk for each host are:

PROFILE EXEC (for CMS start up, as needed)  
XNET MODULE  
xxxx CONFIG (your configuration file for that node)

Once everything is ready in the Hercules, CP, and XNET configurations, you can start XNET from a CMS command line (after an IPL 190) by typing in:

*XNET config-name*

Where *config-name* is the filename of your CONFIG file you wish to use. If you do not specify *config-name* the default filename is XNET, and the assumed configuration file will be XNET CONFIG \*.

During start up, the configuration filename selected is confirmed. Then local node (the node of the host that XNET is running on) should activate within a few seconds. Then an attempt will be made to connect with a remote host. Of course, XNET should also be started on the remote host(s) as well.

You will know when the connection is successful when you receive messages that look similar to these (shown from HOST2's start up):

```
xnet host2
XNET010 Cross System Net - V1.2.0   10/08/18 08.51
XNET002 XNET using configuration file HOST2      CONFIG    *
XNET020 HOST2      active
XNET034 LINE 066 initialization in progress...
XNET016 CTCA 500 initialization in progress...
XNET017 CTCA 500 is Y-side
XNET019 CTCA 500 connected to node HOST3      (NETID=   3)
XNET036 LINE 066 is X, LINE 067 is Y
XNET037 LINE 066 connected to node HOST1      (NETID=   1)
XNET020 HOST1      active
XNET020 HOST3      active
```

The initialization process to connect and negotiate with remote hosts can take several seconds. 15-30 seconds is not unusual for BSC lines.

After XNET remote hosts are active, use a 3270 emulator to DIAL to the XNET machine. You will be presented with the XNET hosts menu. This is from the point of view of a HOST2 user:

```

XNET - Cross System Net                                HOST2 - 0C9
A - HOST1
B - HOST2
C - HOST3
> _
```

Example XNET host selection menu display.

The terminal *cuu* and host that you are dialed to is at the upper right. Select by letter the host you wish to connect with, then press Enter. You should then see the remote host's VM logo displayed upon your terminal where you could then proceed to log on. Use the EXITPFK to drop the dialed connection. If you are in a session with a remote host, use the EXITPFK to return back to this menu after you log off of the remote userid. Select a different host or press the EXITPFK again to drop the dialed connection.

## Troubleshooting

**CTCA problems:** Start up of CTCA devices via XNET is very reliable. But sometimes CTCA devices can occasionally be somewhat difficult to get started up. Sometimes the state of the CTCA device is unknown from the last time it was used and XNET can't get it started; it may be necessary to re-IPL 190 and launch XNET on both sides again, or use the XNET 'V cuu INACT/ACT' command to restart the CTCA. In most cases it is quite reliable and usually starts on the first attempt. XNET commands are documented below.

On the Hercules console, occasional CTCE messages may be issued. This is normal. What is not normal is if CTCE messages are repeating and scrolling up the Hercules console in a rapid fashion without stopping. In this case, either restart XNET or attempt to drop the connection and restart it with the 'V cuu INACT/ACT' console command.

At present, the CTCE device driver software in Hyperion does not close the TCP/IP socket when a CTCE is detached from Hercules or if the Hercules emulator on the remote side of the CTCE connection goes away (for example if a remote host is shut down each night). This means that the socket remains active (the port number is in use). If you wish to reestablish a CTCE connection with the same or different remote Hercules, you should

detach the CTCE device from your Hercules and reattach a new one with different port numbers. For example (note: these are Hercules commands):

```
detach 500
attach 500 CTCE 33080 192.168.1.17 33080
```

The remote side will also need to use the corresponding new port numbers.

Restarting Hercules (by exiting the emulator and restarting it) also closes all ports and the original numbers can be reused if the remote Hercules also restarts.

**LINE problems:** LINE devices can occasionally need to be restarted if they do not connect. LINES can also take 15-30 seconds for synchronization before they are started. If XNET can't get it started; it may be necessary to re-IPL 190 and launch both sides again, or use the V cuu INACT/ACT command to restart the LINES. Use only the first line pair *cuu* in the V command.

LINE problems can also cause line i/o error (IOE) messages to be issued to the XNET console. An odd IOE message or two is fine. If they are repeating and won't stop, either restart XNET or use the V cuu INACT and then ACT command to restart them.

A more common problem with LINES is getting them to 'sync-up' during initialization. This occurs more frequently if the LINES were used previously and left in an unknown state. This typically shows itself during XNET initialization via messages that reject a connection with an invalid NETID number and suggesting an error in the configuration file. Usually, the bogus NETID id number is a larger number that isn't part of the configuration file. Also, initialization 'sync-up' problems can be evidenced if the XNET messages issued on both sides of the connection say that *cuu* 066 (for example) is X, and *cuu* 066 on the other side also says it is X. When these types of problems occur, usually there is no resolution no matter how many times you restart XNET. The solution is to detach both line *cuu*'s from Hercules, and reattach them using completely new port numbers; this must occur on both Hercules sides.

**XNET problems:** XNET should never abend. If it ever does, it will display a disabled wait PSW of X'FF'. This is by design for now, to stop everything dead in its tracks so the problem can be diagnosed.

Some other less likely disabled wait PSWs are also possible:

```
00FF - program check
0103 - virtual console must be on channel 0
0104 - attempt to destroy task failed because task was not found
DDDD - internal error dispatching task
9777 - storage area overlaid
9888 - size of storage area to be released is invalid
9889 - internal error in storage area management
9998 - storage address to be released was not found (not allocated)
9999 - storage request cannot be filled (out of storage)
```

## **Limitations and other Considerations**

XNET has some internal limitations for device addresses and the number of connections.

- GRAF devices defined to XNET must have virtual addresses in the range 000-0FF.
- LINE devices defined to XNET must have virtual addresses in the range 000-0FF.
- CTCA devices defined to XNET must have virtual addresses ending in zero, and must not be on virtual channel 0. Thus, the valid address range for CTCA is 100-FF0. The theoretical limit for the number of defined CTCA devices is 240. However, more than one CTCA per channel is not recommended; busy conditions on a CTCA will impede all other devices on the same channel. Therefore the practical limit is 15 CTCA devices: one each on channels 1 through F.
- Up to 255 nodes may be defined with NODE statements, numbered from 1 to 255. Node names and node numbers (netid numbers) must be unique across XNET's known network. Node numbers need not be numbered consecutively. Node names can contain from 1 to 8 characters.

The virtual reader, printer, and punch devices are not used by XNET at this time. They can be removed from the directory entry if additional device addresses are needed for LINES or GRAFs.

XNET at present has no 'alternate path' capabilities. For example, in a multiple host arrangement, if a path to a remote host were to become inoperative, XNET will not automatically use an alternate pathway to that remote host even if one exists via a different remote host (for example, if three hosts were connected in a triangle).

If two or more pathways are defined to the same remote host, for example two CTCA devices, or a CTCA and a LINE pair, the last device to complete initialization and negotiation will be the pathway that is used for traffic between the hosts. If that pathway should become inoperative, XNET will not switch to the alternate pathway. There is no benefit at present to defining more than one pathway to a remote host.

Virtual CTCA devices are supported and can be used to connect two or more XNET virtual machines running on the same VM system, or alternately, to connect with a 2nd level VM system (e.g., running VM under VM) to tie first and second level XNET machines together.

## XNET Virtual Console Command Summary

### D TASKS

Displays the currently active tasks and their status, along with any associated connections where applicable.

```
d tasks
XNET025 Task Status 18:30:49
TaskID  Type Name      Status  Association
025118  CONS 009      ACTIVE
0252A8  NODE      ACTIVE
025478  LINE 066/067 ACTIVE  HOST1      XNET120
025880  CTCA 600      ACTIVE  HOST3      XNET120
027220  LDEV L000     ACTIVE  HOST1      040
027460  LDEV L001     ACTIVE  HOST3      07C
027688  GRAF 0C9      ACTIVE  HOST1      L000
027A30  GRAF 0CA      CONNECT
```

The left side display of the task type and name represent devices and tasks active on the host where the command was issued (in this example, the command was issued from HOST2). The LINEs and CTCA tasks represent those that are defined and what host they are connected with and what the version level of XNET is on the remote side. The LDEV tasks represent logical 3270 sessions initiated by remote users; the remote HOST and terminal cuu are displayed under the association heading. The GRAF tasks are those terminals belonging to HOST2 that are currently DIALED to XNET, and which may have an associated logical 3270 connection on a remote host.

#### Status Types:

NOTOPER	- device is not presently operational.
INACT	- device or node is inactive.
ACTIVE	- device or node is active and connected.
INIT	- device is initializing/attempting remote contact.
CONNECT	- 3270 device active but presently displaying host selection menu.
DACTC	- CTCA device in process of deactivating

## D NODES

Displays the current status of all nodes defined in the configuration.

```
d nodes
XNET032 Node Status 18:36:30
Netid   Name      Status
  1     HOST1     ACTIVE
  2     HOST2     ACTIVE    (LOCAL NODE)
  3     HOST3     ACTIVE
  4     HOST4     INACTIVE
 10     VM6BETA   INACTIVE
```

Status Types:

INACTIVE      - node is inactive.  
ACTIVE        - node is active and available for users.

## M nodeid text

Allows sending of instant messages to the XNET virtual machine console on another active node. Its purpose is to assist with communication to a remote XNET operator when trying to activate links or inquire about problems with the connections. Assume, for example that the XNET operator on HOST1 issued the following M command:

```
m host2 is your connection to HOST3 working?
```

Then on the HOST2 XNET console, the following appears:

```
@HOST1 is your connection to HOST3 working?
```

## V cuu ACT | INACT

Allows the XNET operator to vary active or inactive a particular LINE or CTCA device. The primary purpose is to allow the manipulation of these devices to avoid having to restart all XNET machines across the network simply to get a LINE/CTCA to reinitialize and connect. For LINES, use the *cuu* of the 1st line of the pair defined in XNET CONFIG.

When a LINE/CTCA is deactivated, all active terminal connections across that link are terminated.

During deactivation, an attempt is made to deactivate in an orderly fashion and send the request to the remote side of the affected LINE/CTCA so that it can deactivate as well. However, if the remote side is unable to receive the request or is otherwise non-responsive, the local side will still wait for a response to its request. After a few moments if the response is not received and the LINE/CTCA has not deactivated, reissue the *V cuu INACT* command a second time to force the LINE or CTCA inactive. It can then be reactivated when the remote host is again available.

In this example, the connection between HOST2 and HOST3 is to be deactivated; the command is issued from HOST2's XNET console. Note the dropped user session LDEV L001 from HOST3 terminal 07C.

```
v 500 inact
XNET024 CTCA 500 disconnected
XNET031 HOST3      inactive
XNET015 LDEV L001 disassociated with HOST3      07C
```

Later, when HOST3 is again available, the connection can be reactivated (and a corresponding 'v 500 act' command needs to be issued at HOST3 as well, (or alternately, XNET could have been restarted on HOST3). This is the reactivation attempt from HOST2:

```
v 500 act
XNET016 CTCA 500 initialization in progress...
XNET017 CTCA 500 is Y-side
XNET019 CTCA 500 connected to node HOST3      (NETID= 3)
XNET020 HOST3      active
```

When users are dropped due to a connection failure, those userids are placed into a force disconnected state by CP. If the connection can be reestablished, those users can logon to the userids again and resume where they left off. In some cases, CP may fully force off userids that are left in a force-disconnected state after 15 minutes.

On the other side of the dropped connection, 3270 terminals that were dialed to XNET are dropped back to their XNET host selection menu when the connection terminates. The user can wait for the connection to resume, or select a different host to connect with, or press the EXITPFK to drop the dialed connection with XNET.



## **About XNET**

XNET was written entirely in S/370 assembler and as of this writing consists of 6891 lines of code comprised of 19 CSECTs. XNET runs in S/370 extended control mode (EC mode) and has its own interruption handlers and i/o routines in order to execute code and process i/o concurrently.

XNET works by using the Logical Device Facility (LDF) to create "logical 3270" devices on the remote system and then the data streams going to and from that logical 3270 are transmitted over the CTCA or LINE pathway to the real 3270 connected to the local XNET by a 3270 emulator.

XNET contains an internal "router" function that serves to direct XNET data traffic to the right place. Data coming off the LINE or CTCA device needs to be routed to the specific 3270 (logical or real) task that is waiting for it, or perhaps be routed to a different LINE or CTCA device in order to continue on its journey to a different host. Eventually, the data stream will be received by a 3270 task at the final destination host and it will be presented to the logical or real 3270 device for display.

## **XNET Release Summary**

XNET 1.0.0 04 APR 2018

- Not publicly released
- First successful test across the internet

XNET 1.1.0 09 JUN 2018

- Not publicly released
- Support for emulated BSC line pairs to a remote host in lieu of CTCA.
- New console support and console commands
- New console command D TASKS
- New console command V cuu ACT/INACT
- Bug fixes

XNET 1.2.0 08 OCT 2018

- New node controller task which keeps track of which nodes are active or inactive as their status changes when LINE/CTCA devices are started or stopped.
- Rewrite CTCA initialization logic for improved reliability and start-up ease
- New console command: M *nodename* message text
- New console command: D NODES
- Console command D TASKS shows version of XNET of directly connected nodes
- Improve user terminal selection menu handling.
- Unhang user terminal at node selection menu if unable to connect to the destination node after a few seconds.
- Allow named configuration files on start up
- Bug fixes