CHAPTER 14

Troubleshooting XNS

The Xerox Network Systems (XNS) protocols were created by Xerox Corporation in the late 1970s and early 1980s. They were designed to be used across a variety of communication media, processors, and office applications. Several XNS protocols resemble the Internet Protocol (IP) and Transmission Control Protocol (TCP), developed by the Defense Advanced Research Projects Agency (DARPA) for the U.S. Department of Defense (DoD).

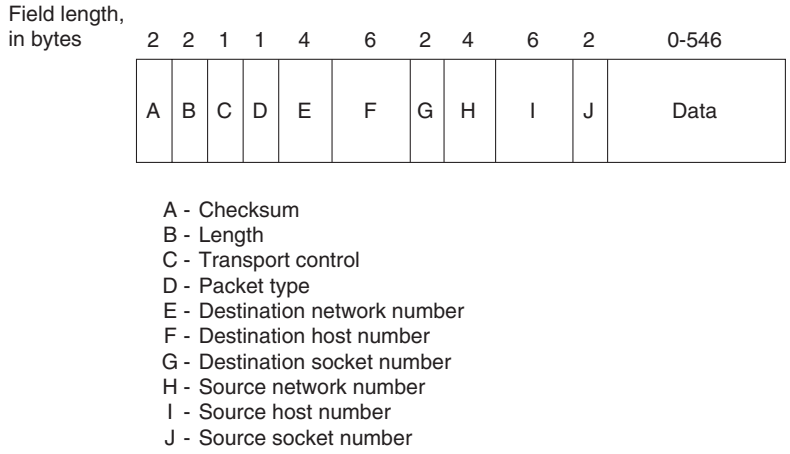
Because of its availability and early entry into the market, XNS was adopted by most of the early LAN companies, including Novell, Inc., Ungermann-Bass, Inc. (now a part of Tandem Computers), and 3Com Corporation. Each of these companies has since made various changes to the XNS protocols. Novell added the Service Advertising Protocol (SAP) to permit resource advertisement and modified the OSI Layer 3 protocols (which Novell renamed IPX, for Internetwork Packet Exchange) to run on IEEE 802.3 rather than Ethernet networks. Ungermann-Bass modified Routing Information Protocol (RIP) to support delay as well as hop count and made other small changes. Over time, the XNS implementations for PC networking have become more popular than XNS as it was designed by Xerox.

Although XNS documentation mentions X.25, Ethernet, and High-Level Data Link Control (HDLC), XNS does not expressly define what it refers to as a Level 0 protocol. Like many other protocol suites, XNS leaves media access an open issue, implicitly allowing any such protocol to host the transport of XNS packets over a physical medium.

The Network Layer

The XNS network-layer protocol is called the Internet Datagram Protocol (IDP). IDP performs standard Layer 3 functions, including logical addressing and end-to-end datagram delivery across an internetwork. The format of an IDP packet is shown in Figure 14-1.

Figure 14-1 The IDP Packet Format



The fields of the IDP packet are as follows:

- **Checksum**—A 16-bit field that helps gauge the integrity of the packet after it traverses the internetwork.
- **Length**—A 16-bit field that carries the complete length (including checksum) of the current datagram.
- **Transport control**—An 8-bit field that contains hop count and maximum packet lifetime (MPL) subfields. The hop count subfield is initialized to zero by the source and incremented by one as the datagram passes through a router. When the hop count field reaches 16, the datagram is discarded on the assumption that a routing loop is occurring. The MPL subfield provides the maximum amount of time, in seconds, that a packet can remain on the internetwork.
- **Packet type**—An 8-bit field that specifies the format of the data field.
- **Destination network number**—A 32-bit field that uniquely identifies the destination network in an internetwork.
- **Destination host number**—A 48-bit field that uniquely identifies the destination host.
- **Destination socket number**—A 16-bit field that uniquely identifies a socket (process) within the destination host.
- **Source network number**—A 32-bit field that uniquely identifies the source network in an internetwork.
- **Source host number**—A 48-bit field that uniquely identifies the source host.
- **Source socket number**—A 16-bit field that uniquely identifies a socket (process) within the source host.

IEEE 802 addresses are equivalent to host numbers, so a host that is connected to more than one IEEE 802 network has the same address on each segment. This makes network numbers redundant, but nevertheless useful for routing. Certain socket numbers are well known, meaning that the service performed by the software using them is statically defined. All other socket numbers are reusable.

XNS supports Ethernet Version 2.0 encapsulation for Ethernet and three types of encapsulation for Token Ring: 3Com, Subnet Access Protocol (SNAP), and Ungermann-Bass.

XNS supports unicast (point-to-point), multicast, and broadcast packets. Multicast and broadcast addresses are further divided into directed and global types. Directed multicasts deliver packets to members of the multicast group on the network specified in the destination multicast network address.

Directed broadcasts deliver packets to all members of a specified network. Global multicasts deliver packets to all members of the group within the entire internetwork, whereas global broadcasts deliver packets to all internetwork addresses. One bit in the host number indicates a single versus a multicast address. All ones in the host field indicate a broadcast address.

To route packets in an internetwork, XNS uses the dynamic routing scheme RIP. Today, RIP is still in use, but has largely been replaced by more scalable protocols, such as Open Shortest Path First (OSPF) and Border Gateway Protocol (BGP).

The Transport Layer

OSI transport-layer functions are implemented by several protocols. Each of the following protocols is described in the XNS specification as a Layer 2 protocol.

The Sequenced Packet Protocol (SPP) provides reliable, connection-based, flow-controlled packet transmission on behalf of client processes. It is similar in function to the Internet Protocol suite's TCP and the OSI protocol suite's Transport Protocol 4 (TP4).

Each SPP packet includes a sequence number, which is used to order packets and to determine whether any have been duplicated or missed. SPP packets also contain two 16-bit connection identifiers. One connection identifier is specified by each end of the connection. Together, the two connection identifiers uniquely identify a logical connection between client processes.

SPP packets cannot be longer than 576 bytes. Client processes can negotiate use of a different packet size during connection establishment, but SPP does not define the nature of this negotiation.

The Packet Exchange Protocol (PEP) is a request-response protocol designed to have greater reliability than simple datagram service (as provided by IDP, for example), but less reliability than SPP. PEP is functionally similar to the Internet Protocol suite's User Datagram Protocol (UDP). PEP is single-packet based, providing retransmissions but no duplicate packet detection. As such, it is useful in applications where request-response transactions can be repeated without damaging data, or where reliable transfer is executed at another layer.

The Error Protocol (EP) can be used by any client process to notify another client process that a network error has occurred. This protocol is used, for example, in situations where an SPP implementation has identified a duplicate packet.

Upper-Layer Protocols

XNS offers several upper-layer protocols. The Printing Protocol provides print services. The Filing Protocol provides file-access services. The Clearinghouse Protocol provides name services. Each of these three protocols runs on top of the Courier Protocol, which provides conventions for data structuring and process interaction.

XNS also defines Level 4 protocols. These are application protocols but, because they have little to do with actual communication functions, the XNS specification does not include any pertinent definitions for them.

The Level 2 Echo Protocol is used to test the reachability of XNS network nodes and to support functions such as that provided by the **ping** command found in UNIX and other environments.

Troubleshooting XNS

This section presents protocol-related troubleshooting information for XNS connectivity problems. It describes specific XNS symptoms, the problems that are likely to cause each symptom, and the solutions to those problems.

This section covers the most common network issues in XNS environments:

- XNS: Clients Cannot Connect to Servers over Router
- XNS: XNS Broadcast Packets Not Forwarded by Router
- XNS: Clients Cannot Connect to Server over PSN

XNS: Clients Cannot Connect to Servers over Router

Symptom: Clients cannot make connections to XNS servers across a router. Clients might be able to connect to servers on their directly connected networks.

Table 14-1 outlines the problems that might cause this symptom and describes solutions to those problems.

Table 14-1 XNS: Clients Cannot Connect to Servers over Router

Possible Problem	Solution
Router interface is down	1. Use the **show interfaces exec** command to check the status of the router interfaces. 2. If the status line indicates that an interface that should be up is "administratively down," use the **no shutdown** interface configuration command on the interface. 3. If the status line indicates that the interface or line protocol is in any other state, refer to the chapter that discusses your media type.
Hardware or media problem	For information on troubleshooting hardware problems, see the chapter that discusses your media type. For information on troubleshooting media problems, see Chapter 15, "Troubleshooting Serial Lines."

Table 14-1 XNS: Clients Cannot Connect to Servers over Router (continued)

Possible Problem	Solution
XNS routing is not enabled on router	1. Use the **show running-config privileged exec** command to view the router configuration. Check whether XNS routing is enabled on the router. 2. If XNS routing is not enabled, add the **xns routing** router configuration command and related commands as necessary. Example: This example starts XNS routing and assigns XNS network numbers to the physical networks connected to two of the router's Ethernet interfaces: `xns routing` `interface ethernet 0` `xns network 20` `interface ethernet 1` `xns network 21` For more information on configuring XNS routing, see the *Network Protocols Configuration Guide, Part 2*.
Mismatched router network number	If the network number specified on the router is different from that configured on XNS servers, RIP[1] is not able to forward traffic correctly. 1. Check the network numbers of network servers.The local XNS server administrator provides the server network numbers. 2. Use the **show xns interface exec** command to obtain the network number specified on the server side of the router. 3. Compare the network numbers. If they do not match, reconfigure the router or the server, as appropriate, with the correct network number. To reconfigure the router, use the following command:

continues

Table 14-1 XNS: Clients Cannot Connect to Servers over Router (continued)

Possible Problem	Solution
Mismatched router network number *(continued)*	**xns network** *number* The argument *number* is the network number, in decimal format. Every XNS interface in a system must have a unique XNS network number. Example: This example starts XNS routing and assigns XNS network numbers to the physical networks connected to two of the router's Ethernet interfaces: `xns routing` `interface ethernet 0` `xns network 20` `interface ethernet 1` `xns network 21` 4. If the network numbers match, check the router interface on the client side and make sure that the assigned network number is unique with respect to all network numbers in the XNS internetwork.

Table 14-1 XNS: Clients Cannot Connect to Servers over Router (continued)

Possible Problem	Solution
Misconfigured access list	1. Use the **show xns access-list privileged exec** command on routers in the path from source to destination. This command shows whether there are access lists configured on the router. 2. Disable all access lists that are configured on the router using the **no xns access-group** command. 3. Test the connection from the client to the server to see whether connections are now possible. If the connection is successful, an access list is blocking traffic. 4. To isolate the problem access list, apply one access list statement at a time until you can no longer create connections. 5. When the problem list is identified, alter it so that necessary traffic is allowed to pass. Configure explicit **permit** statements for traffic that you want to be forwarded by the router. 6. If problems persist, continue testing for problem access lists on all routers in the path from source to destination.

Table 14-1 XNS: Clients Cannot Connect to Servers over Router (continued)

Possible Problem	Solution
Backdoor bridge between segments	1. Use the **show xns traffic exec** command to determine whether the bad hop count field is incrementing. The XNS network updates by default occur every 30 seconds: `C4000#show xns traffic` `Rec: 3968 total, 0 format errors, 0 checksum errors,` `0 bad hop count,` `3968 local destination, 0 multicast` `[...]` 2. If this counter is increasing, use a network analyzer to look for packet loops on suspect segments. Look for routing updates. If a backdoor bridge exists, you will probably see hop counts that increment up to 15, at which point the route disappears. The route reappears unpredictably. 3. Use a network analyzer to examine the traffic on each segment. Look for known remote network numbers that appear on the local network. That is, look for packets from a remote network whose source address is not the source address of the router. The backdoor is located on the segment on which a packet from a remote network appears whose source address is not the source address of a local router. To prevent XNS routing updates from being learned from the interface connected to the same segment as the backdoor bridge, you can use the **xns input-network-filter** command. Example: In the following example, **access list 476** controls which networks are added to the routing table when RIP packets are received on Ethernet interface 1. Network 16 is the only network whose information will be added to the routing table. Routing updates for all other networks are implicitly denied and are not added to the routing table: `access-list 476 permit 16` `interface ethernet 1` `xns input-network-filter 476`

1. RIP = Routing Information Protocol

XNS: XNS Broadcast Packets Not Forwarded by Router

Symptom: XNS servers do not respond to broadcast requests from clients.

Table 14-2 outlines the problems that might cause this symptom and describes solutions to those problems.

Table 14-2 XNS: XNS Broadcast Packets Not Forwarded by Router

Possible Problem	Solution
Missing or misconfigured **xns helper-address** command	**Caution:** Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use. 1. Enable the **debug xns packet privileged exec** command and check the output for XNS packets that have an unknown type *xx* specification. 2. Use the **show running-config privileged exec** command to view the router configuration. Check the configuration of the client-side interface to see whether an **xns helper-address** interface configuration command entry is present. 3. If the **xns helper-address** command is not present, add it to the client-side interface. Syntax: **xns helper-address** *network.host* Syntax Description: • *network*—Network on which the target XNS server resides. This is a 32-bit decimal number. • *host*—Host number of the target XNS server. This is a 48-bit hexadecimal value represented as a dotted triplet of four-digit hexadecimal numbers (xxxx.xxxx.xxxx). The host must be directly connected to one of the router's directly attached networks. The number FFFF.FFFF.FFFF indicates all hosts on the specified network. Example: In the following example, the server at address 0000.0c00.23fe receives all broadcasts on network 51: `xns helper-address 51.0000.0c00.23fe` 4. If the command is present, make sure the MAC address specified in this command is a type of broadcast. Following is an example of an all-nets broadcast: `interface ethernet 0` `xns helper-address -1.ffff.ffff.ffff`

Table 14-2 XNS: XNS Broadcast Packets Not Forwarded by Router (continued)

Possible Problem	Solution
Missing or misconfigured **xns helper-address** command *(continued)*	The helper address specification differs depending on the network configuration. For more information, refer to the Cisco IOS *Network Protocols Configuration Guide, Part 2,* and *Network Protocols Command Reference*, *Part 2*.
Missing **xns forward-protocol** router configuration command	**Caution:** Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use. 1. Enable the **debug xns packet privileged exec** command and check the output for XNS packets that have an unknown type *xx* specification. 2. Use the **show running-config privileged exec** command to view the router configuration. Look for an **xns forward-protocol** global configuration command entry. 3. If the **xns forward-protocol** command is not present, add it as appropriate. Syntax: **xns** *forward-protocol* **protocol** Syntax Description: • **protocol**—Number of an XNS protocol, in decimal. See the documentation accompanying your host's XNS implementation for a list of protocol numbers.
Misconfigured access list	1. Use the **show access-lists** command to check whether there are access lists configured on the router. 2. Disable any access lists that are enabled on the router. 3. Test the connection to see whether connections are now possible. If the connection is successful, an access list is blocking traffic. 4. Enable access lists one at a time until connections are no longer possible. 5. Alter the problem list so traffic can pass. Configure explicit **permit** statements for traffic that you want to be forwarded by the router. 6. If problems persist, continue testing for problem access lists on all routers in the path from source to destination.

XNS: Clients Cannot Connect to Server over PSN

Symptom: Clients cannot connect to servers across a PSN. Clients can communicate with servers located on the local network.

Table 14-3 outlines the problems that might cause this symptom and describes solutions to those problems.

Table 14-3 XNS: Clients Cannot Connect to Server over PSN

Possible Problem	Solution
Address mapping error	1. Use the **show running-config privileged exec** command to view the configuration of the router. 2. If you are running X.25, make sure **x25 map xns** interface configuration commands are properly configured. Make sure MAC addresses and X.121 addresses are correctly specified. 3. If you are running Frame Relay, make sure **frame-relay map xns** interface configuration commands are properly configured. Make sure MAC addresses and DLCIs[1] are correctly specified.
Mismatched router network number	1. Check the network numbers of network servers. This information will be provided by the local XNS server administration staff. 2. Check the network number specified on the server side of the router. 3. Compare the network numbers. If they do not match, reconfigure the router or servers as appropriate, with the correct network number. 4. If the network numbers match, check the router interface on the client side and make sure the assigned network number is unique with respect to all network numbers in the XNS internetwork.
Encapsulation mismatch	1. Use the **show interfaces exec** command to determine the encapsulation type being used (such as **encapsulation x25**). 2. If an encapsulation command is not present, the default is HDLC[2] encapsulation. For PSN interconnection, you must explicitly specify an encapsulation type. To set the encapsulation method used by the interface, use the **encapsulation** interface configuration command. Syntax: **encapsulation** *encapsulation-type*

Table 14-3 XNS: Clients Cannot Connect to Server over PSN (continued)

Possible Problem	Solution
Encapsulation mismatch *(continued)*	Syntax Description: • *encapsulation-type*—One of the following keywords: • **atm-dxi**—Asynchronous Transfer Mode-Data Exchange Interface. • **bstun**—Block Serial Tunnel. • **frame-relay**—Frame Relay (for serial interface). • **hdlc**—HDLC protocol for serial interface. This encapsulation method provides the synchronous framing and error detection functions of HDLC without windowing or retransmission. • **lapb**—X.25 LAPB DTE operation (for serial interface). • **ppp**—PPP[3] (for serial interface). • **sdlc**—IBM serial SNA.[4] • **sdlc-primary**—IBM serial SNA (for primary serial interface). • **sdlc-secondary**—IBM serial SNA (for secondary serial interface). • **smds**—SMDS[5] (for serial interface).

1. DLCI = data link connection identifiers
2. HDLC = High-Level Data Link Control
3. PPP = Point-to-Point Protocol
4. SNA = Systems Network Architecture
5. SMDS = Switched Multimegabit Data Services