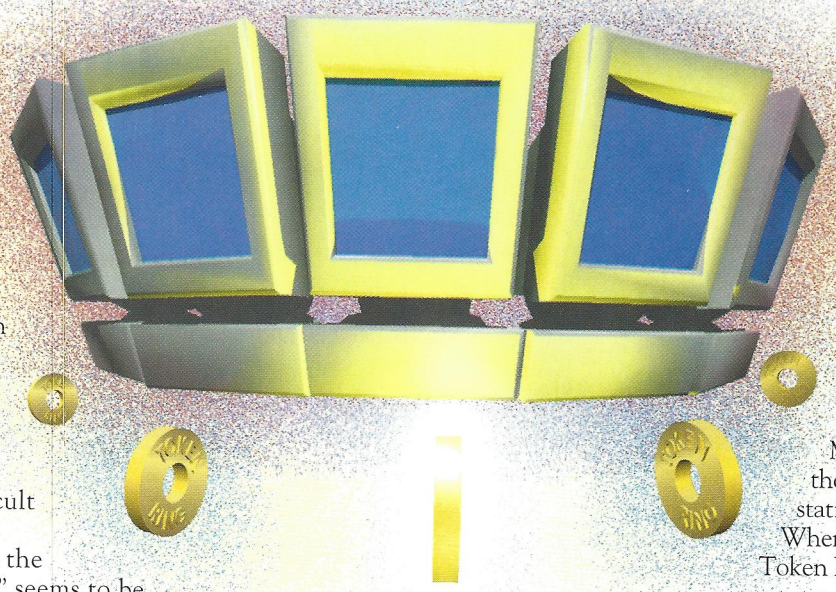
TROUBLESHOOTING

Identifying and Eliminating Problems on Token Ring Networks

By Laura Chappell

This article is part 2 of a three-part series that focuses on troubleshooting and optimizing Ethernet, Token Ring, and NetWare® communications using a protocol analyzer such as Novell's LANalyzer for Windows, LANalyzer Network Analyzer by Network Communications Corporation, or, in this case, Novell's NetWare Management System with NetWare LANalyzer Agents.



When I present network analysis seminars, I ask the audience to raise their hands if they have a Token Ring network. I oftentimes make some comment about the slashed wrists visible in the room when they raise their hands. Generally, people laugh; sometimes people groan. Token Ring networks have a much maligned reputation of being difficult to troubleshoot and get running properly: "Once the ring is up, don't touch it" seems to be a common networking hint among Token Ring users.

In part 2 of this three-part series, we focus on the functionality of Token Ring networks and then delve into various methods for troubleshooting rings that are performing inadequately or are not performing at all.

Before we begin troubleshooting some Token Ring networks, let's examine how a Token Ring network operates.

Illustration: Dave Bronson

Entering the Ring

On a Token Ring network, stations are physically connected to a centralized hub device called a Multistation Access Unit (MSAU). The cable that connects a workstation to the MSAU is called a lobe cable. Although the physical layout of the network is a star, Token Ring is actually a logical ring, with data traveling from station-to-station in a counterclockwise direction, as shown in Figure 1.

When a station's Token Ring network driver is loaded, an electrical current is sent up the cable to the MSAU. The current causes the relay in the MSAU to open up, thereby allowing the station to enter the ring. When you unload your Token Ring network driver, the current stops, the relay closes, and you are no longer in the ring.

Token Ring networks depend on the integrity of the entire logical ring to maintain operations. If the ring is opened, data cannot circulate, and the ring is down. When stations load and unload network drivers and affect the relays in the MSAU, the ring experiences momentary breaks. These short breaks cause a Token Ring error called a "burst." Just as collisions are considered a normal part of Ethernet

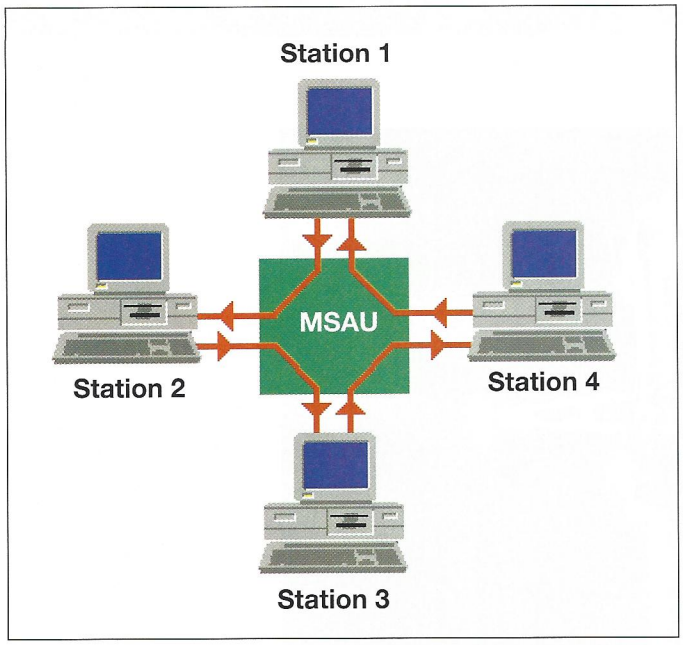


Figure 1: On a Token Ring network, data travels from station-to-station in a counter-clockwise direction.

802.3 network operations, burst errors are considered a normal part of Token Ring network operations.

Transmitting on the Ring

On a Token Ring network, stations must wait for a special 24-bit sequence, known as a token, before they can transmit data. Only one token is on the ring at a given moment.

When a station has data queued to transmit and it receives the token, the station "captures" the token and transmits a Token Ring frame. The frame travels in a counter-clockwise direction around the ring.

When ring stations receive a frame, the stations act as a repeater by repeating the frame onto the ring after it has been viewed. The station that the frame was addressed to buffers the frame and actually changes it slightly before transmitting it back onto the ring.

Each frame has two fields: Address Recognized Indicator and Frame Copied Indicator. The destination station sets the Address Recognized Indicator bits from 0 to 1 to indicate that it has recognized that the frame was addressed to its network address. If the station has buffer space available, it will copy the frame into the buffer space and set the Frame Copied Indicator bits from 0 to 1.

The frame continues around the ring until the sending station receives it. The sending station checks the frame for errors, and looks at the Address Recognized Indicator and Frame Copied Indicator bits to determine if the transmission was successful. The method in which the next token is released depends on the type of Token Ring network in use.

Three Types of Token Ring

Three types of Token Ring networks are available: 4Mbit/s, 16Mbit/s, or 16Mbit/s Early Token Release. The primary difference between the 4Mbit/s and 16Mbit/s Token Ring network types are the media access speeds. You must decide on one media access speed; you cannot mix 4Mbit/s Token Ring boards with 16Mbit/s boards. The ring will not function with both.

On a 4Mbit/s Token Ring network, data travels at 4Mbit/s around the ring. When a frame arrives back at the sending station on the ring, the station must strip off the frame and release a token. The next station on the ring (the downstream neighbor) will receive the token next. If the downstream neighbor has data to send, it can "capture" the token and transmit a frame.

On 4 Mbit/s and 16 Mbit/s Token Ring networks, only a single frame or a single token can be transmitted on the network at any time.

On a 16Mbit/s Early Token Release (ETR) network, however, a transmitting station does not have to receive its frame back and strip it off the ring before issuing a new token. The transmitting station issues a token immediately after sending its frame. In this case, a frame travels around the ring followed by a token. If the downstream neighbor to the sending station wants to transmit data, it will repeat the frame first and then "capture" the token following the frame. The downstream neighbor will then transmit a second frame on the network, followed by a token. There is still only one token on the network at any time.

The advantage of ETR is the ability to have multiple frames on the network at any given time. The interesting fact, however, is that most rings are not long enough to hold multiple frames and, therefore, cannot take full advantage of ETR. For example, a single 24-bit token can take up approximately 1.5 kilometers of cable on a 16Mbit/s. Of course, most rings are smaller, and therefore, the token is buffered on the ring. Imagine how much cabling you must have to support multiple 4KB frames. The real advantage of ETR is the ability to transmit a token immediately after transmitting a frame.

The Token Ring Management System

One of the unique characteristics of Token Ring networks is the inherent self-management capability built into the protocol. On a Token Ring network, many communications problems are solved automatically without manual

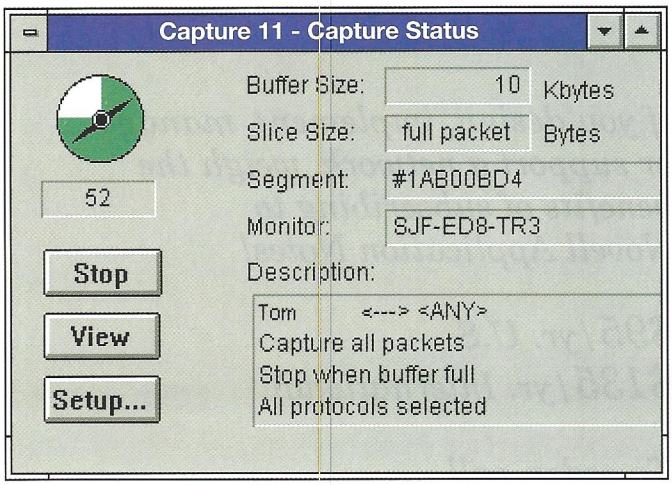


Figure 2: The NMS Console is set up to capture only packets to or from Tom's station.

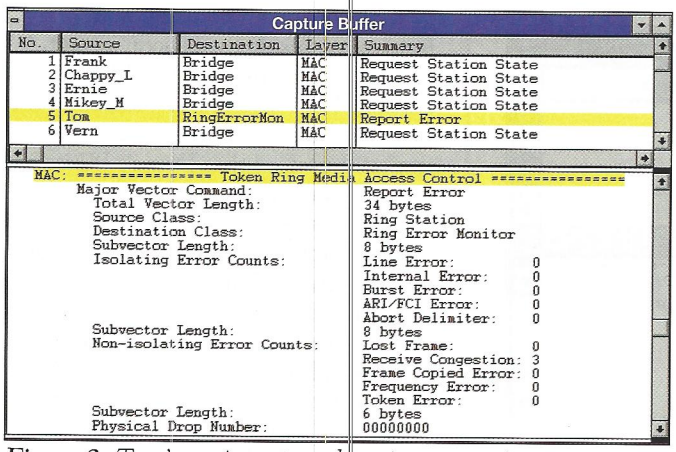


Figure 3: Tom's station reported receiver congestion errors.

intervention. For example, if a station is suspected of causing an "open" in the ring, that station must remove itself from the ring and perform a self-test. If the station fails the self-test, it cannot reenter the ring.

The Token Ring management system includes several special functions. For example, each ring must have one station that acts as the Active Monitor. This duty is automatically assigned and is transparent to the station user.

The Active Monitor station is the only station that can purge all data from a malfunctioning ring and release a new token on the ring to "jump start" the ring after a failure. The Active Monitor also triggers a process known as the Ring Poll every seven seconds on the ring. The Ring Poll process enables each ring station to learn who its nearest active upstream neighbor is.

On a Token Ring network, each station uses this information to "fink" on a neighbor that may be causing errors. For example, if Station 1 in Figure 1 doesn't receive any frames or tokens within a certain amount of time, Station 1 assumes that its upstream neighbor, Station 4, is at fault. Station 1 begins to announce that there is a fault on the network and Station 4 is the most likely cause. Station 4 must remove itself from the network and perform a self-test before returning.

Each station also maintains an error timer and reports errors to a functional station called a Ring Error Monitor. You need special software to enable a station to act as a Ring Error Monitor. The Ring Error Monitor simply tracks errors that stations are reporting on the network. This information is also visible using a network analyzer such as Novell's NetWare Management System (NMS) with the NetWare LANalyzer Agents. These error reports give clear indications of potential problems on the network.

The remainder of this article focuses on two Token Ring networks that are experiencing poor performance and connectivity problems. It examines the symptoms of the problems (some may seem all too familiar to you), and then explains each step of the troubleshooting process, beginning with the isolation and identification of the problem.

Case Study 1: Poorly Performing Station

Tom, a user on a 4Mbit/s Token Ring network, contacted me to complain about his slow workstation. When I typed SYSCON, the station seemed to lock up for about 10 seconds, and suddenly the SYSCON menu appeared. Tom was understandably frustrated with the delay. Tom had not changed any applications on his station for the past three months, but response time had progressively worsened.

The first step, of course, was to check the local station's configuration to ensure the system was not experiencing any delay due to memory configuration errors or such. Once that was verified, I decided to take advantage of the Token Ring network's inherent management by examining the communications on the wire. In this situation, I used the NMS console located on the primary network backbone and set up a packet capture session that would capture only packets to and from Tom's station, as shown in Figure 2.

I immediately spotted the problem when I viewed the packets that Tom was sending to the Ring Error Monitor. Tom's Token Ring board was reporting receiver congestion errors, as shown in Figure 3.

One of the advantages of using the NMS console with the NetWare LANalyzer Agents is you can configure the agents to notify you the minute such situations become evident on the network. In this case, I only had to view the NMS network map to see that errors were being reported on the ring.

Receiver congestion errors are caused when frames are sent to a station that does not have the buffer space to copy the packets. When Tom's station receives frames and doesn't have sufficient buffer space, the Token Ring chipset sets the Address Recognized Indicator bits to 1 (indicating that Tom has recognized the frames are addressed to his station), but the chipset leaves the Frame Copied Indicator bits set at 0 (indicating that the board did not buffer the frame).

Receiver congestion errors are often caused by a Token Ring station that cannot keep up with the traffic on the network because it does not have enough receive buffers. Most Token Ring boards enable you to set the receive buffer size using dip switches on the board. In this instance, for example, Tom was using an

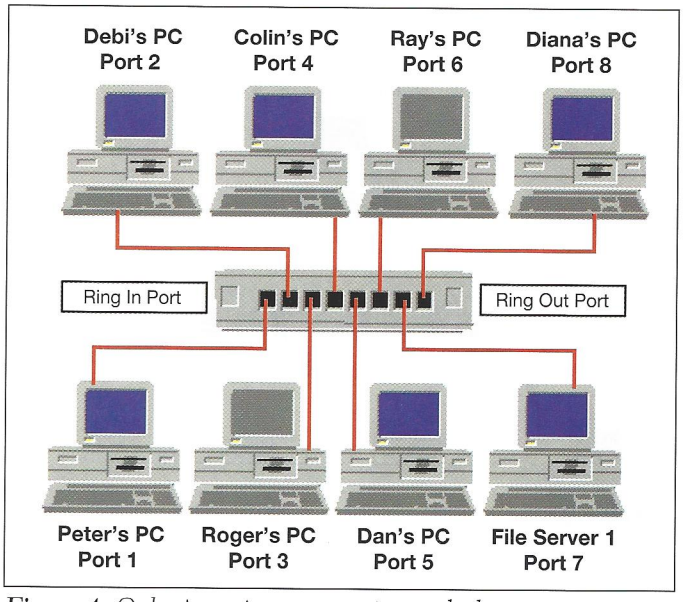


Figure 4: Only six stations were active on the beaconing ring.

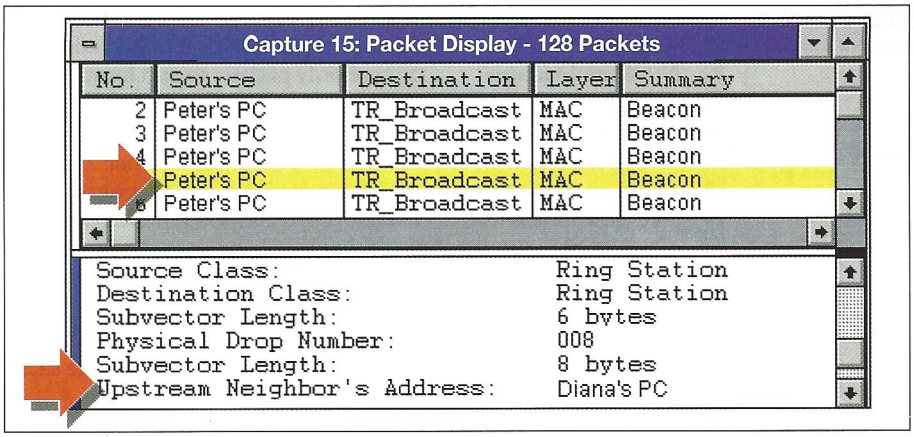


Figure 5: *Beacon packets indicate there is an opening in the ring.*

older IBM Network PC Adapter II board, so I consulted the manual to check the dip switch settings for shared RAM size. I set the switch to 16KB, and the station had twice the amount of memory for buffering packets on the network. Tom noticed an immediate improvement in the station's response time.

Case Study 2: A Beaconing Ring

If you have worked on a Token Ring network, you have probably experienced (at least once) the dreaded beaconing ring, which is caused by an opening in the ring. As mentioned before, the ring must be a closed loop for data to circulate properly. No data travels on a beaconing ring—in effect, the ring is down.

I was recently asked to troubleshoot a beaconing ring. The server console was generating error messages such as "Adapter 1 is beaconing." When my clients tried to load a Token Ring network driver, they received the message: "The ring is beaconing . . ."

Although the thought of a beaconing ring often makes a network troubleshooter's blood run cold, the beacon process can be a blessing in isolating the fault on the network. On this beaconing ring, the first step was to isolate the fault domain, which is the physical area of the ring fault. For example, if the ring is open because of a cable break between two MSAUs, I needed to know which two MSAUs.

This error occurred on a small Token Ring network, as shown in Figure 4. Two of the stations, Ray's and Roger's PCs, were not powered up when the beaconing condition occurred. To isolate the fault domain, I once again turned to the NMS console and began capturing packets on the crippled ring. This time, however, I specified that I wanted to capture all media access control (MAC) packets on the ring. MAC packets are management packets on Token Ring networks. When I viewed the capture buffer, I specifically looked for beacon packets, as shown in Figure 5.

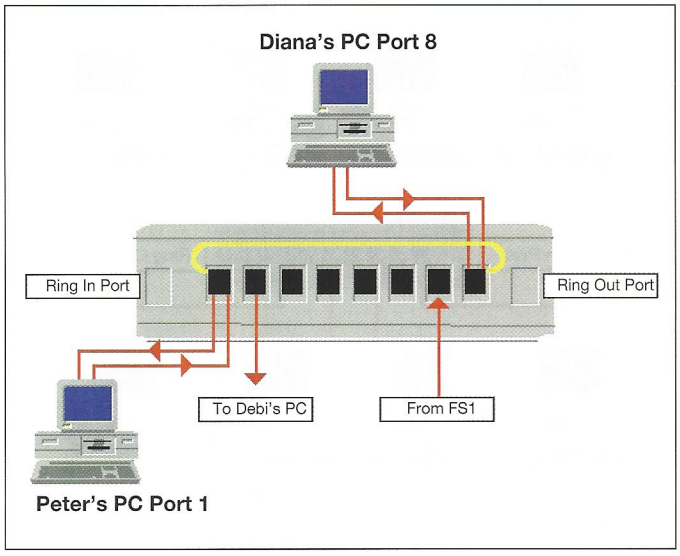


Figure 6: *The logical flow of frames and tokens in the fault domain.*

I noticed that Peter's PC was beaconing. Inside the beacon packet, I looked for the upstream neighbor's address. The NMS console indicates the name of the user logged in using that station. I noticed that Diana's PC is the upstream neighbor.

Beacons are caused when a station suddenly stops receiving frames or tokens on the network. This is clearly an indication that the ring is open. To isolate the fault domain, I looked at the wiring of the Token Ring network and noticed where Peter's station was plugged in to the MSAU and where Diana's station was plugged in to the MSAU.

The fault domain lies between the beaconing station and the station's nearest active upstream neighbor. This includes the MSAU ports between the two stations and the ports on ends of the MSAU—the ring in and ring out ports. Figure 6 shows the logical flow of frames and tokens between these two stations.

As shown in Figure 6, data moves from Diana's station into the loopback feature of the MSAU. The data should not go out the Ring Out port because no cable is attached to the port. Unlike all other ports, the relays in the Ring Out and Ring In ports open up any time a cable is physically inserted into the port. If one of these ports sticks open for some reason, the ring is open.

First, I disconnected Diana's station to see if the beacon condition is caused by a fault in that area. The beaconing continued, but this time Peter's PC listed the file server as the upstream neighbor. This indicates that Diana's PC is not at fault. Next, I disconnected Peter's PC from the network. The next station, Debi's, began beaconing. The fault, therefore, is most likely due to an open Ring Out or Ring In port on the MSAU. I replaced the MSAU, and the ring no longer beaconed.

These were two examples of how a network analyzer can quickly determine the cause of network problems. The next article in this series focuses on analyzing NetWare communications to troubleshoot and optimize the network.

Laura Chappell is a partner of Technology Consortium in San Jose, California. Technology Consortium develops interactive multimedia products for training and product support systems. Ms. Chappell can be reached on CompuServe at 73424,1317.