

Implementing IOS Async Terminal Services

Michael J. Martin

This is the first of a two-part series on implementing an asynchronous serial line terminal server using a Cisco 2511 with 16 fixed asynchronous serial ports or 26x0, 36x0 or 37x0 series modular routers using NM-16A or NM-32A. Cisco terminal servers are a popular alternative to Unix/Linux and Windows servers configured with multi-port serial cards for deploying out-of-band (OOB) access to routers and switches. They can also provide for PPP, SLIP and PPTP remote access or dial-out modem pools. This month's article will focus on theory, especially the need for and problems of implementing OOB access, basic RS-232 operation and function of null modems. In part two, we will focus on the practical -- implementing a Cisco router as a terminal server providing direct asynchronous access to Cisco routers, along with basic dial-in and dial-out services.

The Problem of OOB Access

The need for OOB access is obvious. Networks go down, devices crash and you need to get access to them and get them back up. Perhaps you're looking for a better OOB access solution than the one you currently have in place. Maybe you're having problems with your Cisco terminal server solution and looking for answers. The odds are, however, that you have stumbled across this article as part of a Google search trying to figure out how to implement OOB in your current network. Most likely, an IOS upgrade went south and you had to send someone to the data center to perform password recovery on a router with a lost password, or -- even more gruesome -- you lost all network access to the data center and had no way to troubleshoot. Those looking for wisdom, this series should help. Those looking in a crazed, panic-stricken state for a solution -- shame on you; you should have been better prepared. But have no fear, there is help here for you, too.

OOB access is one of those necessary evils that all network administrators must eventually wrestle with. Ideally, you should consider it when you are designing your network, not debugging it. The idea behind OOB access is simple enough: To deploy a means for providing secondary, direct access to a network device or host for use when the primary access method is unavailable. Pretty straightforward, right? You would think so, but you're reading this article, and if you ask around you'll find that most administrators have overlooked OOB. Why? Well, let's take a close look at the definition.

The first problem is "access." It is mentioned both at the beginning and end of the definition. An OOB solution needs to operate alongside the production network. This requires additional infrastructure that often goes unutilized, because the OOB solution cannot be dependent on the primary network to operate. If the solution is dependent and the primary network goes down, in addition to a broken network, you also have a broken OOB solution. In most cases OOB is built around PSTN (Public Switched Telephone Network) connected modems. Phone lines must be provisioned, installed and -- most importantly -- not touched. If a remote office support person takes the line and hooks up a fax machine to it, you are going to be out of luck when you really need some.

This is why regularly checking the status of your OOB solution is quite important. Not only does it identify line theft, it also verifies that you can connect consistently. Getting through may not be a major concern if you are dialing locally, but it is a major factor if your maintaining infrastructure outside of North America or Western Europe. The quality of PSTN service varies throughout the world. Remember, the PSTN was designed for voice, which can be noisy and a little choppy. Data, however, does not bode well in that kind of environment. Regular testing is important to make sure you will have access when you need it. See, this is already getting a little complicated.

Then, as if connecting the equipment were not enough, you have to wrestle with RS-232. In the case of Cisco routers and switches and most Unix/Linux platforms, low-level access is provided over a serial port using a variation of the Electronics Industries Association (EIA) RS-232 protocol. The RS-232 protocol provides the physical, electrical and logical interface between data terminal equipment (DTE) (i.e., the router or computer) and data communications equipment (DCE) (i.e., the modem or

Implementing IOS Async Terminal Services

Michael J. Martin

serial converter/terminal). RS-232 is also one of the most abused protocols ever issued. Implementations of the protocol vary depending on the vendor in regards to:

- Port configuration or presentation, i.e., DTE vs. DCE
- Physical interface, i.e., DB-25M, DB-25F, DB-9M, DB-9F, or RJ-45M
- Signaling characteristics, i.e., support for flow control, DSR, DCD, CTS or RTS

What RS-232 "abuse" means to you as an administrator is a whole lot of work deciphering which combination of cable types and adapters will allow you to communicate with your most critical network components.

That brings us to OOB issue of "direct access," which heralds our second problem -- security. What do we mean when we say direct access? Direct access affords the administrator complete control over the system, as if she were next to it. It allows full functionality and accessibility to the systems bootstrap (pre OS load state) along with the ability to monitor and interact with the devices' boot process. While preparation for network calamity is a main driver for implementing OOB, there are a many other cases where direct access to the device may be needed. Here are just a few:

- Password recovery
- Operating system upgrades/recovery
- Performing system backups and restores
- Low-level hardware diagnostics

The types of tasks that need be managed using this level of access makes implementing OOB a security problem (and in many cases just a security hole). Why? Well, for starters, you are setting up access to a critical infrastructure component over an interface that is intended to be used for local administrative access, quite often utilizing a modem (not the securest method of communication). The "Console" level access port on many Unix/Linux systems provides the ability to drop the system into maintenance or single-user mode, which allows the user accounting system to be bypassed and/or simply halted. In the case of IOS-based devices, the Console port is just plain special. How special, you ask? First and foremost, the Cisco Console port is always active. If the router is powered on, the Console port is potentially transmitting (and can be receiving) data. This special port is also graced with the following special abilities:

- It can receive a "break" signal, which can (if configured) halt the router's operation
- It provides reconfiguration access of basic operational parameters via ROMMON
- It can be used to upload IOS images onto the flash file system or directly into memory via X-Modem or Y-Modem
- It provides the only monitoring access point for the router's boot process (critical when doing IOS upgrades)

Although the deployment of a terminal server enhances an administrator's ability to support the network, the degree of access provided by the "ports" to which the server is a gateway demands stringent security. At a minimum, two-layer authentication should be implemented. That requires that both the terminal service device and the host devices force user authentication, in addition to local authentication. In addition, adequate session attempts, session-timeouts and exec-timeouts should be set.

Implementing IOS Async Terminal Services

Michael J. Martin

With that said, I leave you with this thought: The plans to implement any network component that will have major implications on the security posture of the network, like a firewall, intrusion detection or a terminal server, should go through some type of formal impact review with your security, support and administration teams. (If you're a small shop, at the very least you should be discussing this with you boss.) Systems of this nature need clearly defined access policies and those that need to know need to be aware they are in place. Otherwise, you or your company could find yourself in an uncomfortable position.

RS-232 Basics

No discussion on implementing serial terminal services can take place without understanding the basics of the RS-232 protocol. The two most commonly implemented asynchronous serial standards are EIA RS-232-C, issued in 1969, and EIA RS-232-D, issued in 1987. The major difference is that the D standard provides remote and local testing facilities and some shielding enhancements. The standard defines transmission rates between the DTE and DCE up to 20,000 bps (2.5 KB/s) with a cable length in excess of 50 feet. However, faster rates may be achieved with shorter lengths. Less than 20000 bps requires the length to be less the 12 feet, where rates slower the 20,000 bps can be maintained on cable lengths that exceed 50 feet.

The standard defines a DB-25 connector providing 25 circuits for data interchange between the DTE and DCE. Signaling uses a +15V to -15V range with a +3V to -3V transition. In respect to data transmission, a voltage between +3V and +15V equates the line as "active" representing a binary 0, where a line voltage between -3V and -15v equals a binary 1. In respect to handshaking and control circuits, the opposite is true. Positive voltage represents a binary 1 and low voltage equates to binary 0. As indicated earlier, RS-232 implementations vary depending on the operational requirements of the DCE. A typical implementation will utilize between nine and 12 circuits. Some implementations, such as the one implemented for use as the Cisco Console port, use as little as six. Here is a table that defines the basic functions and their associated pin/circuit placement on the cable.

Pin (DB25, J45)	Function	Definition
1	Protective ground (GND)	Pin provides electrical earth ground.
2, 3	Transmit data (TD)	Raw data is transmitted from the terminal device (TD) to the modem. When the TD is not transmitting data, the line is marked active by sending a steam of binary 1s.
3, 6	Receive data (RD)	The TD receives demodulated raw data from the modem. Similar to TD, when no data is being received the line is marked active by sending a steam of binary 1s.
4, 1	Request to send (RTS)	The TD sends a handshaking signal to prime the modem, indicating it wants to transmit data. The RTS band is also used for hardware flow control signals with modems equipped with HW flow control. Under normal operation, the RTS signal is active, indicating the modem can transmit data to the TD.
5, 8	Clear to send (CTS)	The modem sends this handshaking signal to the TD to indicate the modem is ready to receive data to transmit. The CTS signal is sent in response to RTS. When the CTS signal drops, the TD should stop sending data to the modem. The modem will only signal CTS when DSR and CD are active. Under normal operation, the CTS signal will be active for the duration of the session. When hardware flow control is in use, the voltage will

Implementing IOS Async Terminal Services

Michael J. Martin

		flux to indicate the need for a temporary break in the transmission of data.
6, 7	Dataset ready (DSR) or data communication equipment ready (DCE-R)	This circuit relays the status of the modem to the TD. When active, the modem is online and ready to send data. Auto-dialing or DCE devices that do not support DSR will pass this signal regardless of status.
7	Signal ground (SG)	Provides ground reference for timing and control. While the GND and SG can independent, this pin is typically tied to pin 1.
8	Data carrier detect (DCD)	The modem uses this circuit to indicate the detection of carrier signal from a remote modem. The carrier signal is the tone used by modems to exchange data. Signal modulation (i.e., variations in amplitude, frequency and phase) relay the binary data streams over the PSTN lines. A loss of carrier signal indicates the loss of the upstream connection and terminates the connection.
20, 2	Data terminal ready (DTR) or data terminal equipment ready (DTE-R)	This handshaking channel controls the modem's PSTN connection. An active circuit indicates that a TD is connected to the modem and ready. A passive circuit indicates that the modem sees no TD device and the PSTN connection should be dropped.
21	Ring indicator (RI)	This channel signals from the modem to the TD that a ring is being received from the PTSN. This signal is active once every five seconds when an incoming call is detected. Either the modem or TD can pick up the call.

Looking at the circuit definitions, it would seem that it is possible to exchange data using only three wires: GND, TD and RD. Without handshaking, the DTE and DCE are unable to agree on when to communicate. Here is the handshaking exchange that occurs between a DTE (computer) and DCE (modem):

- The modem sends the handshaking signal DSR to the TD.
- The modem detects an incoming call. The RI circuit relays the ring to the TD and the TD responds with a DTR.
- The remote modem transmits a carrier tone, indicating that the call is another modem. The two modems will then verify the continuity of the line and "train" to the highest mutually supported data rate the PSTN line can support. Once the connection has been established, the local modem sends a CD handshaking signal to the TD indicating this is a data call.
- The TD will then respond with a RTS handshake.
- The modem will respond with a CTS handshake.
- Now data exchange can take place using the TX and RX circuits.

It is in the handshaking and control circuits that RS-232 compatibility usually falls short. Most of the time, this is due to the number of handshaking channels and the means by which they have been

Implementing IOS Async Terminal Services

Michael J. Martin

implemented. In terms of actual effect, this varies depending on how the ports are being used. In situations where null modems are used to connect TDs, the impact is often negligible. Where your mileage may vary is when you use actual modems.

Null Modems

Routers and switches are DCE devices but utilize DTE serial ports for their console access. While a directly attached modem is fine for a single device, it is not a solution that scales. Null modems are used to translate the handshaking and transmission circuits between two DTEs. By cross-connecting the TD and RD pins along with the needed handshaking channels -- RTS to CTS and DTR to CD and DSR -- both DTEs operate as if they were connected over a modem. Here are the pin-out translations for DB-25, DB-9, and Cisco's "Rolled" RJ45 cables (NS = not supported):

Signal name	DB-25	DB-9	RJ45	To	RJ45	DB-9	DB-25	Signal name
FG (Frame ground)	1	NS	4	5	NS	1	FG	
TD (Transmit data)	2	3	3	6	2	3	RD	
RD (Receive data)	3	2	6	3	3	2	TD	
RTS (Request to send)	4	7	8	1	8	5	CTS	
CTS (Clear to send)	5	8	1	8	7	4	RTS	
SG (Signal ground)	7	5	5	4	5	7	SG	
DSR (Data set ready)	6	6	7	2	4	20	DTR	
CD (Carrier detect)	8	1	NS	NS	4	20	DTR	
DTR (Data terminal ready)	20	4	2	7	1	8	CD	
DTR (Data terminal ready)	20	4	NS	NS	6	6	DSR	

Now that you know more than you may have wanted to know about OOB fundamentals and RS-232, our time is up. As always, questions, comments and pet peeves are welcome. Please send them to us. Don't forget to stay tuned and catch the conclusion next month. We'll put this knowledge to good use implementing a Cisco terminal server.