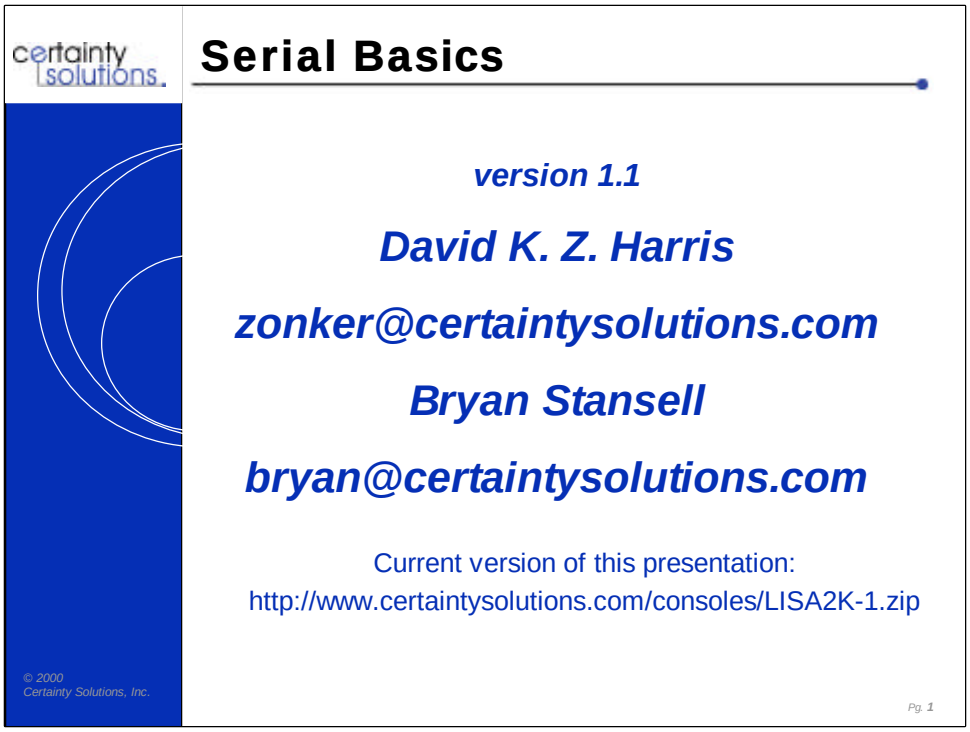
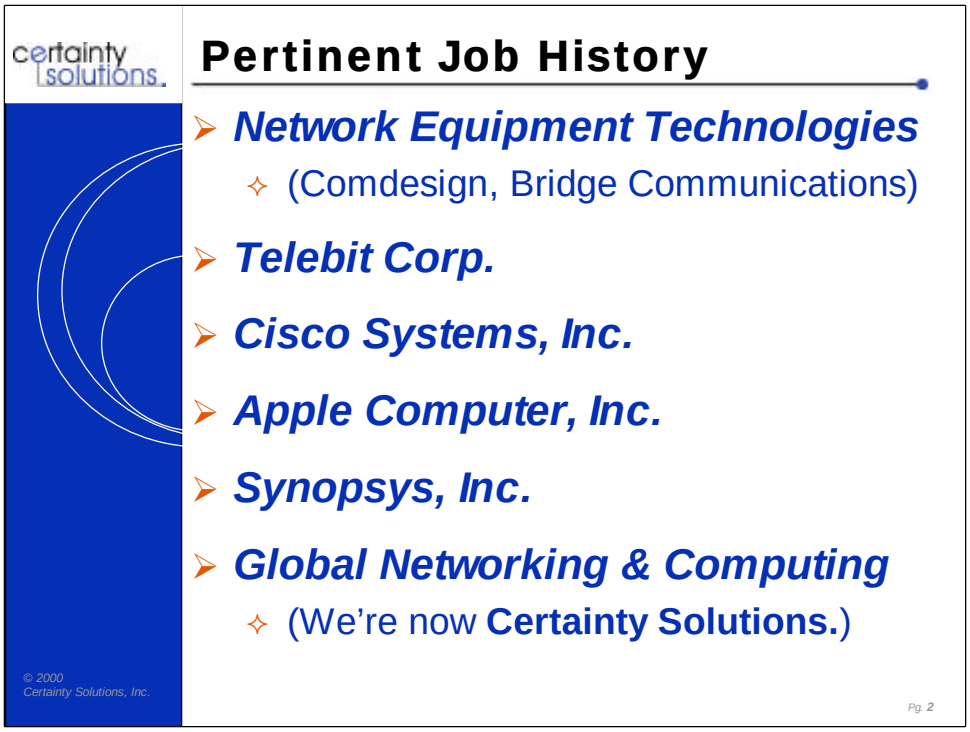


This presentation is a supplement to my console services web pages located at **http://www.certaintysolutions.com/consoles/**.

These pages have a substantial amount of information noted below each slide. We do this to help minimize the amount of note-taking that you need to do in class, and this should give you more time to listen to the instructors.

If you feel that you learn better by taking notes, please feel free to do so.



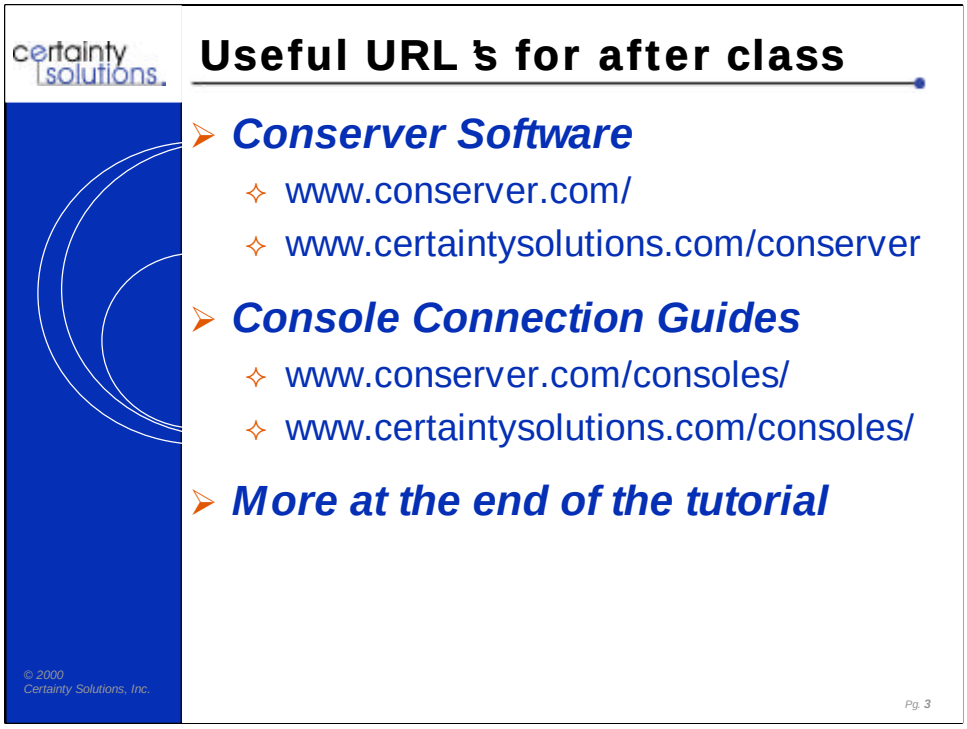
Before moving into networking, David Harris was a hardware hacker, working in repair and R&D roles. He has been tinkering with serial devices for more than a decade.

"My experience, plus reading plenty of manuals, has taught me that there are a few safe bets that you can make when working with unknown serial devices. I've also learned a few tricks that make the job of connecting serial devices to terminal servers easier. I'll share these tricks throughout this presentation."

Bryan Stansell is a Systems Administrator, and has been programming for many years, as an extension to his systems administration tasks, because he enjoys programming.

Bryan has been the keeper of the Conserver application for more than five years, having taken on the program while working for Arnold De Leon at Synopsys, Inc., and he'll help answer questions related to the Conserver application and client software.

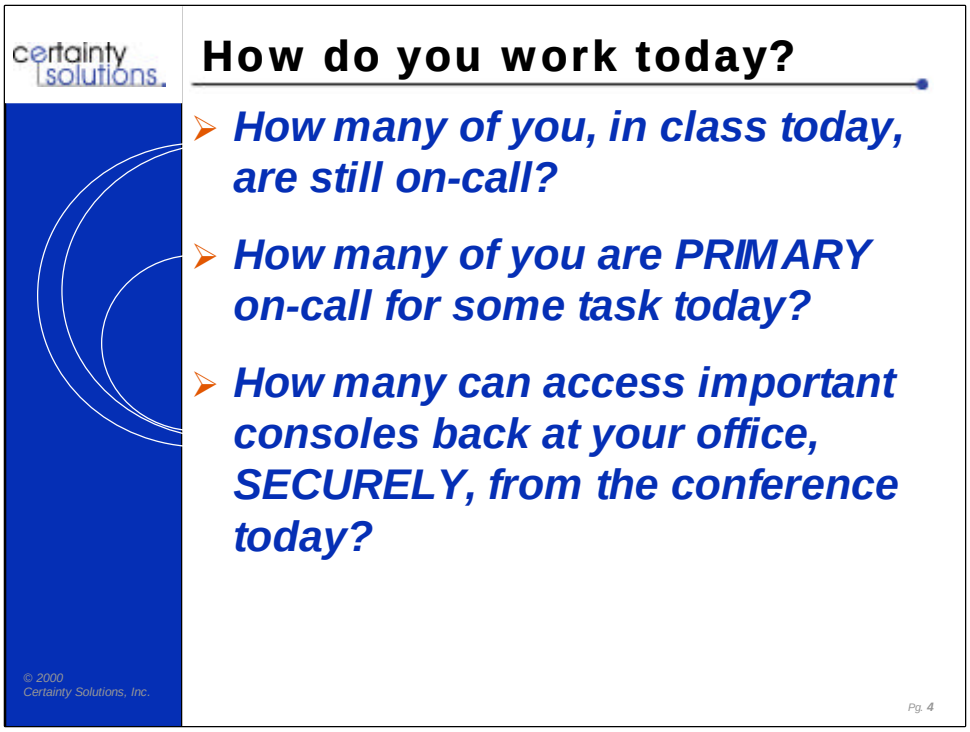
Bryan was one of the original employees of Global Networking & Computing (GNAC, Inc.), more than 4 years ago. David Harris has been with the company for nearly three years. In August 2000, GNAC became Certainty Solutions.



Celeste Stokely has a wonderful archive at http://www.stokely.com/, with lots of useful information for system administrators. Of particular interest is the page related to serial consoles;

http://www.stokely.com/unix.serial.port.resources/index.html

The serial switch link on this page has info about keyboard/video/mouse switches, RS-232 serial switches, and terminal server devices and console servers. (And if you aren't sure what some if these are, you're in the right class.)



If you support many machines, you may be called often to check on one of many devices,perhaps in many different locations.

If you only support a few critical machines, being able to diagnose a problem quickly is important to you.

Maybe you need to support your companies e-presence servers, In this case, time is money, and downtime is money lost!

Remote access to your serial consoles can let you quickly diagnose a problem, and can help you recover from many problems. It can even give you the clues you need to decide if you have to go visit the machine personally, but you will also probably know what parts you need to bring with you, which will save you a second trip.

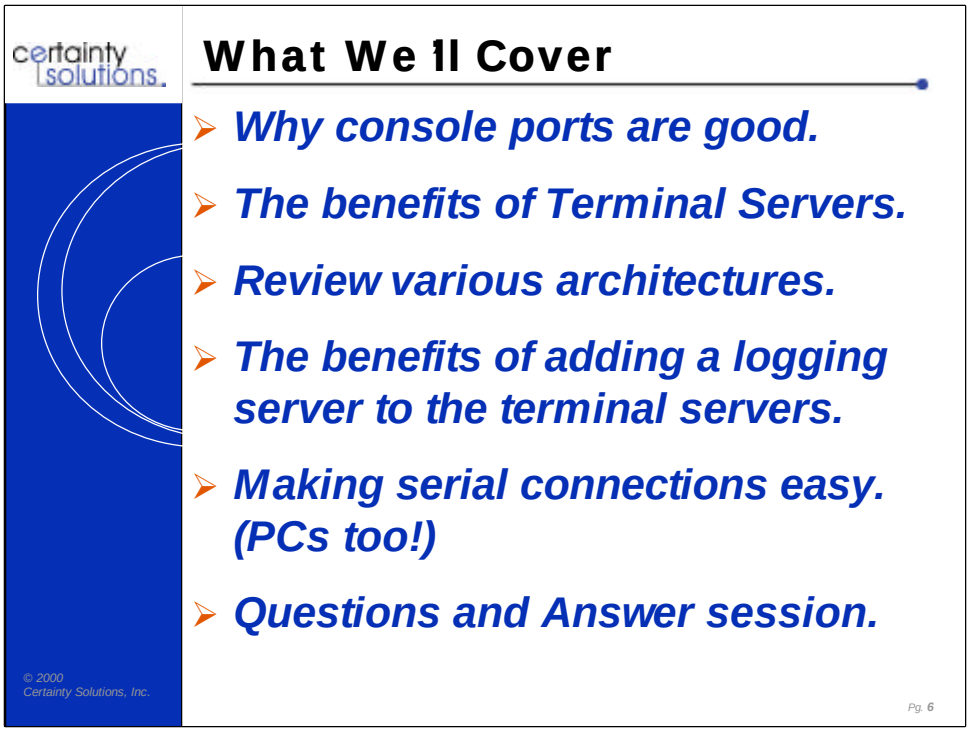
Sure, your job could become a bit more sedentary, but you'll be able to go and get some exercise, knowing that you can quickly get on the console of any device that you care about if the need arises.

When you are trying to monitor traffic on an Ethernet interface, you usually don't want the monitor output traffic to come back to you over the same interface you are monitoring.

You can also use a serial console to report security events. There are some messages that are only available from serial consoles. On a Cisco router, when someone makes a configuration change via a telnet session, a notification message is sent to the console port.

Logging to a serial console, you can send output to a printer or another host without telling a cracker where the output is going!

When properly configured, you can remove the video monitor (saving space). Even if you choose to keep the monitor around, you can save power and reduce heat in the room by turning the monitor off, and using the serial console for most operations.



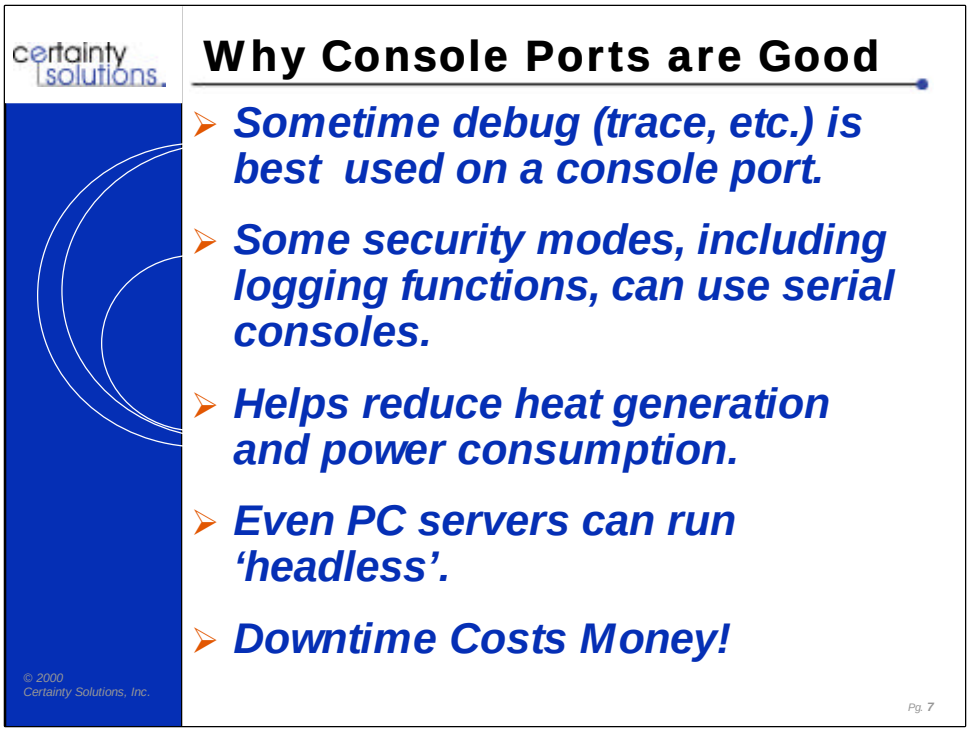
There are lots of good reasons why administrators connect to the console ports on their network. We'll discuss many of these benefits during this talk.

Once we convince you that connecting to console ports is a good idea, we'll advocate using terminal servers to extend your reach throughout your network, and then we'll show you how simple it can be to connect your device consoles to your terminal servers.

(Hint #1: using pre-wired adapters makes this job easy! Hint #2: Signal tracers help take the mystery out of debugging the physical link connectivity.)

Finally, we'll talk about how you can increase the value and benefits from your new console access by adding a client-server application between you and the terminal servers. In some cases, free software can add value to your terminal server investment.

Finally, we'll open the floor to your questions and answers, to try to fill in any blanks you may still have.

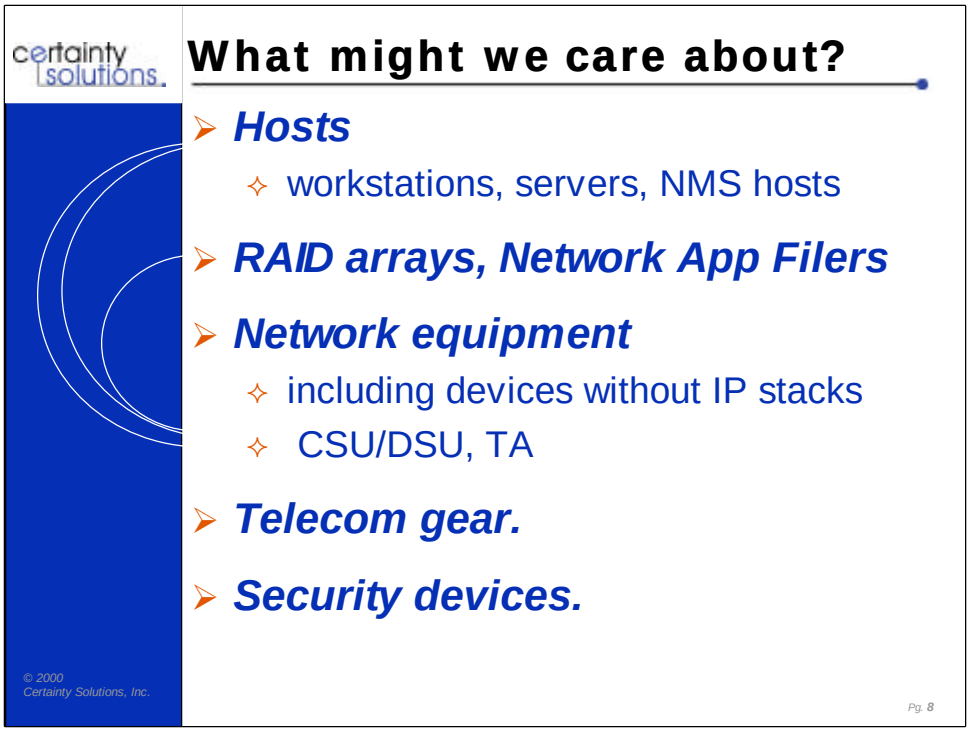


When you are trying to capture data from a network interface, you usually don't want to send the captured data results back over the same interface you are monitoring. If you only have a single network interface, the console port lets you manage the network interface without contributing to the traffic.

Sending messages to SYSLOG is good. Sending them out the console port as well is better. A cracker can figure out where your syslog data is going, but they can't tell where data goes once it leaves your serial port. (Even if they manage to modify the log files, they would be hard pressed to find your console data!)

CRT-type displays take up a lot of room,consume a lot of power, and generate a lot of heat. In large data centers, with dozens of machines, you cannot afford to have one display per machine, and it's difficult to justify a 1:4 or even 1:8 display-to-machine ratio.

There are even options to run your PC-class hardware without a display, using vendor-supplied BIOS, or an add-in EISA card.



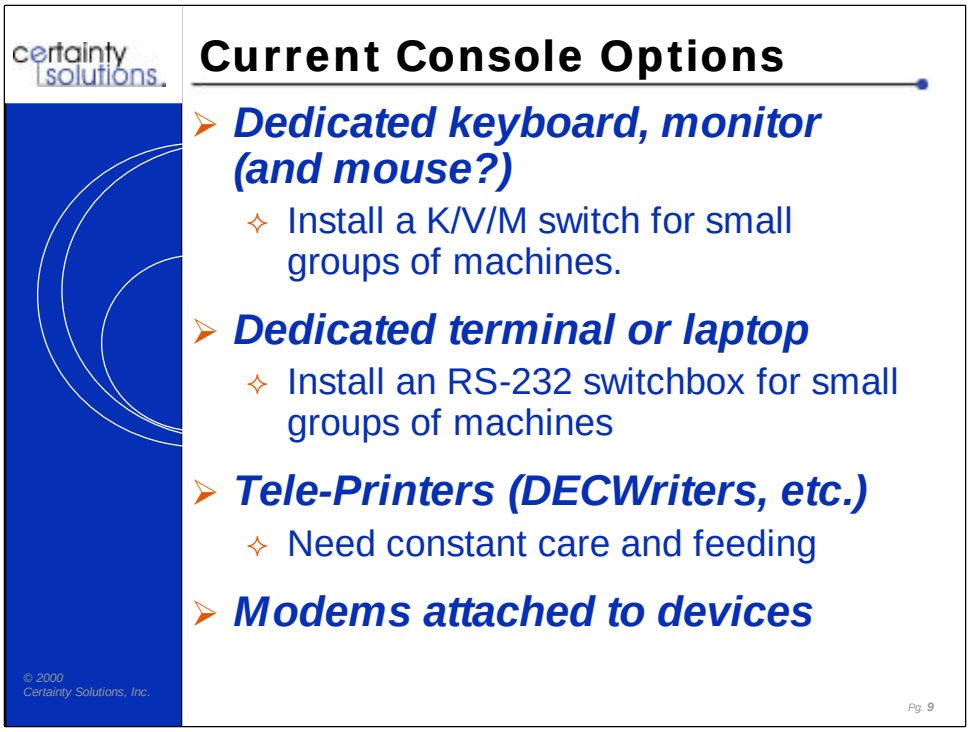
Console ports are not cheap, so we want to use them sparingly.

On the other hand, using a terminal server, you can easily get 16-32 ports in a single device, which may be more than you need initially, which then allows you to connect devices that you may not have considered valuable enough to connect in your initial plans.

Consider that any port that you connect to a terminal server can be logged using an additional logging server. What information do you think is important to capture? These ports should be connected.

If a device constantly repaint, or updates its screen, you may not want to capture the data to a log, but you may still want to connect it, so that you can get to that port from some other place.

Because the terminal server will likely be a capital investment, you may want to share the cost with other departments, and allow them to connect some of their devices to some of the ports.



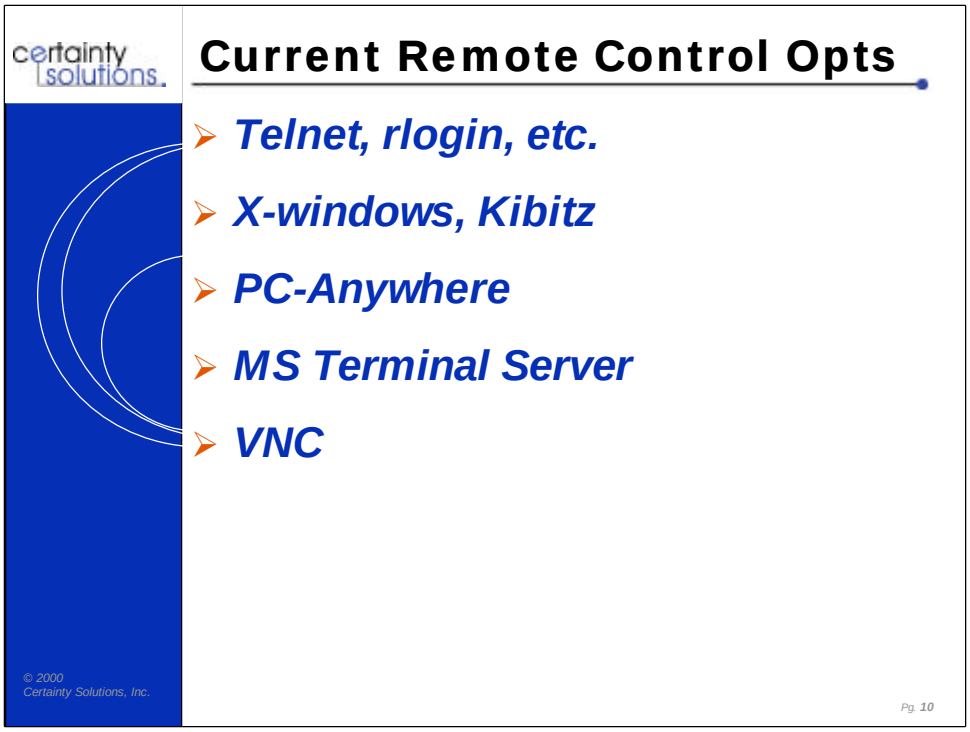
Dedicated monitors for each machine would be impractical.The cost for PC K/V/M ports are cheaper than serial consoles, but are more expensive for other platforms.

K/V/M systems have a limited distance, unless you use expensive technology to extend the monitors. (The cost is prohibitive for all but Network Operations Center applications.)

Dedicated terminals, or portable equipment will not scale. Only one person can use each terminal or laptop at a time, and they need to be physically close to the console being controlled. You may also have a problem with some console ports being set to different port settings than most other consoles (meaning your operators need to know how and when to change the settings for various devices.)

Dedicated teleprinters can jam, and need paper and ribbons changed regularly. You can't easily search for messages on hardcopy output.

Modems cost as much as a terminal server port, plus phone line.



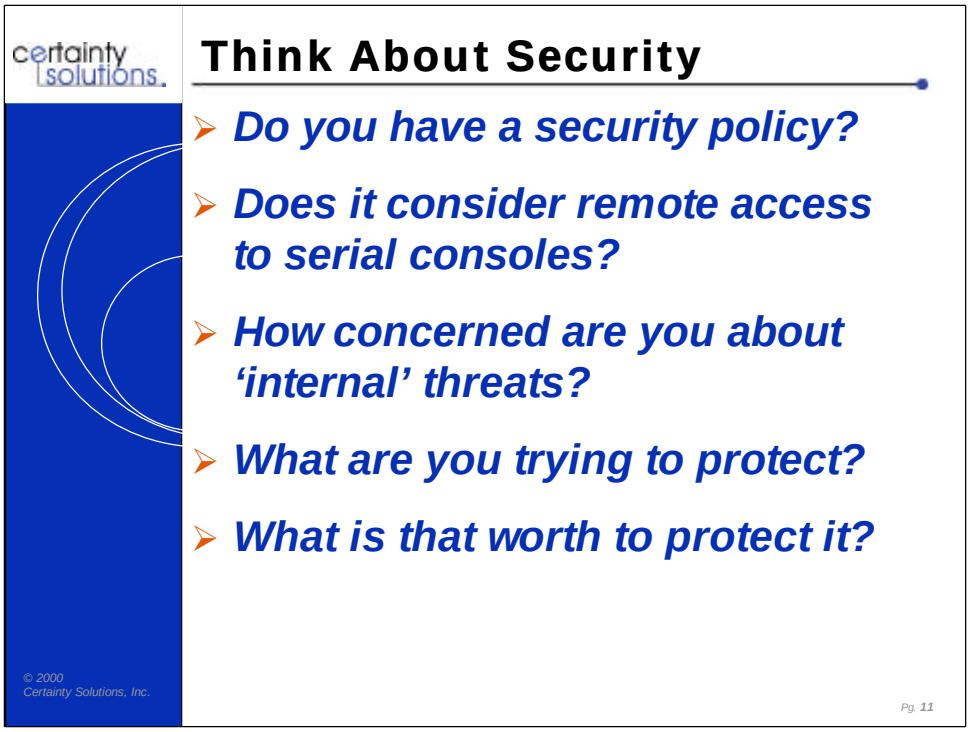
Telnet and rlogin do not provide an easy method to get current messages to the console port/window.

X-Windows sessions give you a bit more control of the type of session that you can run. You can even set some screens on the remote host to be displayed on your terminal. Kibitz requires the cooperation of someone at the far machine to help invoke the shared session(s).

Solutions for Windows OS machines also exist, but using them pushes you into more RAM on the machine in order for the software to run without slowing down other programs.

ALL of these options depend on the machine being booted, and the OS working properly.

When the machine is not booting, or operating as expected, THAT is when you need to be able to get on the physical console for the box, to see what is happening, and to hopefully be able to fix the problems.



We will discuss some security issues in this talk, but security is a touchy subject, and a few aspects of most cases are unique.

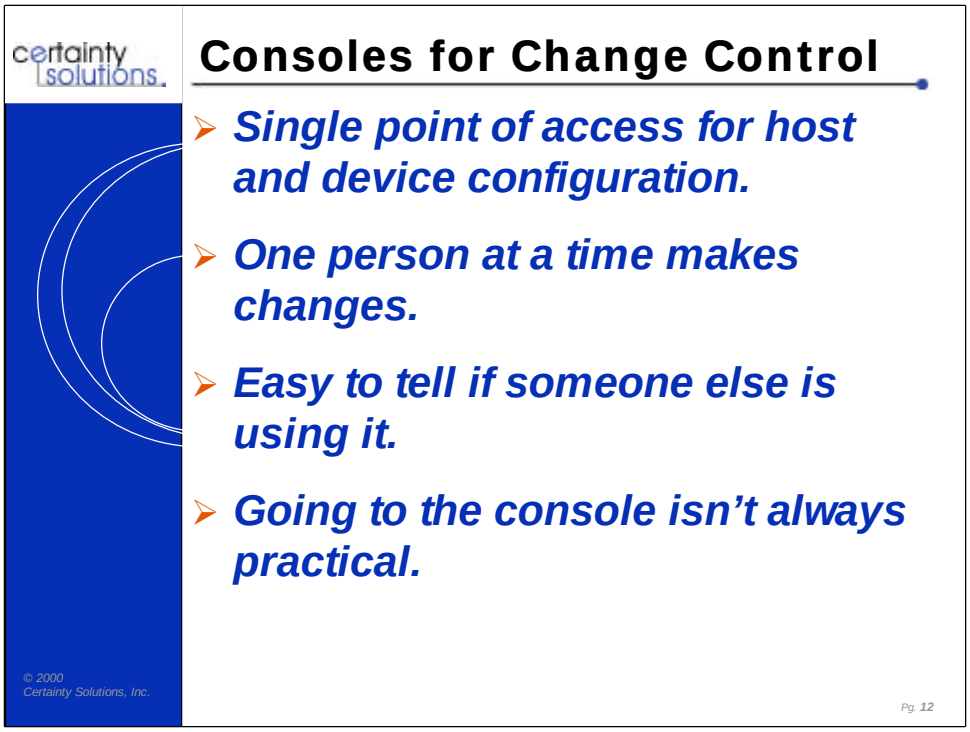
Due to this, we will discuss general points during the class, and the materials will give you some questions to think about and discuss.

During the conference, we will be holding a Birds of a Feather session, and we'd welcome any additional questions there, if you are comfortable asking them in that forum.

The biggest worry is whether you are concerned with the console traffic being monitored within your network. (Most companies use an jelly bean security model…hard on the outside, soft on the inside…meaning that they are not worried about folks on the inside sniffing the wires.

In a switched ethernet environment, it's harder for folks to see the packets to and from the terminal servers.

If you have a console server host, you should consider if it is worth making that host single-purpose, and limiting the login accounts



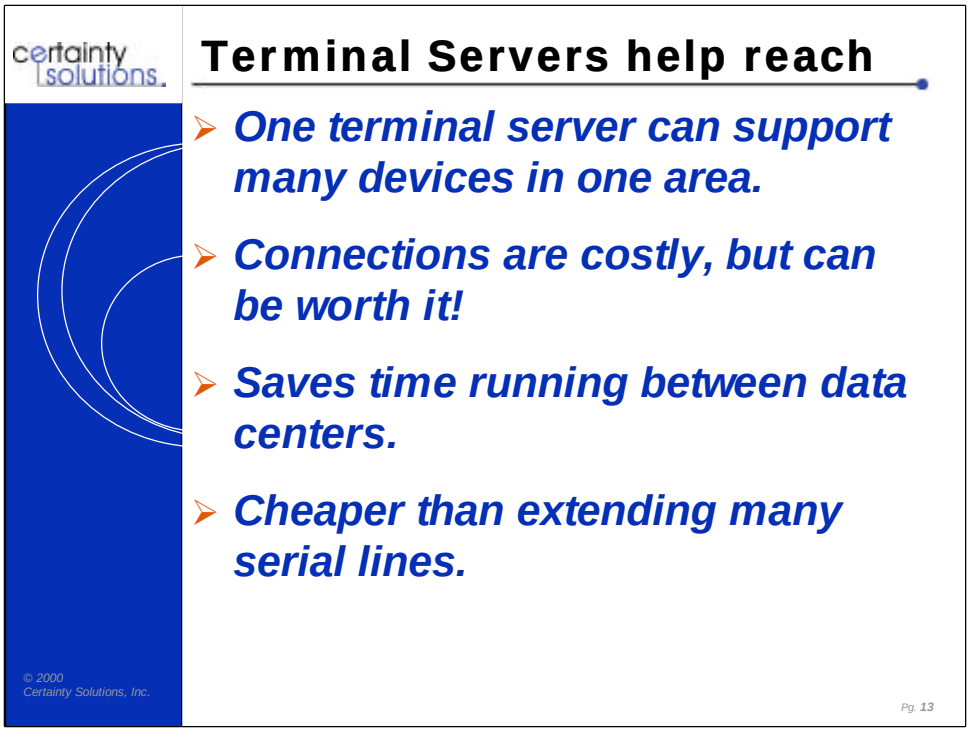
This is more of a process issue and a discipline issue. Even if you have remote access to the serial console, it is possible to have someone else making changes via a remote control session through the network.

When you have many administrators supporting any given device, you need to consider a change control process, to prevent one person from clobbering changes made by another person.

You can assign just one, or two, administrator(s) to manage each device, but that method doesn't scale well, and can sometimes suffer when an administrator goes on vacation, gets sick, has to serve on jury duty, etc.

Terminal servers will tell you if someone else is already attached to a particular port, but it may not tell you who is attached without the user taking extra steps to find out.

Does it snow where your office is? Does it get beastly hot in the summer? How long does it take to get to the different data centers and wiring closets that you support? How often do you visit each?

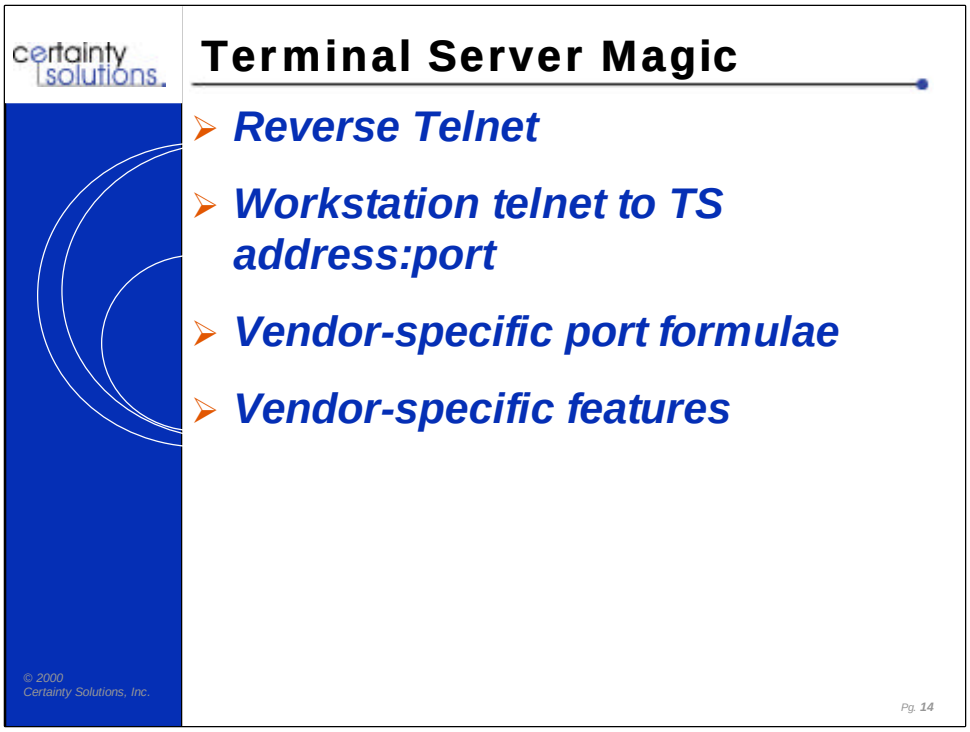


If you have a bunch of devices at a remote location, you likely have a network already in place for the devices. Usually these are hosts in a remote data center, but they could include network gear in wiring closets, or even a remote site with telemetry and control equipment. In these cases, the terminal server becomes one more addressed device on the network, and it can connect to dozens of devices at one time.

The cost for a 10/1000 switch port is currently cheaper than a serial port some of the expensive terminal servers. However, the costs for the serial ports are beginning to drop. (And they are still cheaper than gigabit ethernet ports!)

The value of deploying terminal servers is from the reduced downtime because technical staff can fix problems more quickly when they can get to the consoles of ailing devices more quickly.

You can use serial extenders, if you have plenty of fiber between the main and remote console locations, but that fiber has value to you as well, which affects your cost equation.



Originally, modems or ‘dumb terminals’ were connected to terminal servers, and users would telnet from the terminal server to other points around the network.

Today, most terminal servers allow you to open a socket-based connection to the IP address of the terminal server, but at a high TCP port number, to connect to a particular serial port. (This is known as Reverse-Telnet, because it was the reverse direction to the normal direction of attached terminals using telnet to reach hosts and servers around the network.)

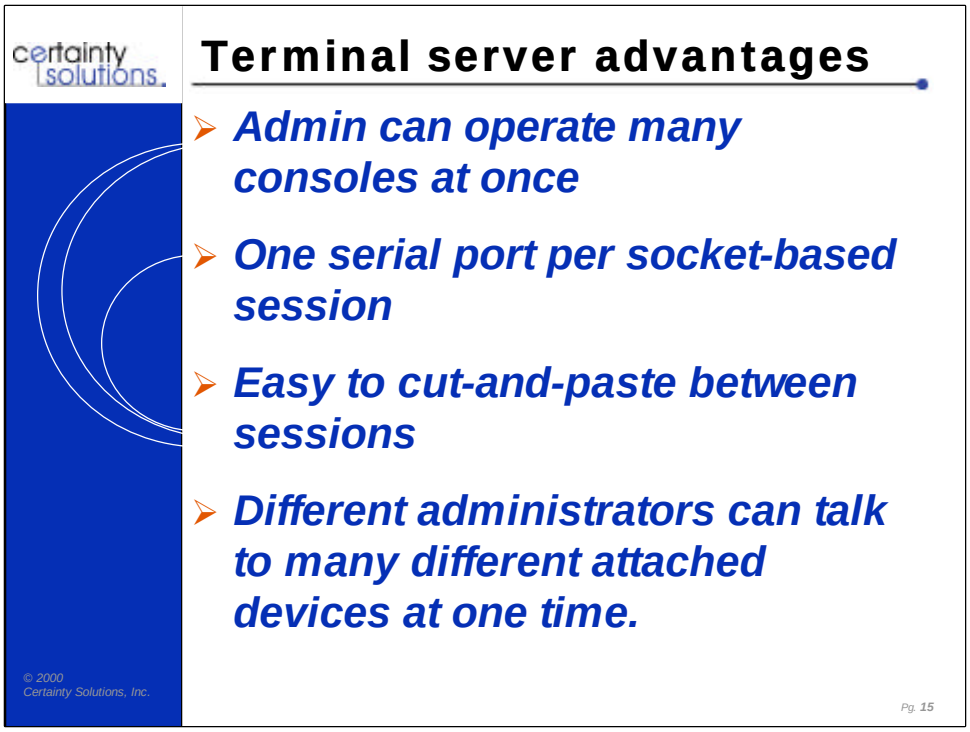
Some vendors allow only 7-bit sessions, while others provide the option for full 8-bit sessions, and even “non-escapable” sessions (where the attached device needs to drop the DCD or hardware handshake lead to disconnect your session).

The list below tells you the formulae to determine the TCP port number for two of the more popular terminal servers (where ‘n’ is the line (serial port) number you wish to connect).

Cisco: 2000 + n

Xyplex: 2000 + (100 * n)

IOLAN: 10000 + n

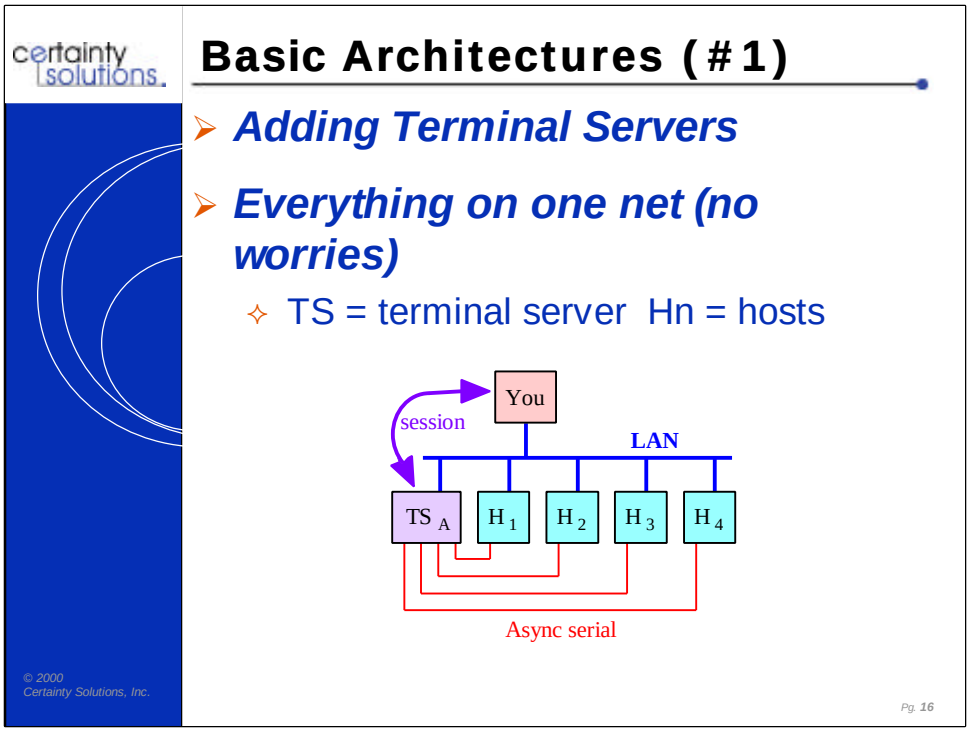


It's common to open many windows to different ports, so you can copy and paste between them.

Many administrators can work with different hosts simultaneously, from different workstations, rather then waiting in line to use a shared terminal or laptop.

Only one person can be connected to each port at any given time. This is normally a good thing, as it can prevent two administrators from making configuration changes at the same time, providing that you make it a policy to make changes from the serial console. If another person tries to connect to a busy port, the connection is refused, with a message to the user.

There is a downside to this, however, in the case that one administrator connects to a port, and then goes to lunch, or home. With the idle session still connected, nobody else can connect to that same port. (Someone with administrator privileges on the terminal server needs to log onto the terminal server, and reset that serial port to break the connection.)



This presumes that you are not worried about someone on your internal network sniffing the console sessions. In a small network, this provides the convenience factors (fewer display devices, no switchboxes, and different serial port speeds don't bother us).

The terminal server is connected to the serial consoles that we care about, with speeds set on the various terminal server ports to accommodate the speeds of the attached devices.

The administrator(s) can sit at their workstations, and connect to the serial ports that they want to control (using Reverse Telnet).

Remember that only one administrator can connect to any given serial port at once in this configuration. (Although administrators can connect to many different serial ports at one time.)

If nobody is connected to a serial port, and data coming in from the attached device is lost. If someone *is* connected and watching, you still only have as much logging as your scrollback buffer will provide for each session.

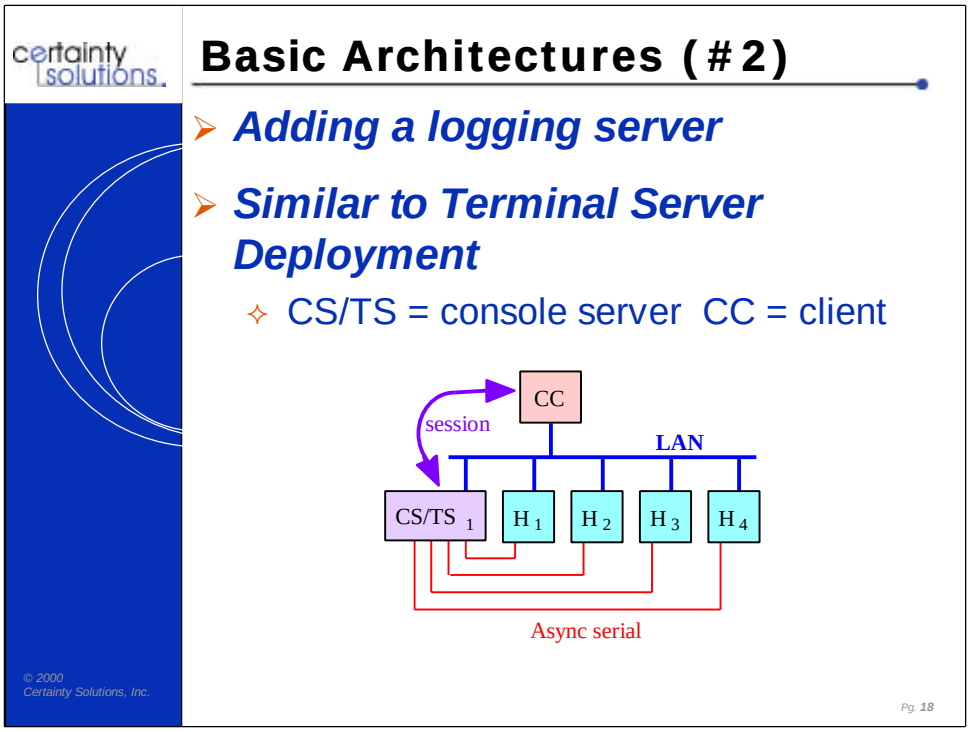
SSH has become a vital checkbox for many vendors to add to their terminal servers and other network devices, but the interoperability of the actual implementations, architectures, and failure modes are still far from good. You can get some single-vendor solutions running if you need them today, but you may not be able to integrate it later with other gear that you'll want or need.

Ask your vendor how to manage multiple accounts across multiple terminal servers. What happens when a user changes their password on one device…how does the change propagate to the other devices?

If you authenticate against a central authentication server (ala Cisco TACACS+, or RADIUS), what happens to user authentication if contact with the authentication server is unavailable? (This is an issue for terminal servers in remote offices if the WAN should fail...do you want to install and maintain multiple authentication servers in various offices? Can that management be automated?)

You will have more flexibility if you can implement a management network, to provide the level of security for physical access to the logging and client session data streams.

If you feel you need to add SSH today, you may be limiting your options. Of course, this will change over time, as customers push vendors to address the interoperability issues.



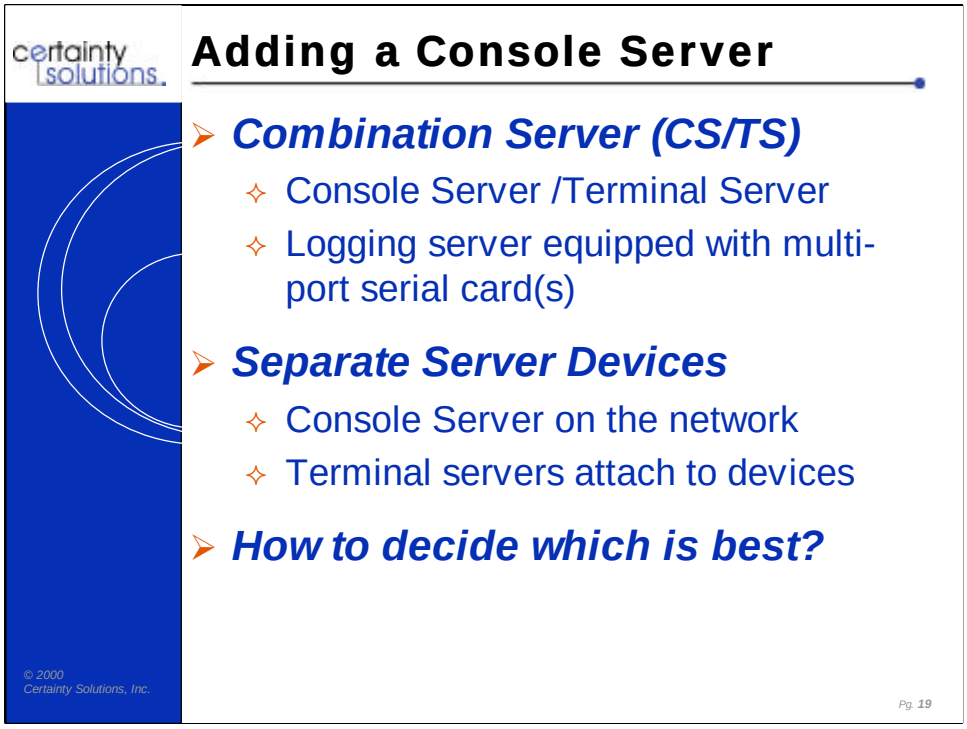
In this configuration, we still don't worry about internal security.

In this case, the Console Server (CS) has multiple serial ports installed, connected to the various consoles that you care about.

Any data coming in from each attached device is appended to a log file for that specific device. (As an administrator, you may need to watch the size of the log files, and rotate them occasionally.)

In our example, you would use a client (CC) to connect to the console server (CS), and attach to the console logging session that you want to talk to. Anything you type on the client goes to the serial console you are talking to, while anything coming back from that console is logged, and then sent to the client(s) attached to that session.The client software, and/or the server software, watch the data stream from the user, looking for meta-characters coming from the client, to allow for session control.

Some console servers allow more than one client to attach to a single session at once. All clients see the data coming from the attached device, but only one client can type to the session. (Conserver behaves in this manner.)



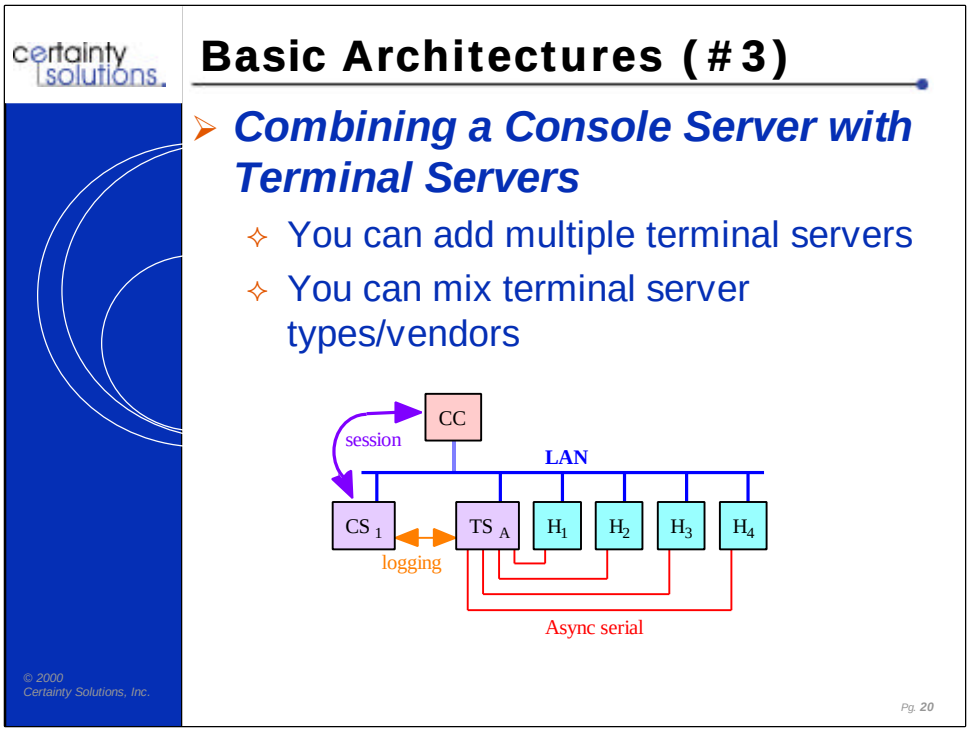
Some console server applications allow you to add some multi-port serial interface cards, and attach serial consoles directly to the console server. In the preceding illustration, we showed this as “CS/TS”, because the Console Server is also performing the serial attachment duties of the Terminal Server.

Conserver can support this mode, although we don’t normally recommend it, except in small, non-production environments, because you are usually going to be limited by how many serial ports can be hosted on the CPU. (This is a function of the number of spare slots, and the number of poets per card. You wouldn’t usually see more than 96 ports on a single server in this mode.)

A secondary concern, in our mind, is that the serial ports will normally send a serial BREAK signal when you power-cycle the CPU. In a Sun environment, that signal on the console will halt the server, and you’ll have to wait until the CPU boots before you can get back on each console individually and type “go”.

The main advantage of this mode is that the logging traffic is kept within the computer, rather than on the network between the terminal server and the console server. (The network load is only partially reduced, since the client sessions still traverse the network to get to the console server.)

As a result, the security issues of sniffing the logging traffic is also reduced, but the client sessions are still exposed, unless the client can use SSL or SSH for the connection to the console server.



This is the most versatile configuration, in our opinion. Again, security on the internal network is not a concern in this model.

You can get more ports on a terminal server (TS)than you can practically add to a console server (CS) host. And, you can add multiple terminal servers around the network. All of the logging traffic travels from the terminal server to the console server over the same LAN as the host communications and everything else.

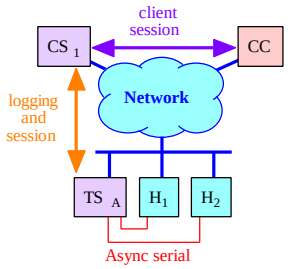
If you are using a switched ethernet network, you are already gaining some practical protection from folks sniffing the console sessions.

The logging traffic from the terminal servers are fairly light, even from a large number of busy hosts. Your mileage may vary, of course, as you increase the amount of data that you are logging, and the number of terminal servers that are added around the network. (This traffic is normally contained with the Console Server when you are using built-in serial ports only, instead of terminal servers.)

certainty solutions.

Basic Architectures (#4)

- ***Client can access from anywhere on your network***
  - Terminal Server is located physically near consoles
  - Console Server doesn't have to be close to monitored consoles.



The console client (CC) can be anywhere on your network.

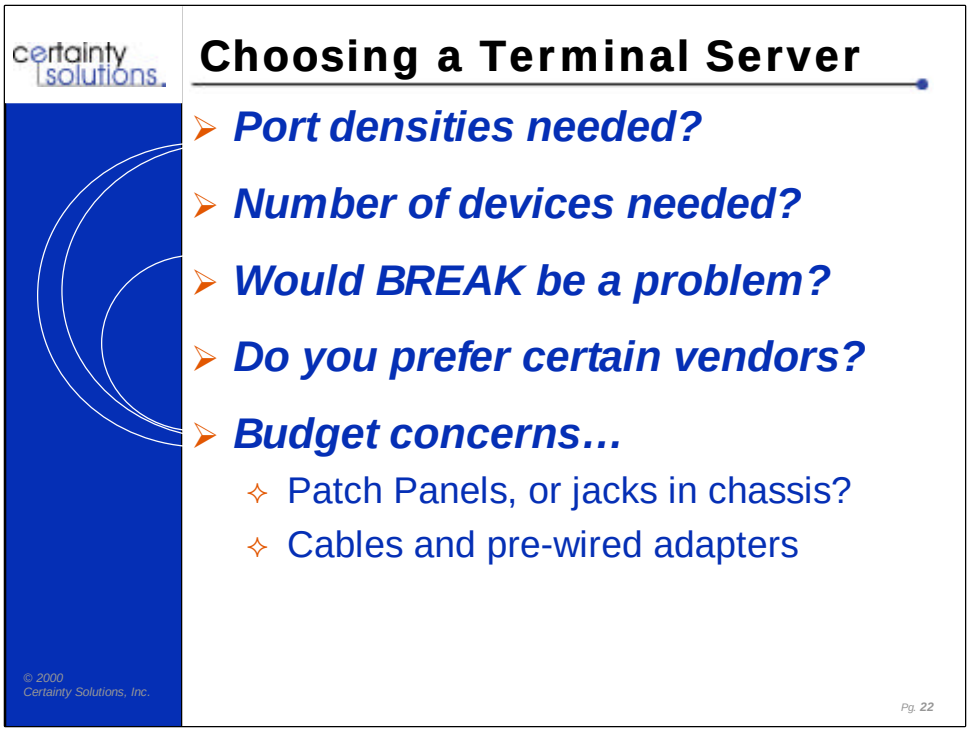
+ dialed in from home (through your dialup authentication)

+ across the network; on another floor; in another building

+ across the WAN, from another location

If you need to access across the Internet, we suggest that you use SSH to get to your home network, and then use a client on a machine on your network, rather than using a client on your laptop from the terminal room, for (we hope) perfectly obvious reasons.

Asynchronous serial connections do have length limitations, so you need to have your terminal servers (TS) close to the devices that you want to control remotely. (If you are using serial cards in your Console Server (CS), then that also needs to live near the equipment that will be attached to it.)

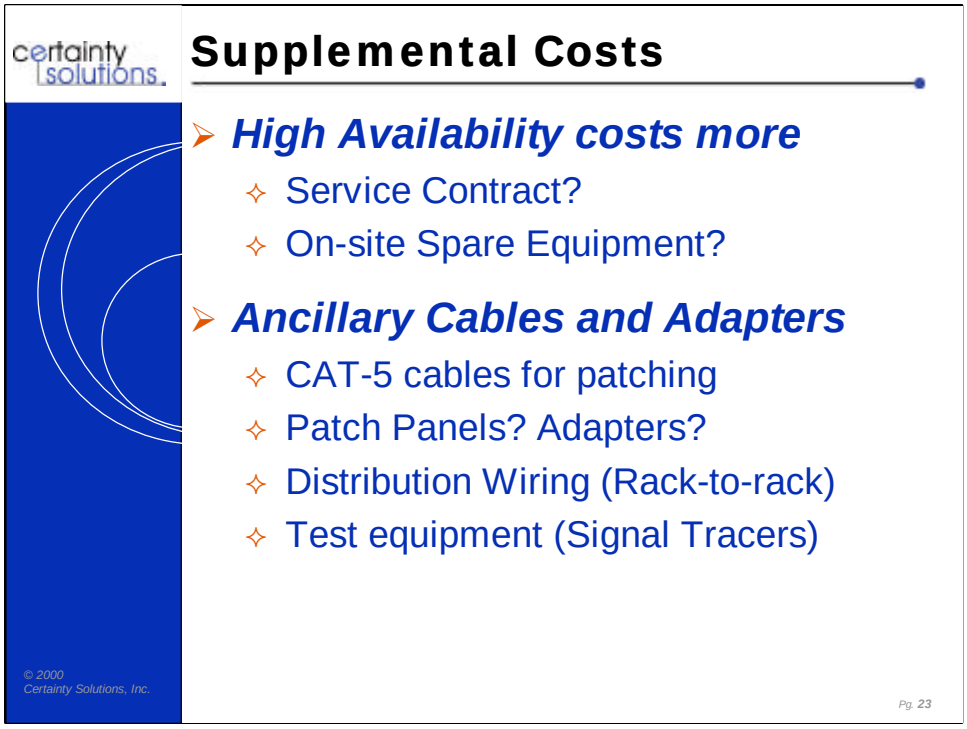


Choose your port densities with cable lengths in mind, since you want to keep your serial cable lengths shorter than 50 feet (16 meters). While most devices can drive a signal farther, some cannot, and those shorter lengths will prove to be limiting during your deployment phase. The cable length and the number of devices will help you decide how many terminal servers you should have in a given data center, and where they should be placed.

If you are serving large numbers of ports, you should consider purchasing multiple terminal servers, and distributing them around the data center, rather than having a single, large server. This becomes your classic “single point of failure” discussion.

Some devices are sensitive to the serial BREAK signal. (Test equipment, modems, and Sun hosts are the most sensitive.) If the serial BREAK would be a problem for some of your devices, your choice of vendors will be smaller.

Terminal servers can be expensive, but the payback comes from minimized downtime. You can add more value to the capital investment by adding free logging software. In addition, you may be able to deploy a modular terminal server, starting with fewer ports now, but the ability to add modules later, thereby spreading the cost over a couple of fiscal quarters (maybe even bridging Fiscal Years).



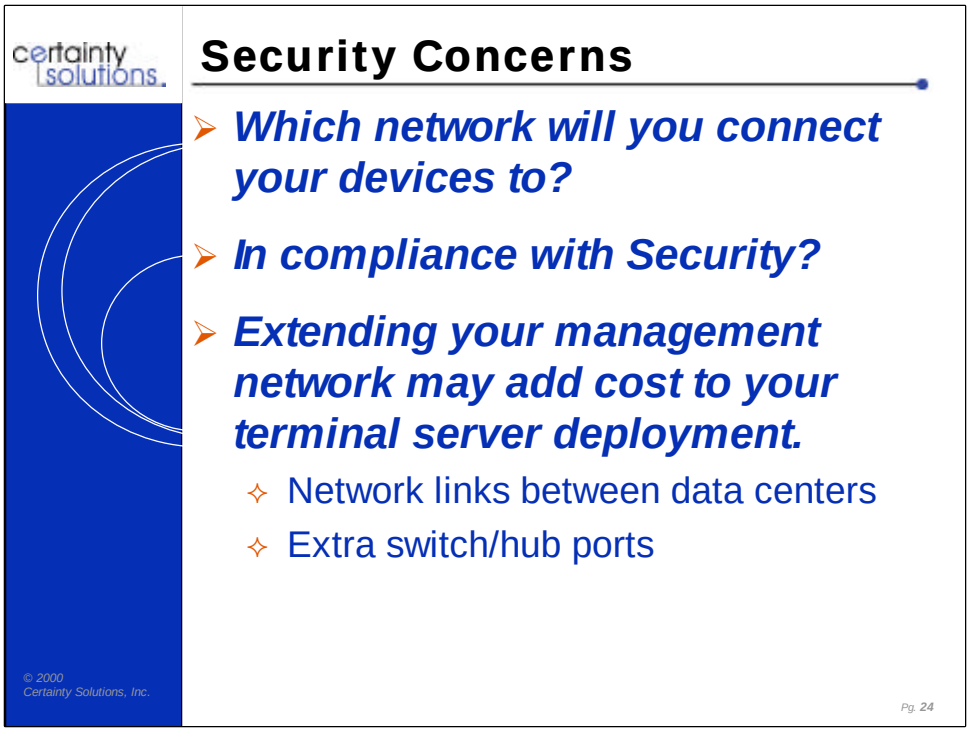
When you come to depend on some infrastructure equipment, you want to ensure that a hardware failure does not cripple that infrastructure.

If you are ordering a large number of smaller servers, you can probably afford to buy an additional unit to keep as a “hot spare”. If a module or component fails, you can swap a working part from your spare, and then pursue the RMA for the filed component.

When you are using larger servers, that spares cost is higher, and the high cost may be too much to bear. Your options can include paying for a higher level of hardware support from the vendor, that can include short turn-around time for RMA parts, and/or advance swap components. The cost of this higher service contract will be a recurring cost, year after year, but it isn’t the single, big cost for the spare unit.

Remember that the spare unit does not need to be fully stocked! If you have a terminal server with multiple slots, you only need one or two modules for spares use, rather than filling all of the slots.

When you are purchasing your terminal servers, remember to order sufficient stock of RJ-45 adapters, and and needed patch cables (and patch panels?) that you will need to deploy the terminal servers. You should also have an adapter kit, with one of each standard adapter for that terminal server, as well as passive signal tracers for troubleshooting.

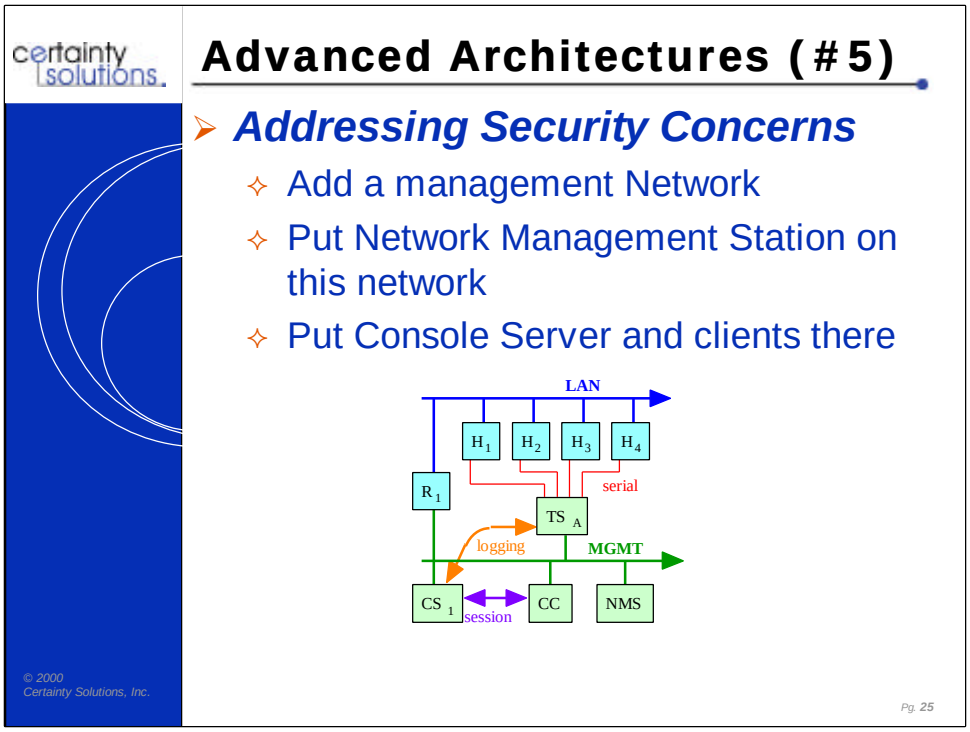


Determine which network(s) will host the terminal servers.

Is your deployment in compliance with any existing security policies? (You may not be able to connect to certain networks, or you may not be allowed to extend those networks into any other non-secure areas.

You may plan to put the terminal servers on the management network for security reasons. But find out if that network is currently in the area(s) where you want to put the terminal servers. (You may need to use media converters to let you use spare fiber between data centers, or add memory to routers to allow you to use encrypted tunnels, etc.)

Do you have enough open network connections in those places, on those networks, reserved for your terminal servers (or do you need to order more hubs or switches)?

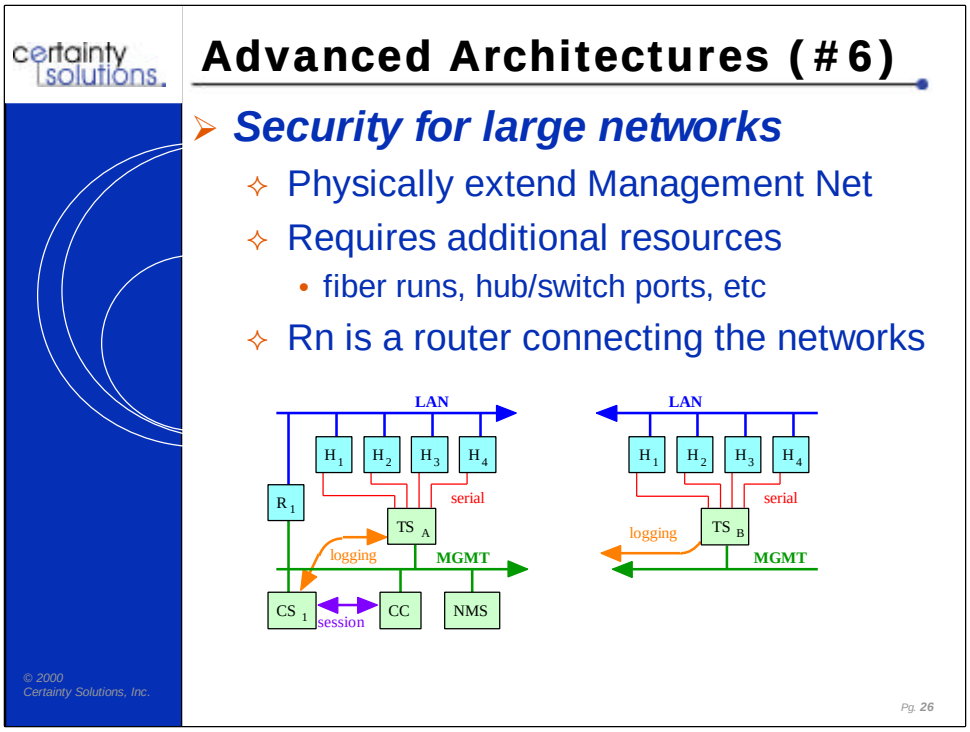


If you are concerned about someone sniffing the client-to-server connections, or the logging streams, then you probably already have a control/management network in place, where your monitoring and control activities take place.

In this model, the console server (CS), and the client(s) all live on the management network along with the terminal server(s), so that the client sessions, and the logging activity, all happen on the management network.

A good practice is to ensure that your management network is connected using switches, rather than hubs…

Few console servers use SSL or SSH for the client-to-server connections, so these sessions travel in clear text. For this reason, if you are located outside of the management (or security) perimeter, you should consider making an SSH connection to a client host that is on the management net, and then making the client connection from there.

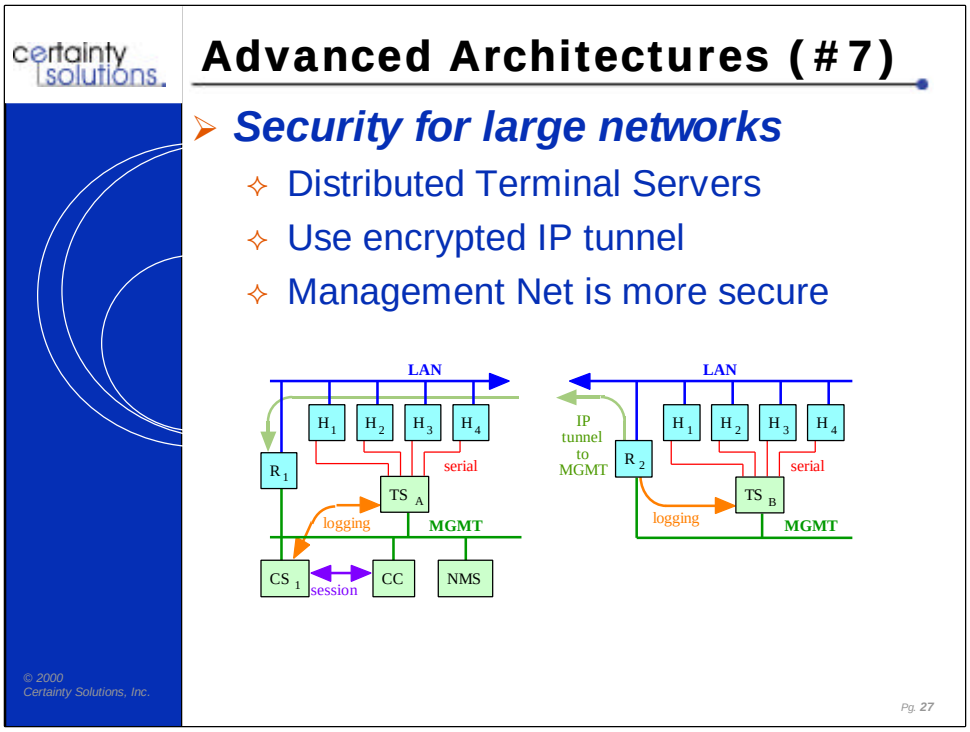


If you have a need to support multiple data centers, you may want to extend your management network to those other data centers. (Remember that serial lines have fairly limited distances, and it is easier to extend an ethernet network than to extend serial lines.)

If you have additional fiber pairs between the data centers, you can do this with media converters.

Adding capacity for parallel networks between data centers is an expensive project. The commitments for your fiber may extend to your security needs, telecom systems, and more.

Even if you have enough capacity today, your needs may change in the future. Since you are extending a management network, you should try to think into the future, and plan for a connection that will stand the test of time.



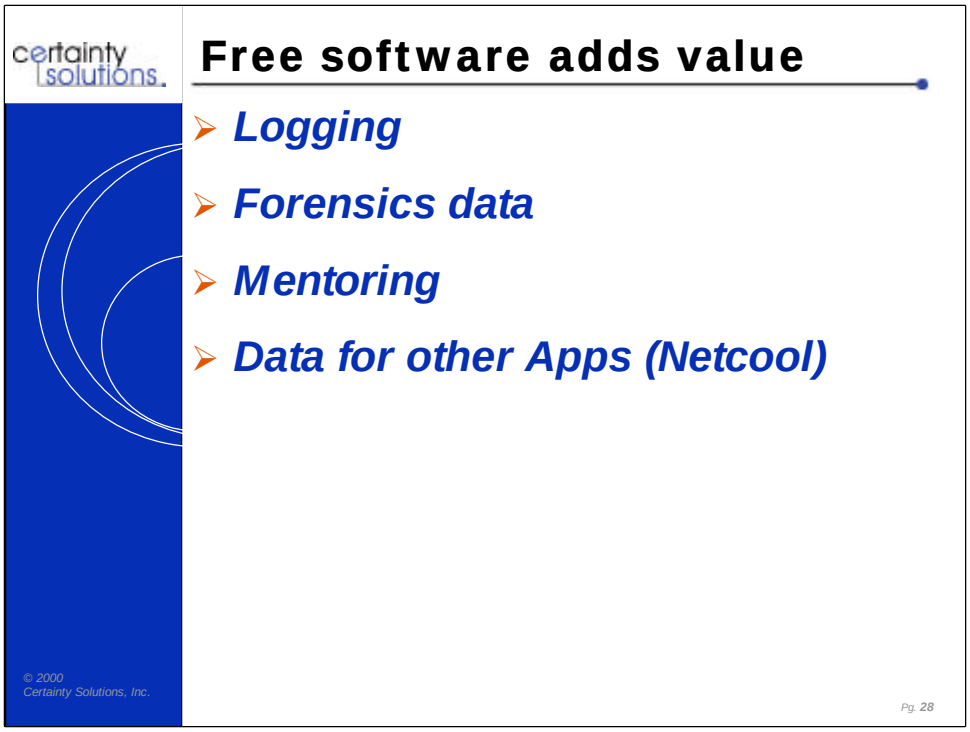
In an environment where network connection paths are limited, you may want to consider adding an encrypted tunnel from one of your management networks to another.

This may cost you some more memory for the routers, and it may drive you to increase the bandwidth if you are communicating over a WAN link.

It may even push you into a larger router with more capacity (or a hardware compression solution).

When considering this plan, try to think into the future. If you are ordering more RAM, more bandwidth, or a more powerful router, will it be capable of the growing needs of your network.

Changes and upgrades often mean downtime, and you may as well try to do one change instead of many.

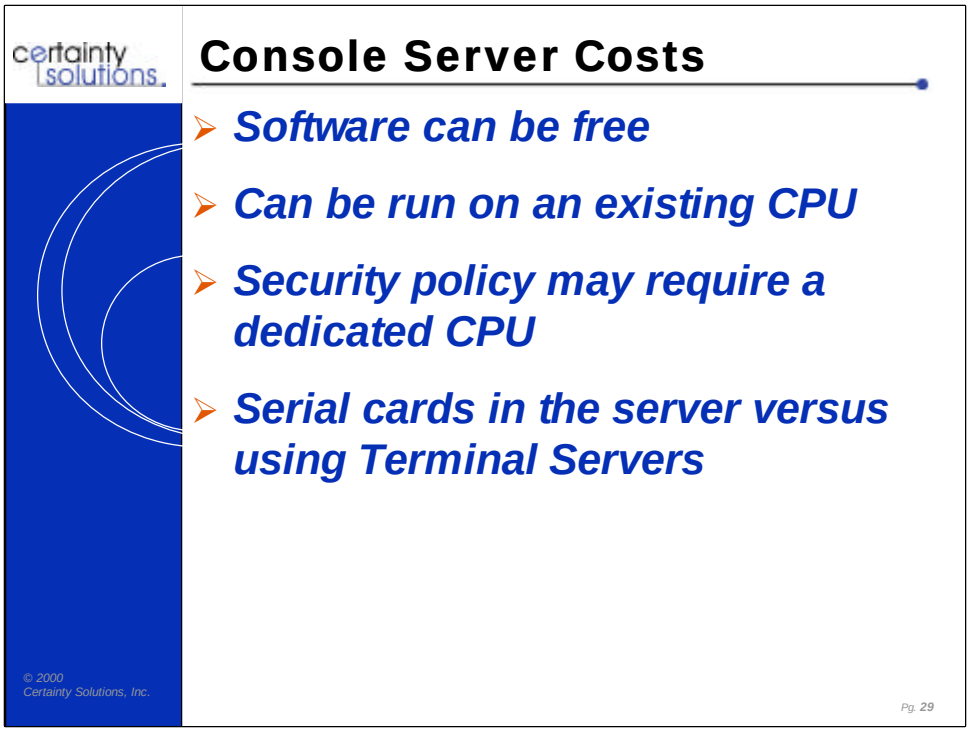


Terminal servers give you easy access to the serial ports around your networks. A logging console server will open a reverse telnet session to each of the serial ports that you care about, and log any data sent by the attached devices, even if nobody is connected.

Logs provide good forensic data after a crash, break-in, or other problems, especially if nobody was looking when it occurred.

A good console server allows many users to watch a session at the same time,while allowing only one session to write, allowing junior and senior level staff to work together, creating a great mentoring environment.

Each device being logged has a separate log file, which can be easily searched (if you have proper access privileges). Scripts can do this automatically. Some applications (SystemEDGE, from http://www.empire.com) can watch logs for specific messages, and then send alerts or alarms to administrators, or even to other applications, such as network management workstations.



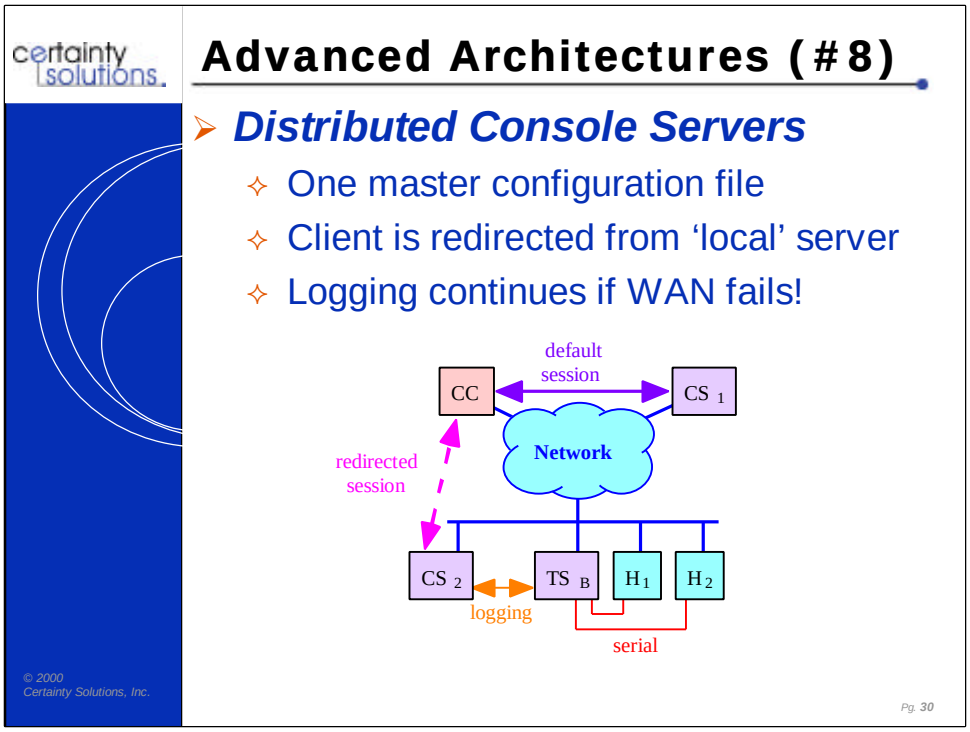
There are many free solutions available, depending on the features that you want. There are at least two commercial applications available to you. (Aurora Technologies, Computer Associates) Our favorite free application is Conserver, available at http://www.conserver.com/

You can often use free software on an existing server, as the needs of the software themselves do not require much memory or CPU.

However, shared machines often mean security vulnerabilities in the other applications, which could allow unauthorized folks to see the logs (or tamper with them). A shared system often means more users that have legitimate access to the machine, which can be another vulnerability.

If security and integrity of the logs is a concern, you should consider purchasing a dedicated machine, installing a hardened OS, and limiting the number of folks who have login access to the CPU.

Adding multi-port serial cards to a machine increases power consumption, heat output, and (depending on the card) can also increase CPU load.

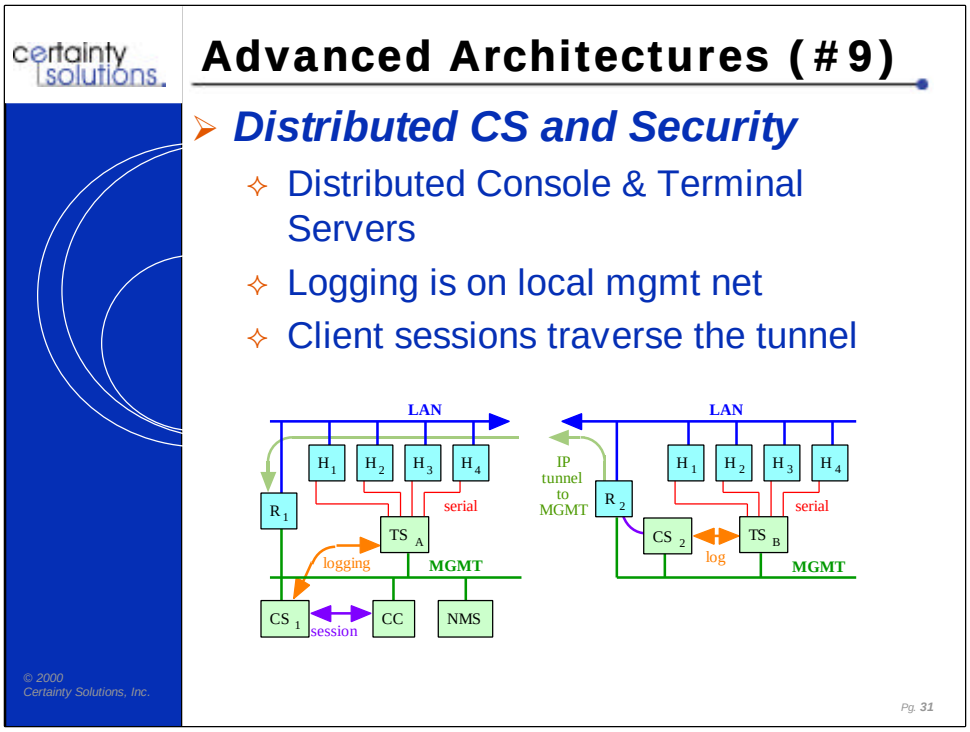


In the case of Conserver, you can indicate in the configuration file which of a number of distributed conserver hosts is maintaining the logs for any particular device, as well as indicating which remote terminal server is directly connected to the device.

You can then push this master configuration file to all of the distributed conserver hosts.

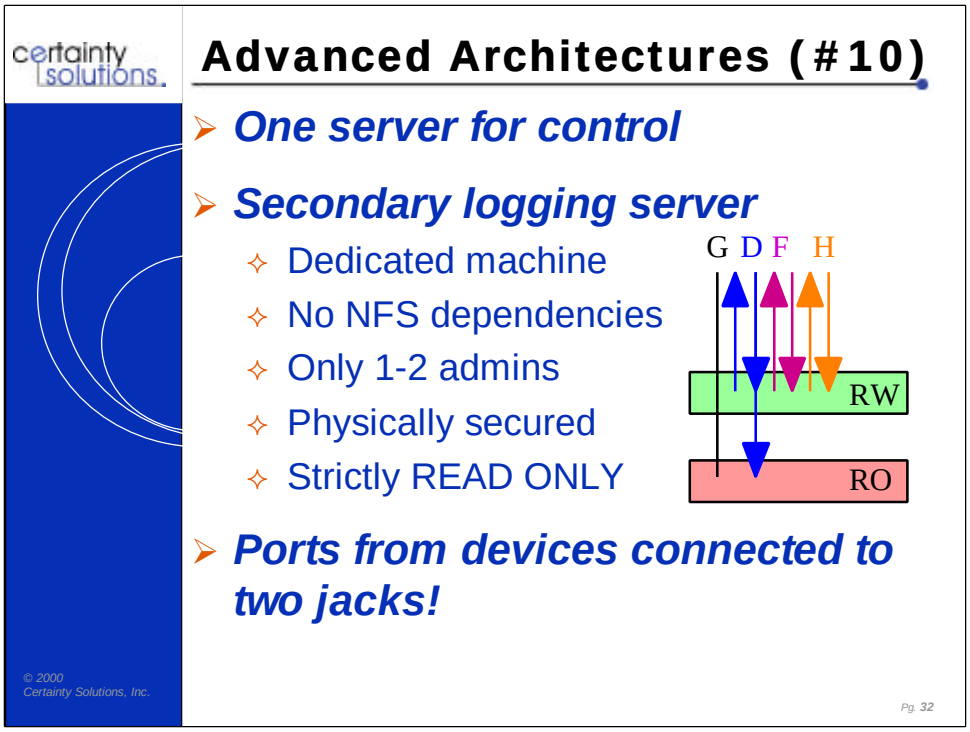
Each client should try to connect with the conserver host that is closest to the client for all connections. (You can do this via hostnames, or even with a local host file entry.)

If a client (CC) connects to his local conserver host (CS1), but is trying to reach a device that is connected to another conserver host (CS2), the client is redirected to the other conserver host. The client session is then connected with the session of the desired device.



We can extend the idea of distributed console server hosts in our earlier security models. In this case, all of the logging traffic is kept on the remote management network, as peer-to-peer traffic, and only the client sessions from other sites is carried over the tunnel.

It is possible to have console clients in the remote office, even though it was not shown on the diagram above. With a single configuration file, a client on the remote MGMT network would connect to the console server on the same local network, and it could be redirected to other console server hosts if the client wanted to connect to console devices hosted on other networks.



Legend: G = signal ground, D = tx and rx data, F = hardware flow control leads, and H = hardware handshaking leads.

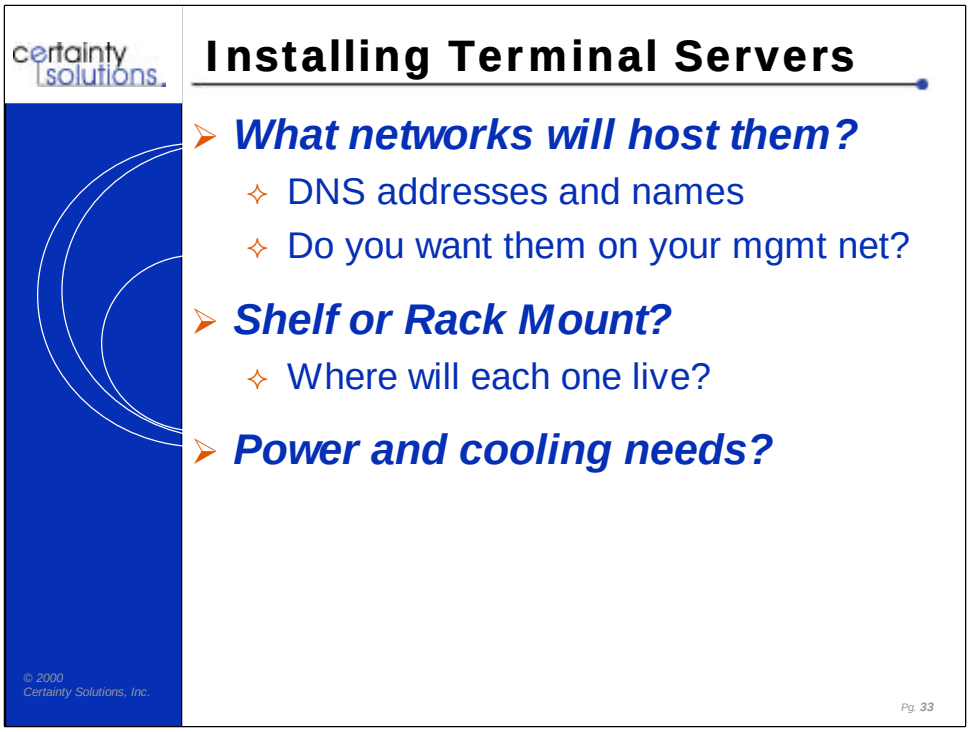
Basically, only the signal ground and the data coming from the attached console is delivered to the second port. There is no way to control the attached device from this second port, but you do get a second logging server, to provide a method for you to validate logs on the primary server.

This can be applied in an environment where your logs may be audited, and you need a way to reduce the chances of someone being able to tamper with logs.

One console server is deployed, and is used by console clients. This server can be managed across the network, and probably is a dedicated device, with a minimum number of login accounts.

A second console server is added, which is also dedicated to this task, but is also designed to be a stand-alone server, with no file dependencies to other hosts on the network. You would have a local monitor and keyboard, but this is all in a physically secure location, to prevent access by unauthorized folks.

The console wiring from the attached devices is brought back to the secure location, and connected to TWO sets of jacks (for each attached device). All wires are connected to the RW jack, while only signal ground and receive data are connected to the read-only jack. The RO jacks connect to the second server.

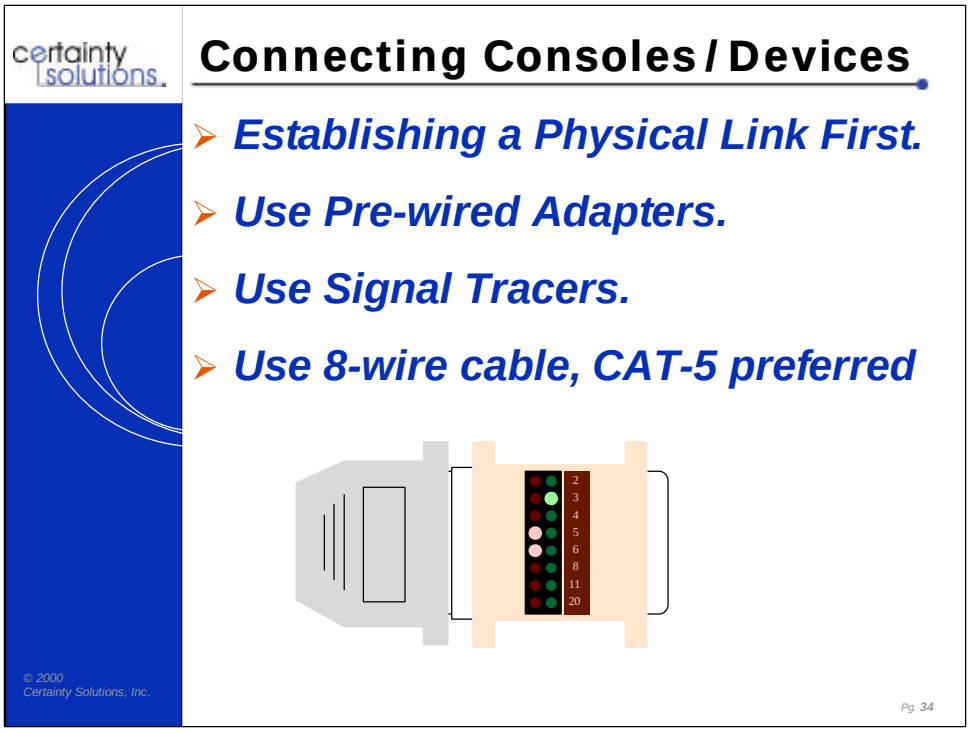


Before your terminal servers arrive, you should know where they will physically live (so you will know if you can order a rack-mount version, or if you have to buy shelves). You should have decided what network(s) they will be connected to.

With that information, you can pre-assign IP addresses, and DNS names for the terminal servers.

Before the terminal servers arrive, mark their new locations, so that other administrators don't try to 'squat on your land'. This can mean installing and marking the shelves, or marking the rails where rack-mount gear will be placed. (You should try to reserve your power and network connections as well.) You don't want to go to the data center on install day and find another machine already connected and running…which may delay your installation.

Think about cooling for the new equipment. Do you need to advise someone that new equipment will be installed in the data center? Is there adequate power and cooling capacity in the room?



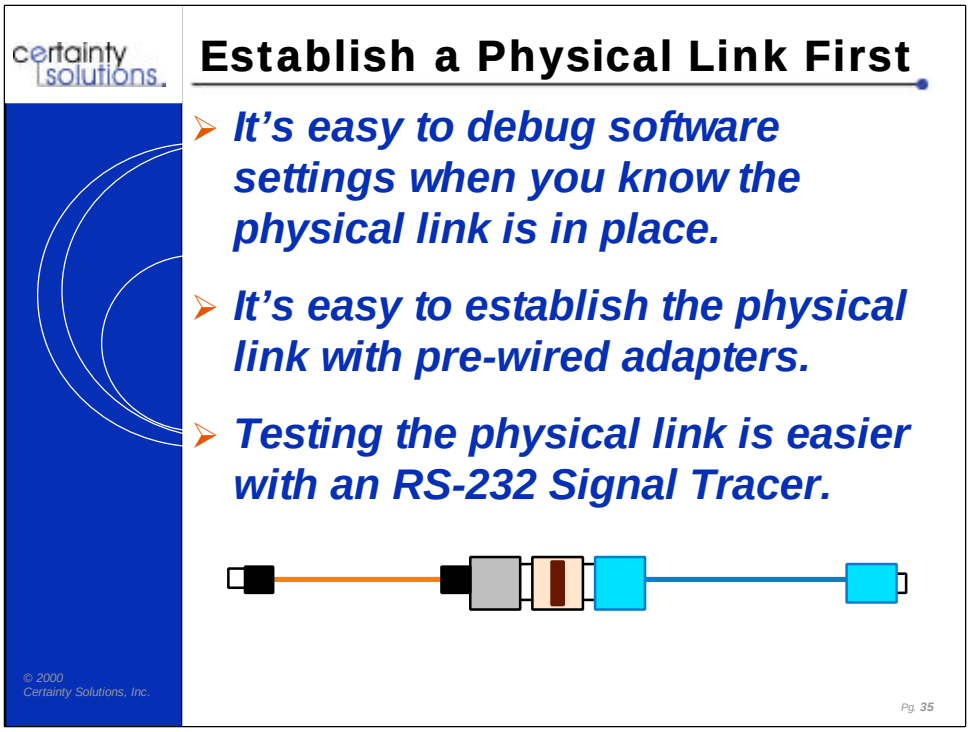
Once your terminal server is connected, your next task is to connect various serial consoles to the terminal server ports.

Pre-wired adapters will save you time and trouble, and we highly recommend using them. (The biggest benefits: consistently wired and labeled.)

If you are using different wiring on the RJ-45 connector for different vendors terminal servers, you should order your parts with distinctive colors, so you will know at a glance if you have a properly wired adapter for a specific vendors terminal server.

Passive signal tracers are the best tool for determining if you have the correct adapter, and if you have data passing between devices. We strongly recommend using them.

We'll show you how to use pre-wired adapters and signal tracer in the slides ahead.

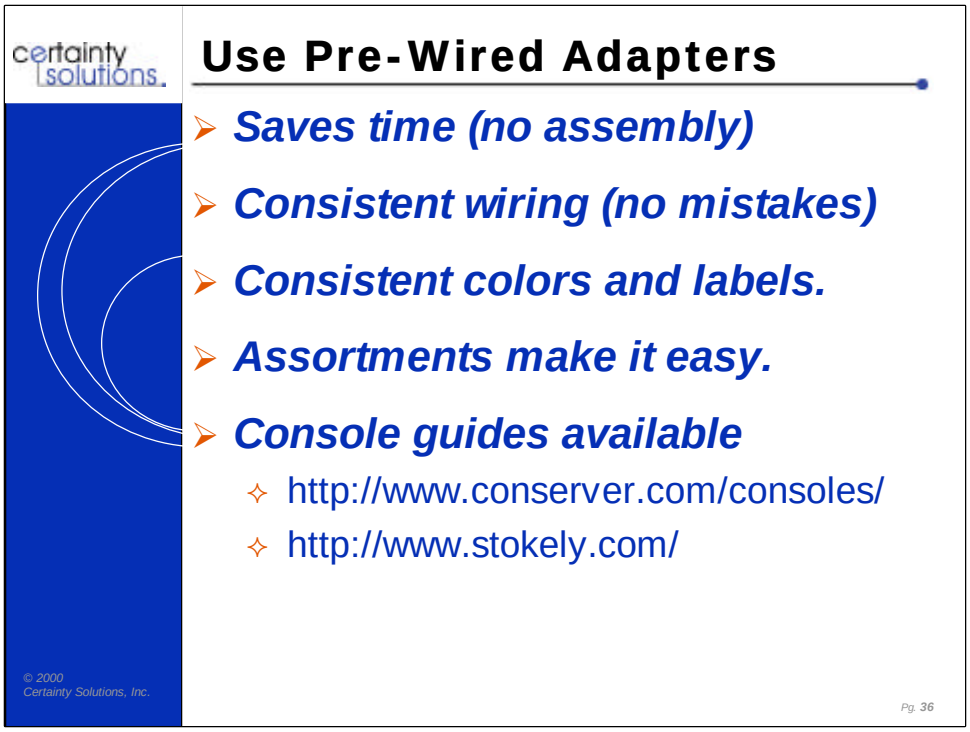


You could use an asynchronous serial protocol analyzer to determine whether your software settings are correct. Or, you could use a small, passive device to help you get the hardware signals in place.

I find it is easier to get the hardware signals connected, and then work on the software settings. Using a signal tracer or breakout box will help you determine if the wiring is correct.

Using pre-wired adapters helps ensure that all of the signal leads are properly connected. Then you can hack with the software settings with confidence that your hardware connection is ready to pass data and hardware signaling.

You can find my **Console Information Guides** for a number of vendors at http://www.certaintysolutions.com/consoles/ which include reference pages for Cisco, Xyplex, IOLAN, and Xylogics/Annex/Bay/Nortel servers.



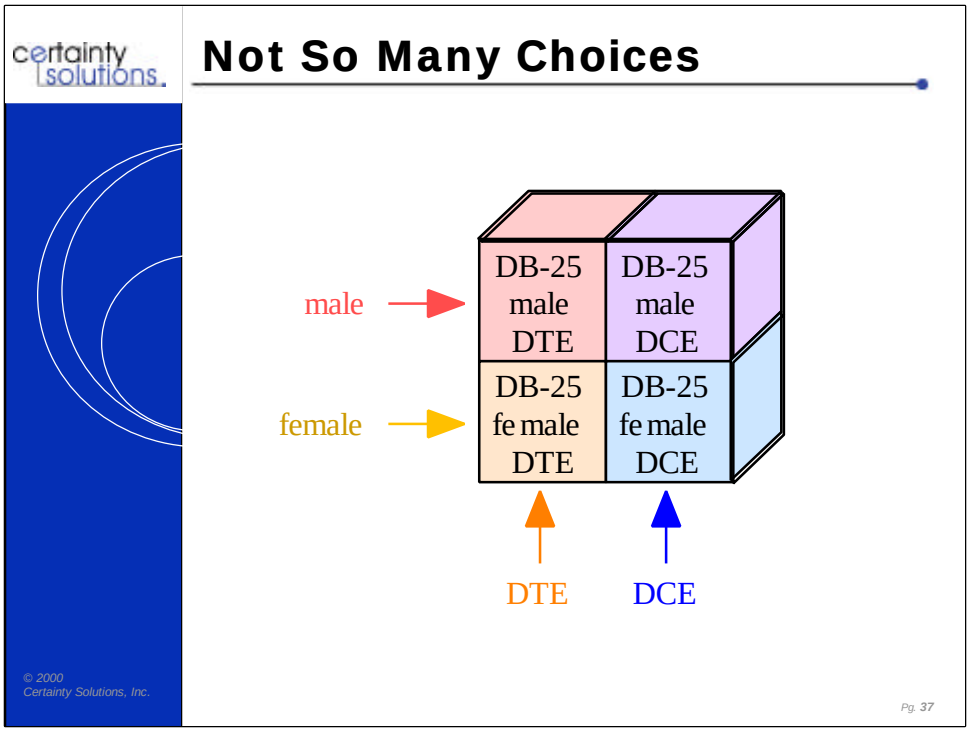
Most of the terminal server vendors will supply a small assortment of RJ-45 adapters with their devices. Other vendors make these adapters available as an option. Many of the vendors also show wiring examples for a few adapters in their manuals. (If you make them yourself, you should always label the adapters!

Making connections quickly is easier when you have a good selection of reliable adapters. (More on this later…)

Pre-wired adapters can also be ordered in different shell colors. If you have more than one RJ-45 wiring format in your shop, it's a good idea to use different shell colors to separate the different RJ-45 wiring formats. Maybe use black for Cisco, green for IOLAN, blue for Xyplex, red for Annex/Bay/Nortel.

In a co-location facility, you may have a variety of terminal servers deployed, each with a different RJ-45 wiring format. In a large installation, you may have to install "strategic partner" equipment next to your own. In these cases, having different colors for the shells makes it easier to see when a device is plugged into the wrong terminal server.

Americable, Inc. carries a wide range of adapters for a variety of terminal server RJ-45 wiring formats. More information is available at http ://www.certaintysolutions.com/consoles/ under Console Adapters Kits.



For each connector type, you have two choices for gender.

For each gender, you have two other choices, DTE and DCE.

We'll look at on of the most common connectors in use for RS-232 Serial communications today, the DB-25. In the illustration above, you have four possible combinations;

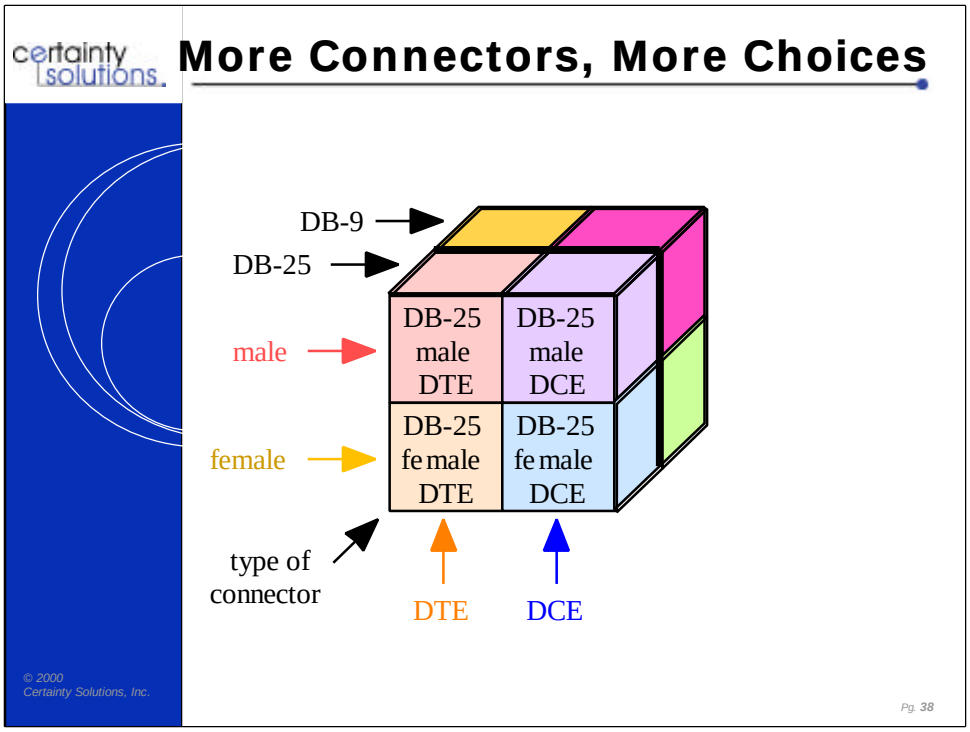
female-DCE

female-DTE

male-DCE

male-DTE

Now, only four combinations isn't that bad, is it?



So, for each connector type you use (DB-25, DB-9, etc.), you have only four choices.

female-DCE

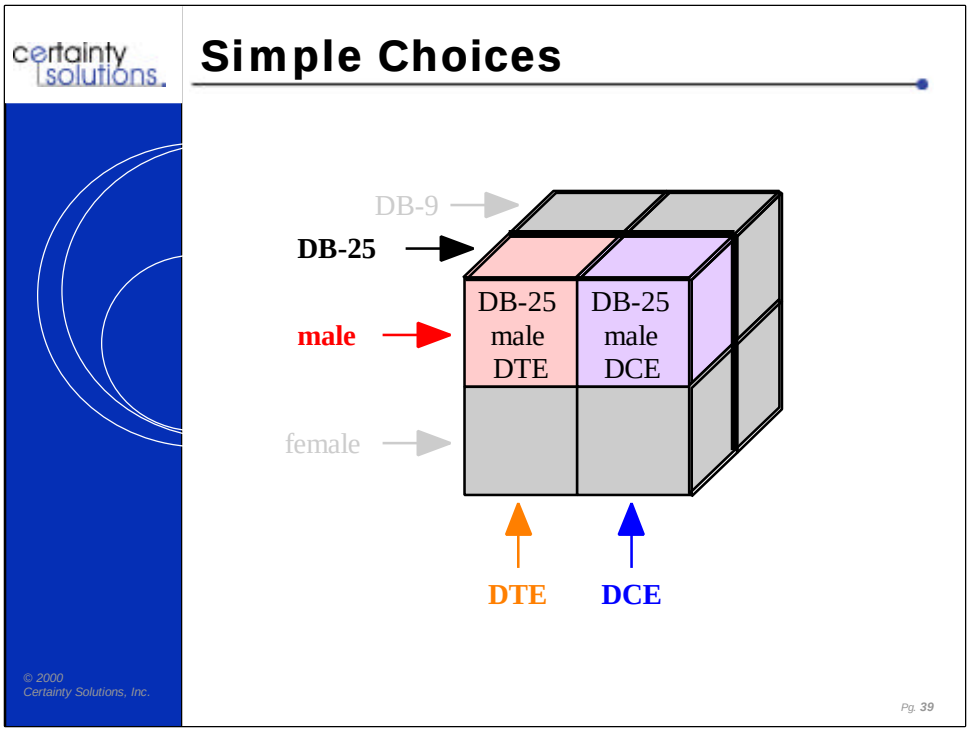
female-DTE

male-DCE

male-DTE

With two connector types, you have eight choices. Three connectors give you only 12 choices. The progression is simply 4 combinations per connector type.

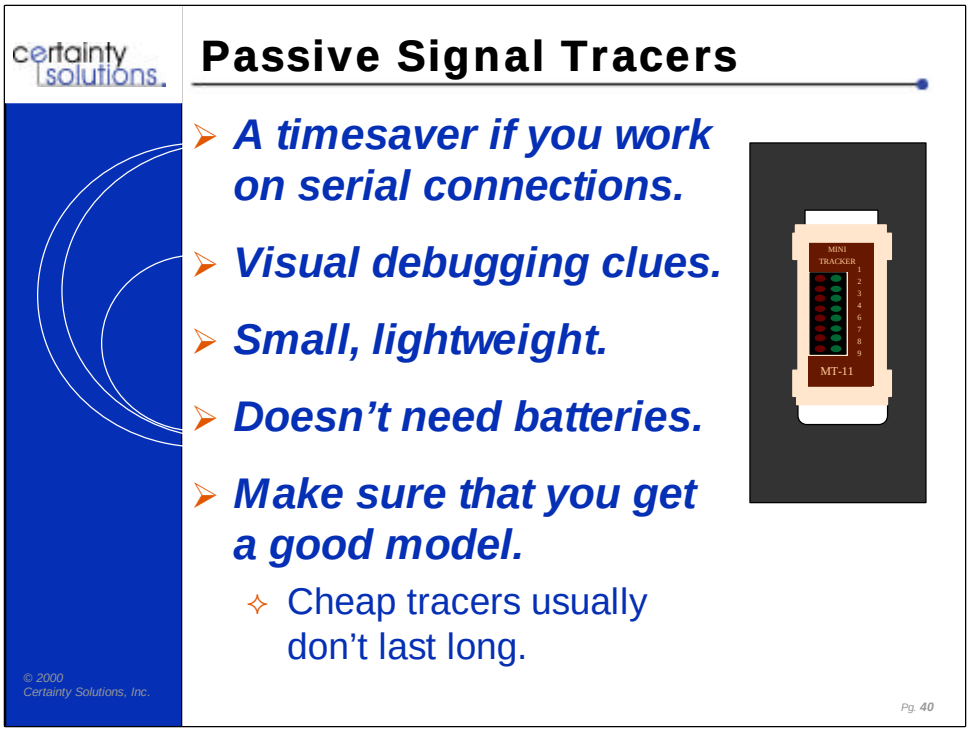
The good news is that there are not that many connector types in use for serial consoles. DB-9 and DB-25 adapter sets will provide for more than 80 % of your connections!



The good news about connecting devices is that you can look at your serial port, and you will know what connector type you have, and you know the gender, leaving you just two adapter choices, DCE or DTE.

You could try one of the two adapters, and test your software settings, and then try swapping the other adapter and test settings again. (You should ask yourself “how far is the walk between the console port and your application testing console?” and “How many times do I want to go back and forth?”.)

On the other hand, you could use a passive signal tracer or a breakout-box to determine which adapter is the correct one, and then work on the software settings with confidence. Just bring one of each of the proper DCE and DTE adapter, and your trusty signal tracer, and things should be fine.

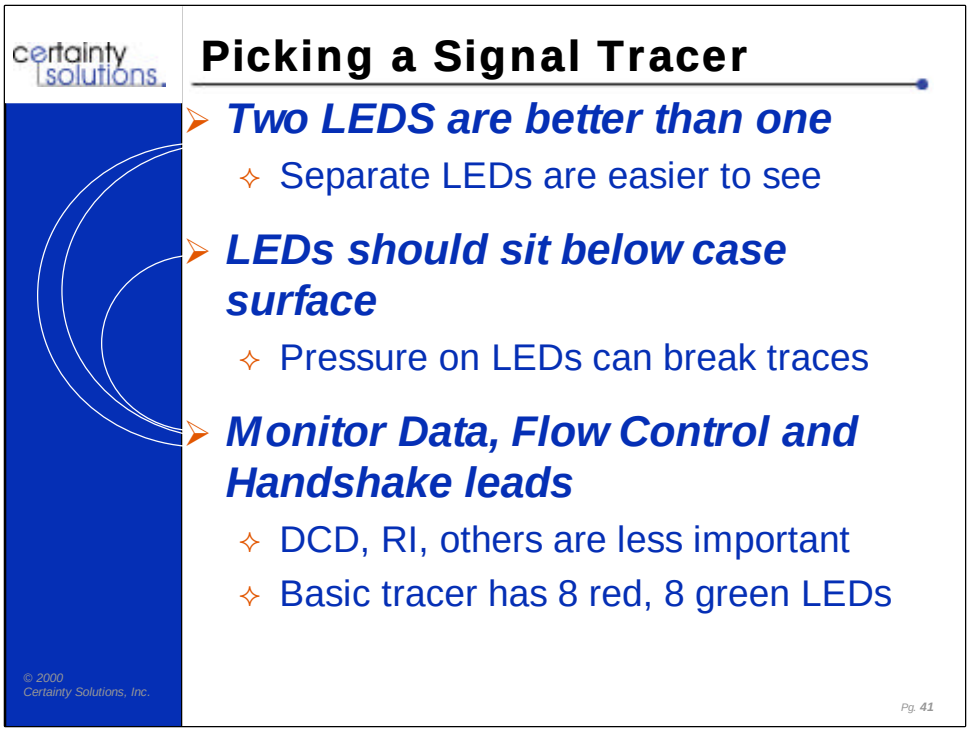


In my opinion, a passive signal tracer is more useful than a breakout box for most testing circumstances. I believe this because most cases do not require us to make special crossover cables, or jumper pins together, especially when you use pre-wired adapters that support all handshaking and flow control leads.

The weight of a break-out box can slowly damage connector pins, if you leave it hanging for long. (And we often left it hanging, when it was making a special crossover, didn't we?)

Most breakout boxes also require two (or more) batteries to operate. While adding to the weight, the batteries are also prone to failure, which means you also need to carry spare batteries.

Instead of all of these little problems, I prefer a small, lightweight device that shows me the signals. It can show me if I've got the correct adapter. It can show me if an adapter is wired for the wrong RJ-45 pinouts. It can also show me if this is one of the rare cases that I need to go find a breakout box, to test special wiring. (So, yes, I still have a breakout box in my desk, but not my bag.)



I've been at this a while, and I've learned a few things by trial and error.

1) It's easier to see fast signal transitions (single keystrokes) when you have separate "0" and "1" LEDs.

1a) Bi-color LEDs are not as easy to read for high port speeds

1b) A single LED doesn't tell the whole story

2) You want a rugged device. It doesn't matter how cute it is, how good it looks, or how small it is. Inexpensive usually equals poor quality. You probably should avoid the low-cost unit, unless it has had great reviews from someone you trust.

2a) If you can't always trust it, don't put it in your bag! Get a good one!

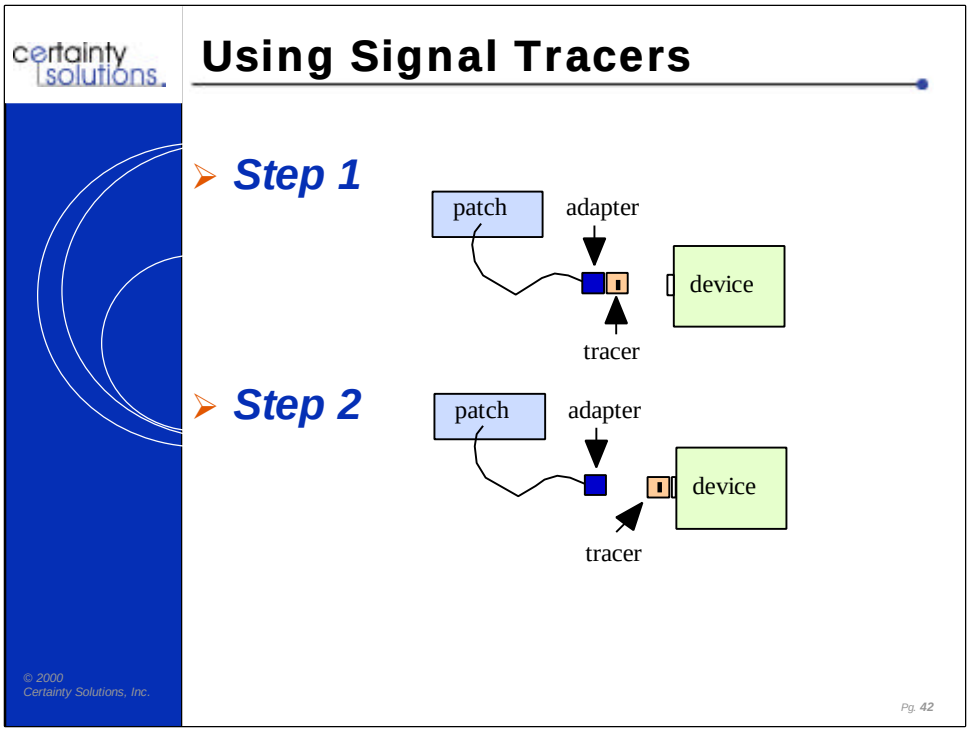
2b) A flaky, intermittent unit will cost you more in troubleshooting time

2c) An expensive unit is probably worth the money.

2d) Look for the features listed here first, then consider the price.

3) Look across the surface of the signal tracer. If you can see the LED sticking above the case surface, put it back and keep looking. When LEDs stick up too high, they will probably be less reliable over time.

I like the Datatran Corp. Mini-Tracker series units. The 8-signal units will typically cost $25-$30, and I've had a couple of these for nearly 10 years now, without a failure. (Americable stocks these as parts 90001 - 90003.)

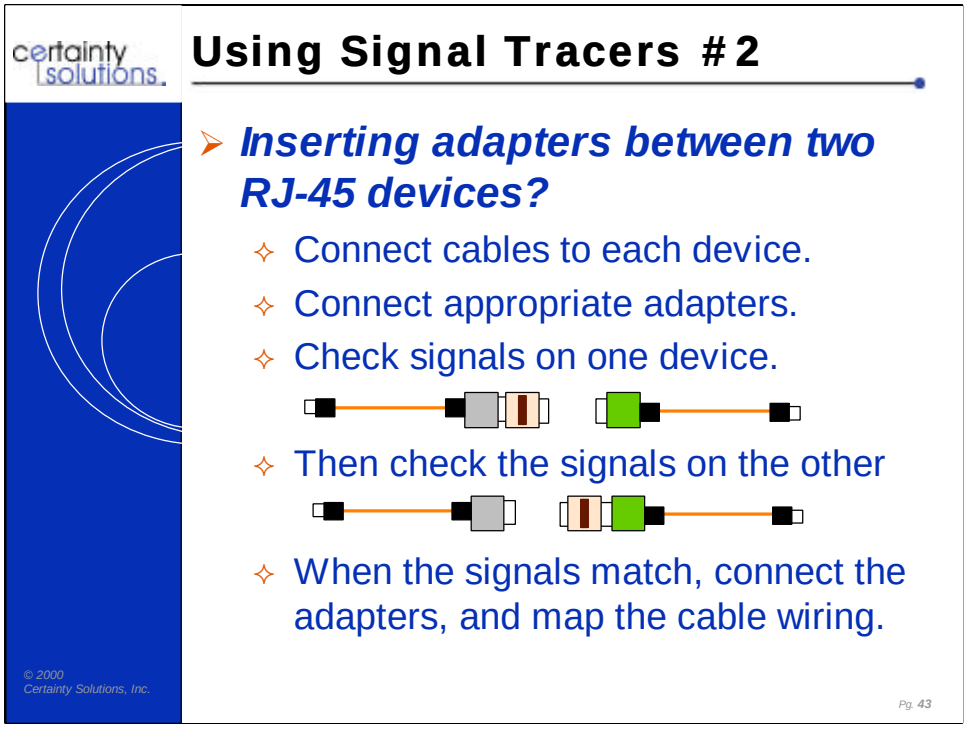


When you use pre-wired adapters, the first step is to make sure you are using an adapter with the RJ-45 connector wired for your terminal server. (This may mean that you have a pair of adapters, DTE and DCE, with the appropriate connectors for your host...at this point, you should pick one to test with.)

Step 1: Plug the adapter into the patch panel with a CAT-5 cable, and put the signal tracer on the adapter. Make a note whether you see a signal on pin 2 or pin 3. (You can ignore the other signals for now.)

Step 2: Take the signal tracer off the adapter, and move it to the console port on the device, and see whether you see a signal on pin 2 or pin 3. (If you have a signal on the same pin on the devices as you saw on the adapter, then you need to use the other (DTE/DCE) adapter for the connection.)

Once you have one signal on pin 2, and the other signal pin 3, you are ready to connect the adapter to the device, screw it down, and go test your software settings.



When you are connecting two RJ-45 devices, you usually have one of these two cases;

1) Each device is using a different RJ-45 wiring format.

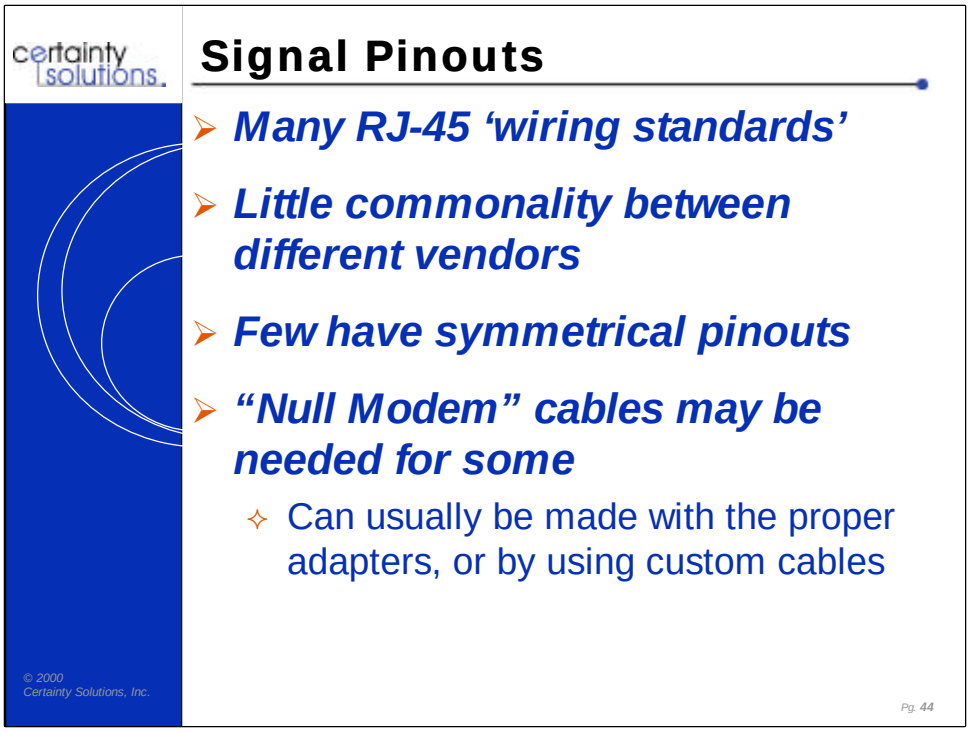
In this case, your job is a bit tricky, since you need to match the RJ-45 adapters the the wiring pinouts of the RJ-45 device. If you have a limited selection of adapters for one vendor, that may drive your adapter choice for the other wiring format. You will need to have a male/female, DCE/DTE matched pair of connectors in the middle. (That is, a male DCE and a female DTE, or a male DTE and a female DCE, one adapter matching the first RJ-45 wiring format, and the other adapter matching the second RJ-45 wiring format.

Once you have the signals straight between the adapters, you can remove the signal tracer, connect the adapters together, and then make a map of the cable, and then send us a note with the wiring information.

2) Both devices use the same RJ-45 wiring format.

In this case, you probably just want to monitor the signal leads between the two devices, and you need to put your signal tracer in between them. You simply need a pair of adapters with the proper RJ-45 wiring format, but you still need a male/female, DCE/DTE matched pair.

In each case, make sure that you are using 8-wire straight-through cables. Full 8-wire CAT-5 ethernet cables are perfect for the job.



There is no established standard for RJ-45 Signal Pinouts. While there are only 8 pins, you still have a wide variety of possible combinations where the signals can end up.

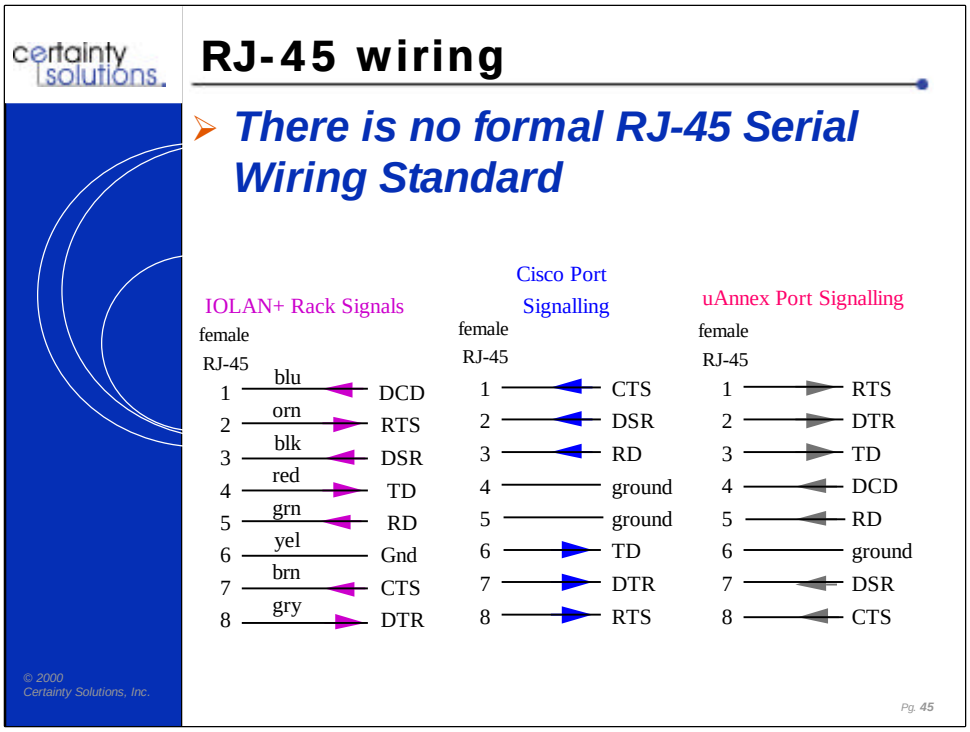
Most Terminal Server vendors use a unique pinout for their devices. Fortunately, most of the vendors use this same pinout across their entire product lines for their RJ-45 consoles.

While you always get the data leads and signal ground, you don't always get the hardware handshake leads and the hardware flow control leads. Few vendors implement DCD as well. (Some overlay it on one of the other leads...you can have DCD *or* the other signal, but not both.)

Cisco Systems, and Xyplex terminal servers use a symmetrical pinout format for their RJ-45 ports. This allows you to use a rolled (USOC-Reversed) cable to make a 'null-modem' connection.

While Cisco and Xyplex are both symmetrical, they are NOT compatible with each other.

Sun picked the Cisco wiring format for the consoles on their Netra T-1 model 105 server.

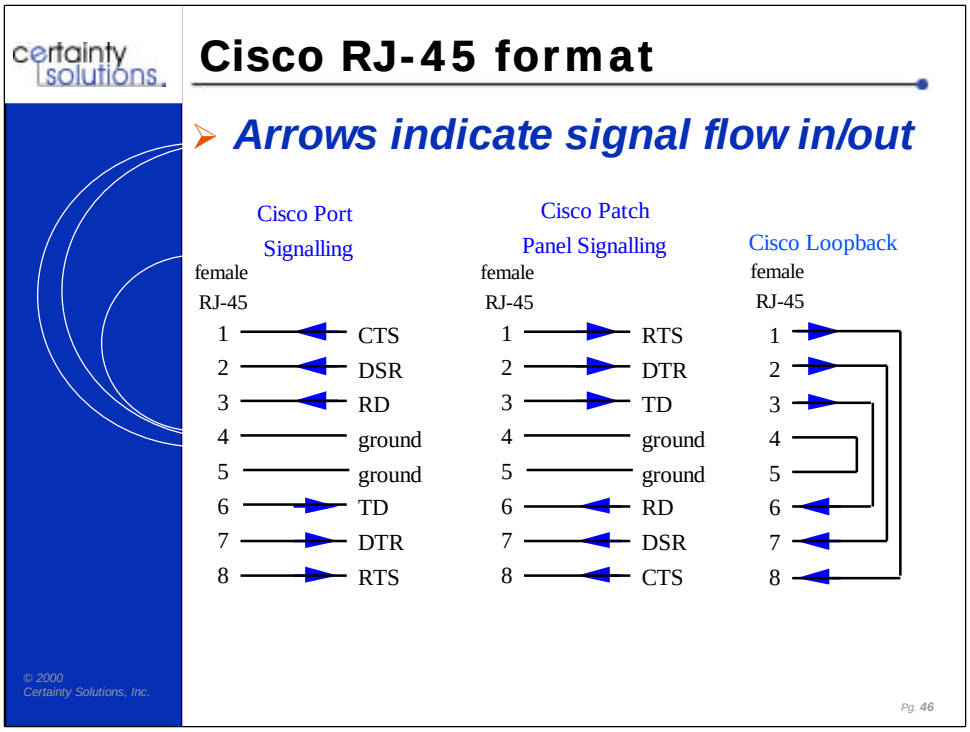


Receive Data (RD) is **usually** an input, but not always. (This confusion is prevalent between DTE and DCE devices, where the question is usually “Is RD a signal input, that I am receiving; or is it a signal that I output, sending data to the receiver of the DTE.?”.)

Most vendors have a unique pinout. I can’t find a good explanations for this. I would think that some vendors would want to align with each other, and share a standard, but this hasn’t been the case so far.

Aurora Technologies (http://www.auroratech.com) has a good, small handbook about remote serial console access, available for the asking. Their focus is to use distributed console servers (CPUs with add-on serial ports, therefore distributing your log files, but reducing the network traffic associated with using a central logging server (such as **Conserver**) and distributed terminal servers.

You can look at the signals shown here, and imagine what would happen if you plugged an ethernet cable in between any two of the three. You may connect one or two output signals to inputs on the other end, but you would rarely connect enough correct leads to establish a working data-only connection.



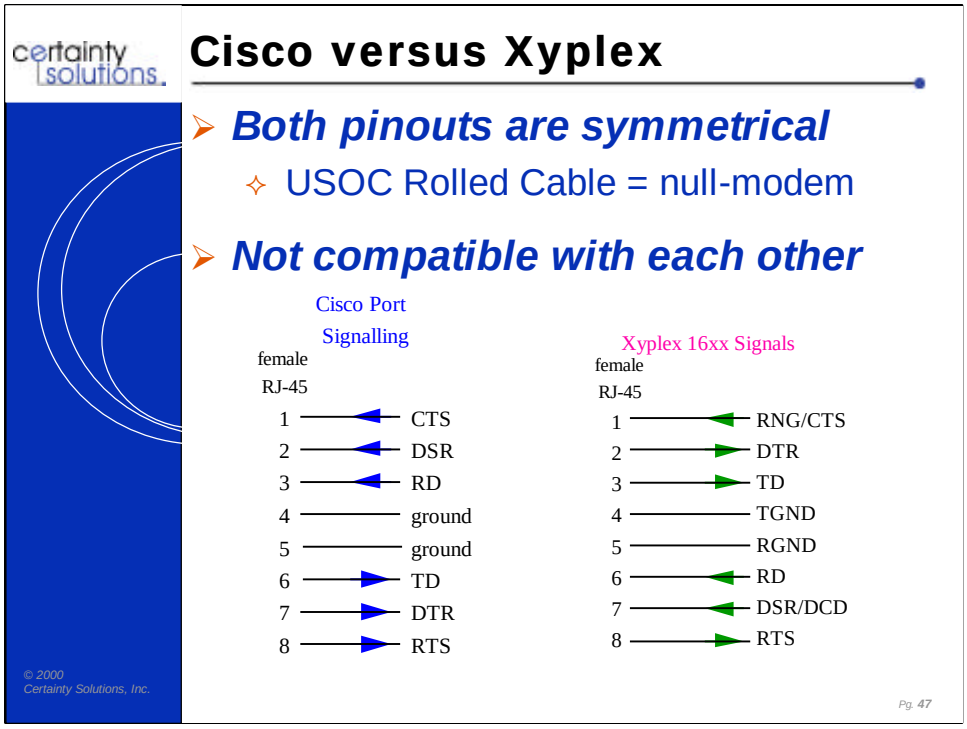
It's not important that you use my diagrams. But it IS important that you be consistent in making your diagrams.

In the Cisco device RJ-45 console ports, CTS, DSR, and RD are input signals from the 'other device' to the Cisco console port.

Notice also, that the signals are symmetrical. That is, the center pair are ground signals, the next pair out are data, the next pair are handshaking, and the outer pair are flow control. (You can see the symmetry in the loopback schematic shown here.)

In the case of the patch panels built by Americable, you can see that these signals are still inputs, but that they now live on different pins. This was done to make the patch panel jacks the null-modem opposite from the consoles, so that can use CAT-5, 8-wire ethernet cables to connect the console to the patch panel.

The Sun Netra T1, model 105 (project name: Nordica) uses an RJ-45 console port, wired to look like the Cisco Patch panel above! You would use a rolled cable to connect this to a Cisco patch panel.



There are similarities:

Symmetrical signals

Simple USOC Rolled cable can be the null-modem.

Ground and data leads are in the center, so a 4-wire RJ-11 cable can provide the data-only connection between the devices.

How they are dissimilar:

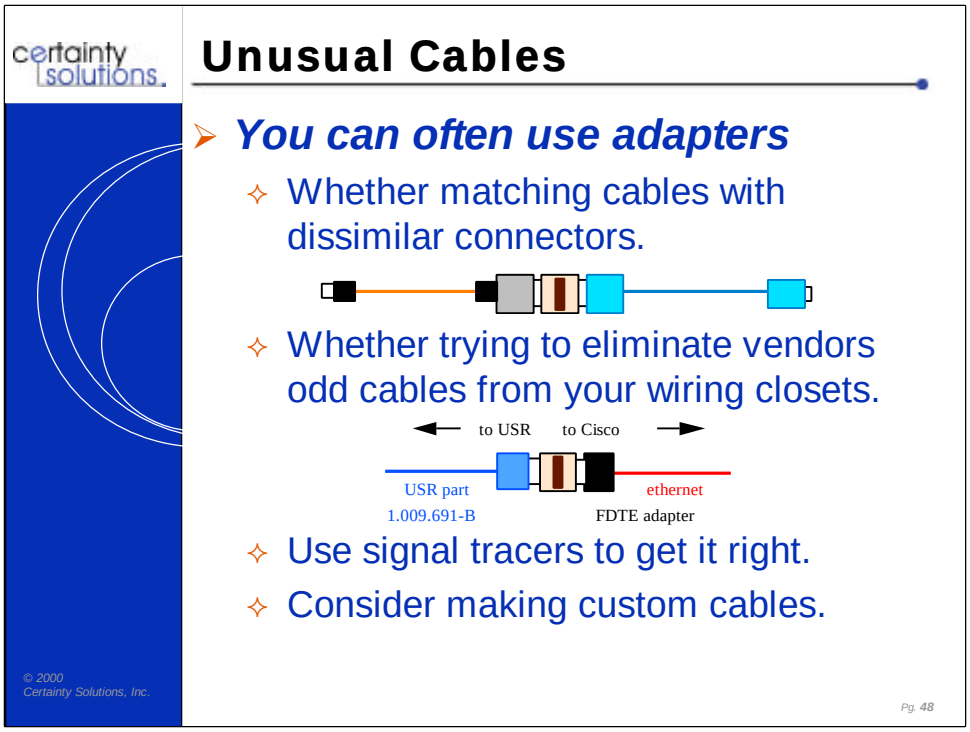
Cisco groups inputs together, outputs together. (Xyplex mixed them.)

Data and Handshake will connect with the wrong adapter, flow control won't.

The Xyplex terminal servers overlay signals on two leads.

Without a passive signal tracer, you might not realize that you have used the wrong adapter, until you lost data when the hardware flow control didn't work.

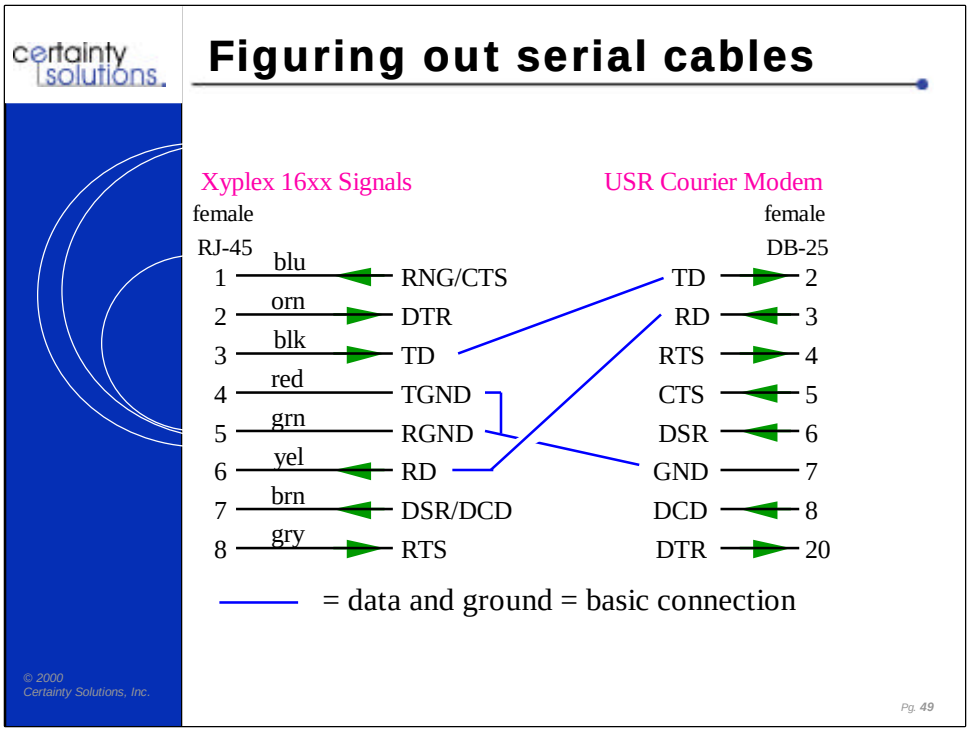
When it comes to mixing the wrong adapter with your terminal server, this combination is about as good as you could hope to get.



When connecting a DB-type connector to another DB-type connector, you can use adapters on each device, and a CAT-5 cable in between. (When you do this, the RJ-45 format must be the same for both adapters.) Then use the passive signal tracer to verify that the signals look correct on both ends.

When you are connecting an RJ-45 console with a different wiring format from your terminal server, you can still use a pair of adapters, but one adapter needs to match the RJ-45 format of the console, while the other adapter needs to have an RJ-45 format to match your terminal server. (Use the passive signal tracer in between the DB-type connectors to ensure that you have the correct DTE-to-DCE match.

When you have adapters in the middle of a cable, you should make sure they will not come apart. You should also consider figuring out the pinouts from one RJ-45 to the other, and make a custom cable between the console port and the terminal server.

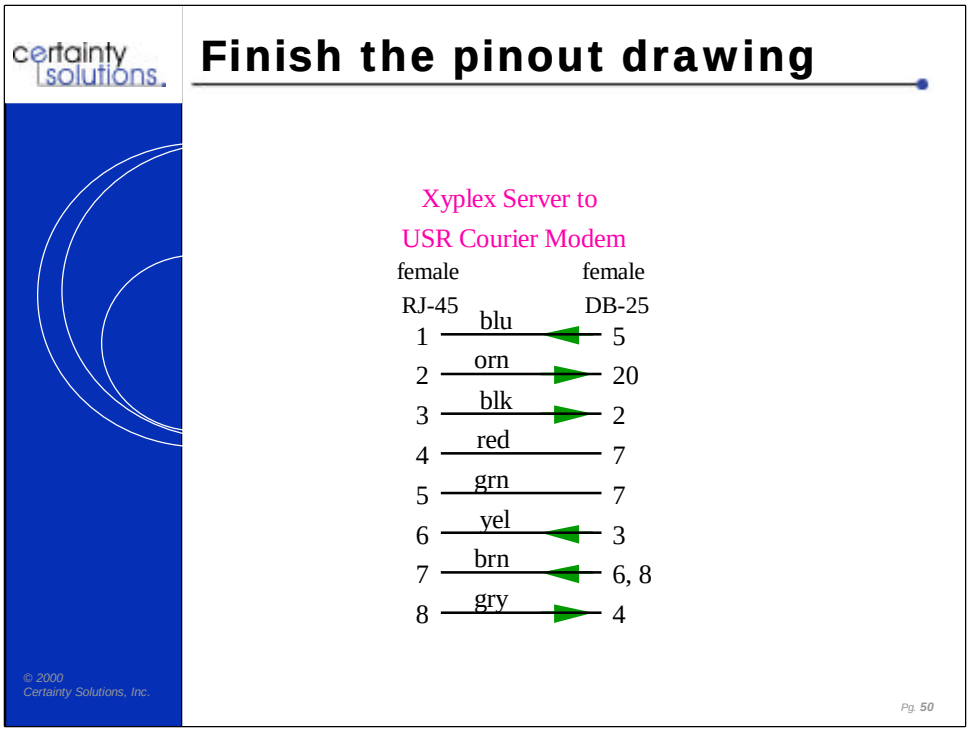


Get the pinout information for each device that you want to connect together. You need to find;

+ The connector type and gender.
+ Which signal is on which pin (for each device).
+ Whether each pin is an input to (or output from) each device.

Next, connect the lines.

+ Find the signal grounds, and connect them. (shown above)
+ Find the data leads, and connect data output from one device to data input on the other device. (shown above)
+ Find the flow control leads, and connect the flow control output on one device to the flow control input on the other device.
+ Find the hardware handshake leads, and connect the output on one device to the handshake input on the other device.

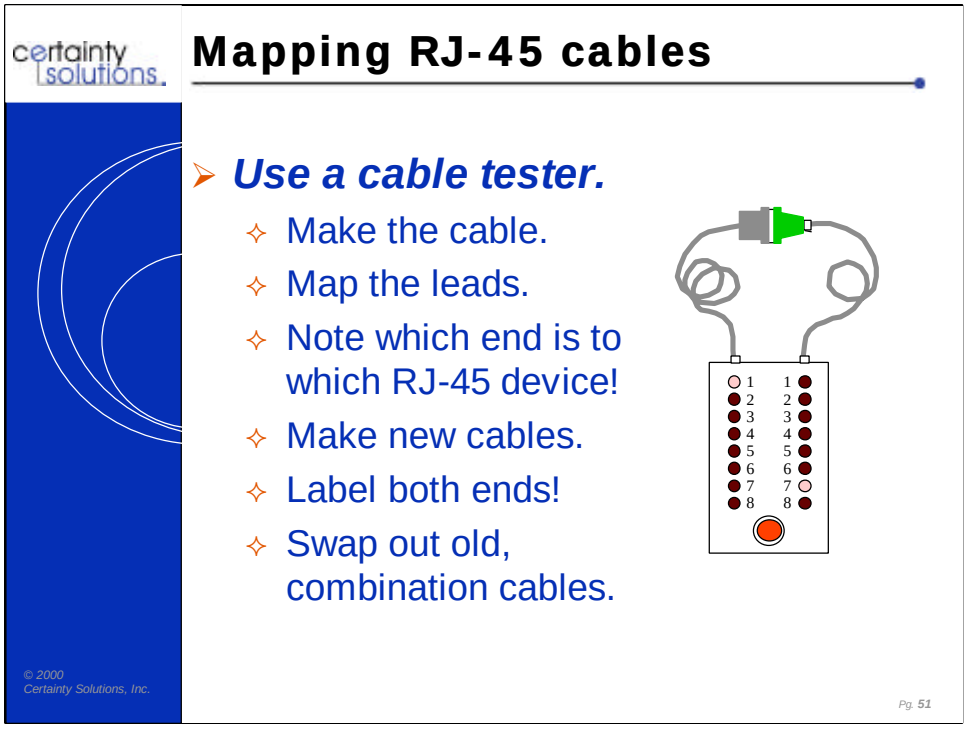


Once you have connected the lines from your drawing, you have the information you need to redraw the diagram concisely.

I make a point of including the signal flow arrows, as an aide for troubleshooting later. It can also provide an important jumpstart when you need to hack a special cable when manuals are not handy or available.

The wire colors shown here are the common colors associated with the RJ-45 female jack, derived from telephone history, and still used today.

I'm always curious to receive pinouts for adapters or cables that I don't have on my adapters pages. If you cobble up a cable or adapter, and you're willing to share that information, I'd appreciate getting the information in an email to **consoles@conserver.com**



You may have used adapters to attach a vendor's specialty cable to some other devices, or to your terminal servers. You may have used adapters to match the signals between two dissimilar RJ-45 wiring formats.

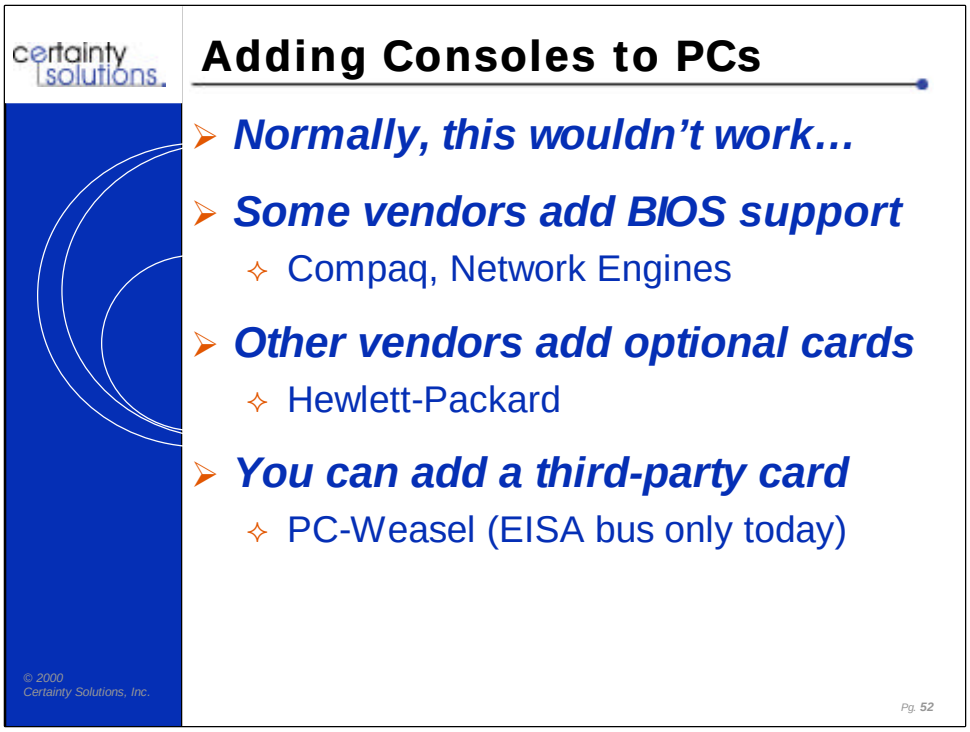
BUT, having adapters in the middle of cables will get in your way, making it tough to recover cables from crowded wire managers, and they often thwart easy troubleshooting.

(Consider that you may have different colored CAT-5 cables leaving the adapters in the middle…and even if the colors match, the serial numbers and lengths will not!)

For the reasons of neatness, and ease of troubleshooting, I recommend that you make specialty cables, once you have figured out which pins on one end go to which pins on the other end. Then, make sure that you label both ends, so that whoever inherits your cables will recognize the specialty cables.

In many sites that I manage, we use beige CAT-5 ethernet cables with black boots and serial number labels to designate serial console cables. When we make specialty cables (in a Cisco terminal server shop), the Cisco-wired end keeps the black boot, but we put an orange boot on the non-standard end, and use a self-laminating label to identify which RJ-45 format the cable is wired for (USR, Xyplex, IOLAN, etc.)

We'd love to get a note from you about custom cables that you have figured out. Send a note to **consoles@conserver.com**.



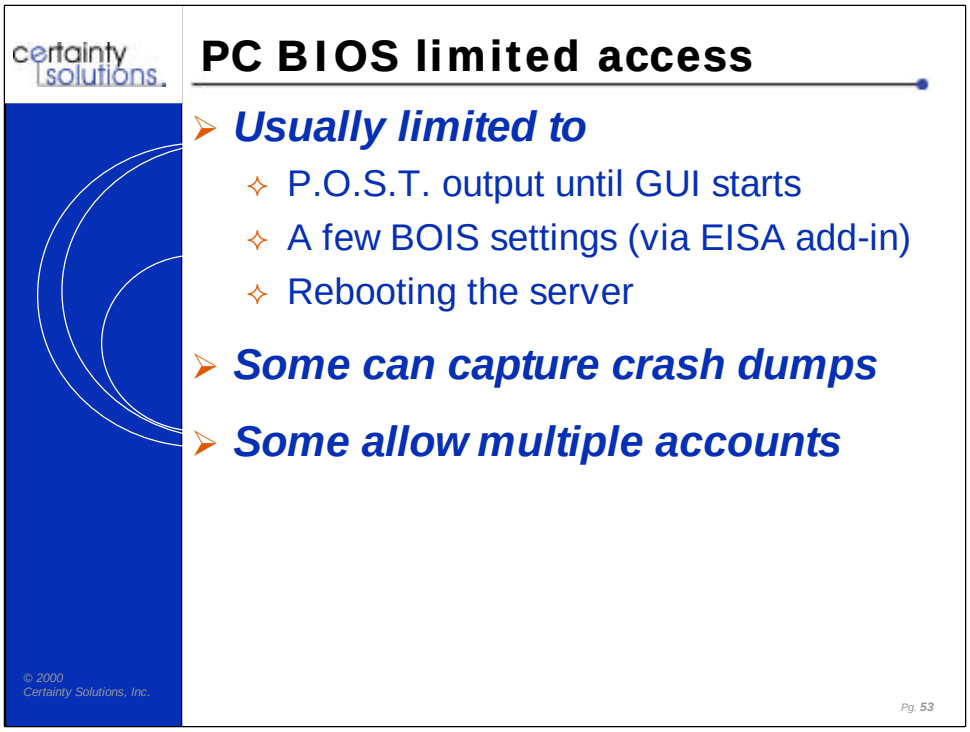
Traditionally, the PC architecture has not lent itself to using the COM ports as serial consoles, they way that UNIX machines have.

Even if you are running a UNIX OS on a PC, you have normally needed to wait until the OS is running before the serial console starts passing traffic.

In recent years, some larger vendors have been adding some console capability in the BIOS. Most often, this is basically just redirecting the Power-On Self Test (POST) output to a serial port.

There have also been attempts to use add-in cards to add another path for managing the basic operations of the servers through a serial console port (instead of a built-in COM port.)

The PC Weasel add-in card is currently only available in EISA bus, but it appears to the PC as a Hercules monochrome card, but it allows you to control the BIOS settings, and redirects all of the monitor output up to the GUI activation, as well as providing some control over remote rebooting. (http://www.weasel.com/)



Owners of newer Compaq servers have the ability to control basic server functions (power cycling, soft resets, some BIOS support) via a serial console port.

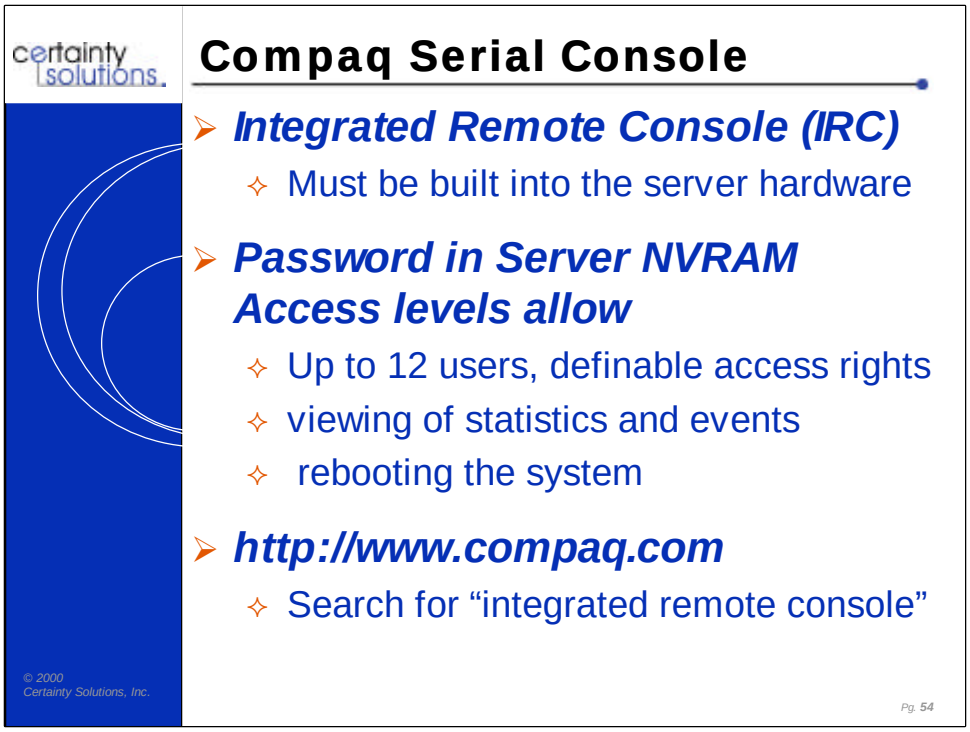
Normally, server owners would connect a modem to each of their server serial ports, with a dedicated modem for each.

Using terminal servers, you reduce the recurring costs of all the phone lines. (Even if you are using analog lines in your phone switch, those lines are not cheap resources.) You can then rely on the strength of your remote access authentication mechanisms to protect access to your server ports, rather than relying on password challenges from the modems.

Modems cannot adequately audit successful (or failed) access attempts, and they don't support remote authentication servers or security token devices.

If you need more security or auditing, your RAS solution can give you more security than modems on console ports.

NOTE: Most PC's communicating via BIOS support require DCD being active before the PC will listen to traffic, even if DSR is active! The developers thought you would only be connecting them to a modem for access...



Go to the Compaq web site, and get the primer for Integrated Remote Console. (http://www.compaq.com/support/techpubs/user_reference_guides/281862-002.pdf) You can order the document as Compaq part number 281862-002.

The "feature" needs to be built into the server hardware. Check the standard features on your servers, to see if you have it.

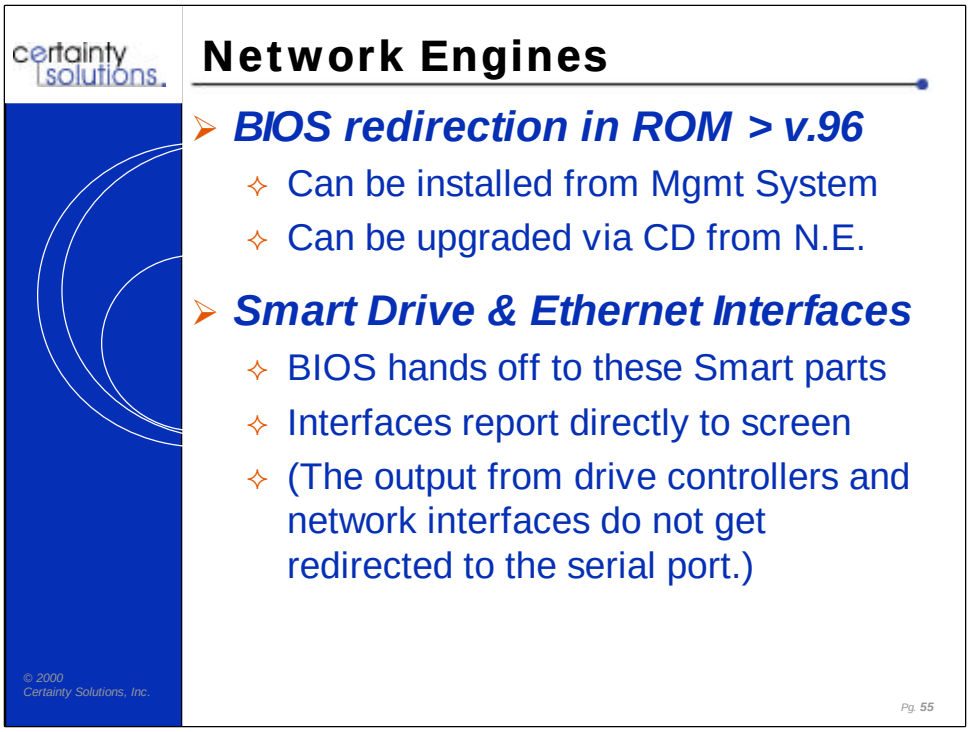
This console redirection goes away after the OS goes into graphics mode.

The hardware does not support remote control of Windows NT.

Console (P.O.S.T.) redirection works as long as the display output is ASCII text. Prior to graphics mode, one can boot to DOS or run the EISA configuration utilities.

David Primmer recommends the NT lightsout guide. It's a set of tools and documents that are designed to provide information about setting up, managing, and running Windows NT systems in a headless "lights-out" environment. Go to the Compaq homepage, and search for the keywords "lights out".

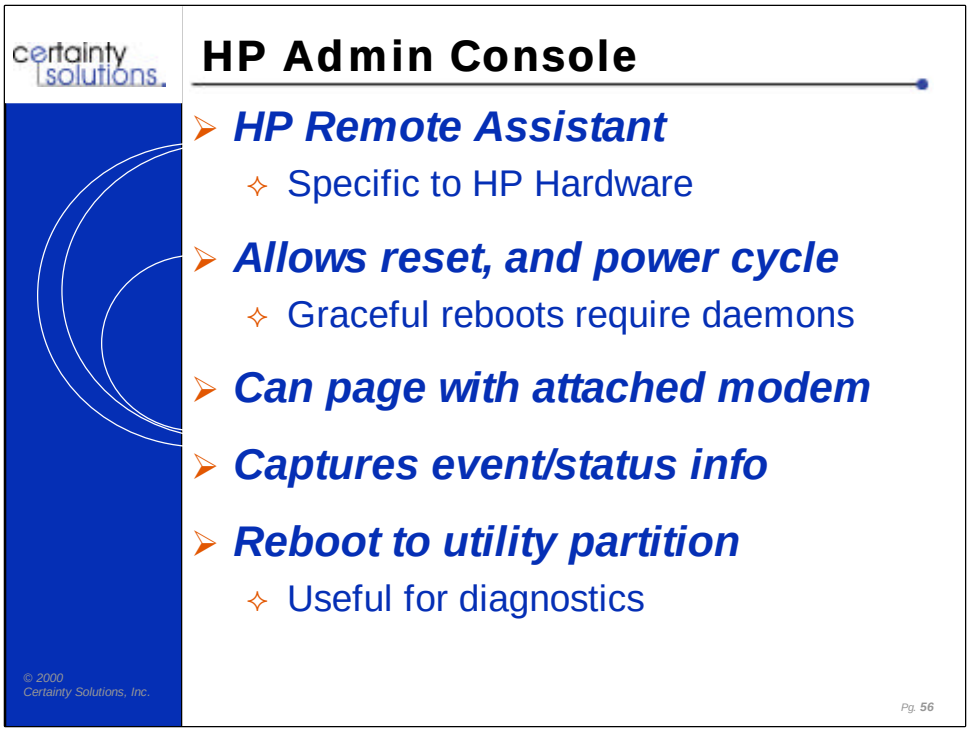
The Proliant and Prosignia lines have an optional Integrated Remote Console, which connects to the ethernet network, and is reachable, even if the server is not functional, using a GUI interface. (http://www.compaq.com/manage/remote-management.html)



The Network Engines WebEngine provides for a level of POST redirection, but the BIOS hands off control to the Ethernet controller, and the Hard Drive controllers…and you never see any of the output from these separate controllers, just from the BIOS.

If you have an older unit, contact your Sales Engineer, and ask to get an upgrade CD, or schedule an appointment to have them upgrade the FlashROMs.

This is an early attempt at adding Console redirection at startup. It was added after the product was developed, and as a result, is limited in ability to control some server functions. But the company now understands the needs, and with input from IBM as a partner, should include enhanced capabilities in the newer product models.



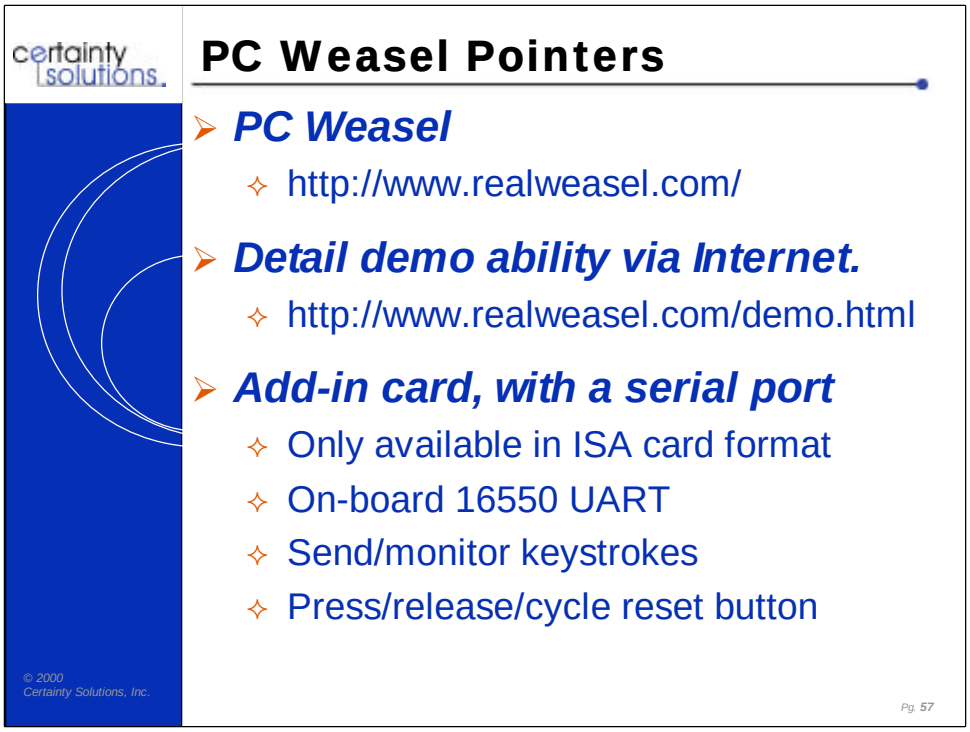
Some Netserver CPUs have some remote control capability built in. (HP/UX does have the ability to use a serial port as a console,) You can also add the control capability by installing an HP Remote Assistant management card, with connections to the motherboard for some control functions. You also need the NetServer Navigator CD to configure the system.

The EISA card has an on-board battery, which allows you to power-cycle the CPU as well, if the server is compatible.

The system really wants to be controlled over ethernet, but can be used with a modem, or a direct serial connection.

The emulation is ANSI, Color!

The console can also give you NT Graphics redirection, with appropriate client software on the workstation(s) that you will use to mange the server(s).



The devices appears to the computer to be a monochrome video adapter. The P.O.S.T. information that is sent to the video adapter interface is then sent to the serial port on the card.

Once the system starts booting the OS, and switches into a graphical mode, the card does two things;

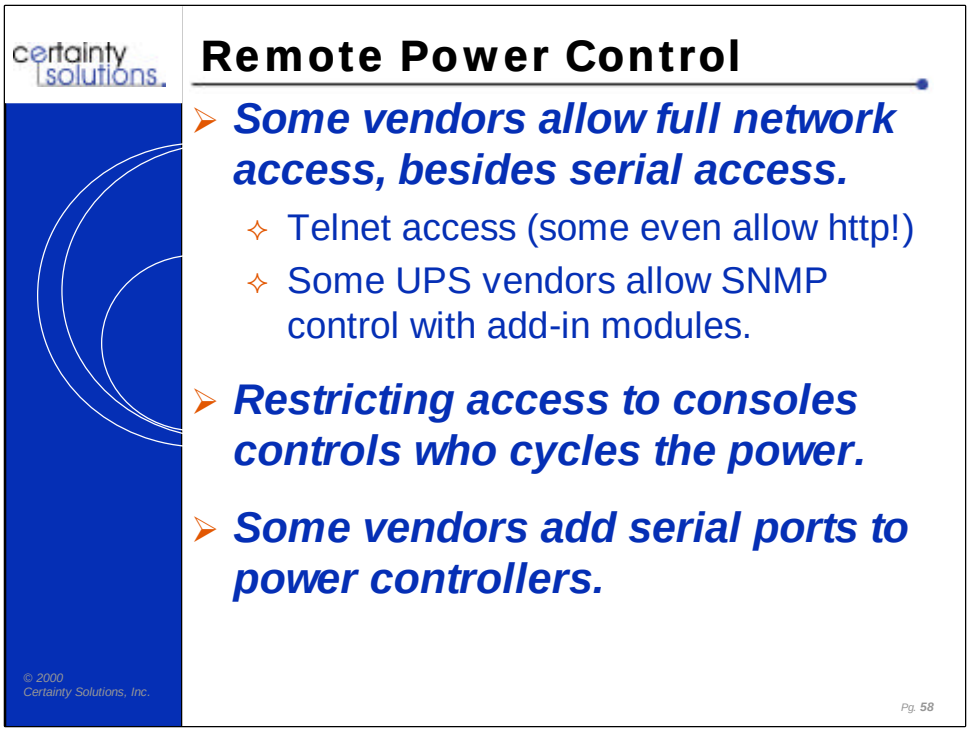
1) Gives you the ability to perform some basic functions, including examining the keyboard buffer, and/or sending keystrokes into the keyboard buffer. You can also push, release, or cycle (push-release) the reset button.

2) If the OS tries to use a serial port as a serial console, the UART on the card will then connect that stream to the serial port on the card. That allows you to use the same connection for your serial console as for basic CPU control.

There have been lots of requests for PCI cards, but clearly not enough (yet).

Canada Connect Corp., Calgary, Alberta

http://www.realweasel.com/

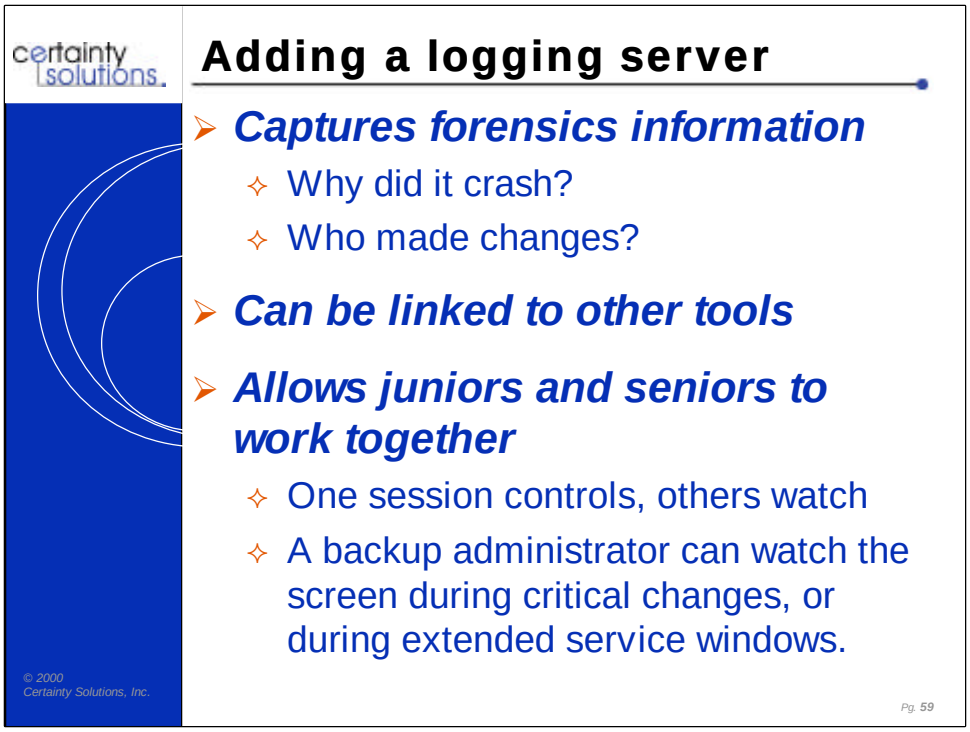


Many UPS vendors allow SNMP management of their larger units. This often includes the ability to shut down output circuits. (However, that's not often practical, since many machines are usually on each circuit.)

American Power Conversion Corp. (http://www.apcc.com/) make a series of remote power control devices that let you cycle power on individual outlets. These units also let you set up a cascaded startup sequence, to avoid power surges when power on the mains return. These devices include a 10-base-t interface, with telnet and http listeners as a standard feature. (http://www.apcc.com/products/masterswitch_plus/index.cfm)

The BayTechDCD folks offer a line of intelligent power controllers in many different configurations. Some of their products even include modems for dial-up access, as well as serial port control. (http://www.baytechdcd.com/)

Server Technologies also has some remote power control devices. Some include a serial switch, so that you can switch your session to one of the attached devices. These also include the feature that you can examine an approximate power consumption for each of the switched outlets via remote control! (http://www.servertech.com/products/Default.htm)



You can integrate a commercial logging server with your terminal servers.

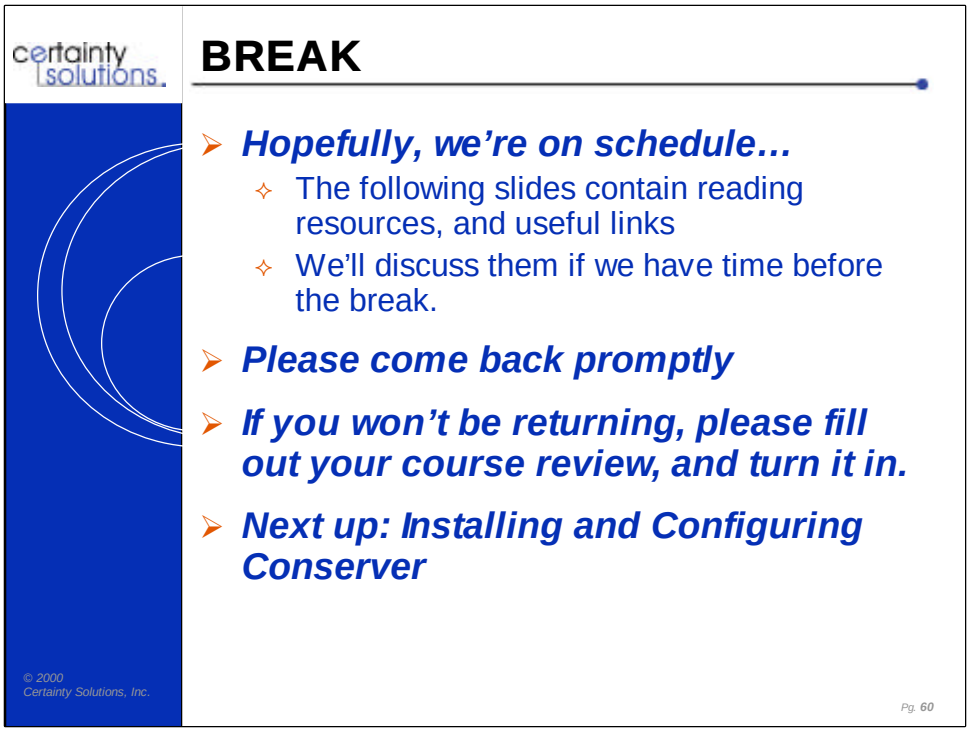
You can install many logging servers *instead* of terminal servers.

You can find freeware servers, and save some money.

You might be able to program your own in-house…

But ADD SOMETHING! The benefits of these tools are that they will save you time, help keep your systems on-line, and give you better visibility into how your devices are operating. They provide mentoring and training capability in an area where most shops have trouble making the effort to train staff. They give you the ability to distribute the workload across many staffers, regardless of their physical locations.

These tools should be in every shop, to help you do more, much more efficiently, and much more quickly.



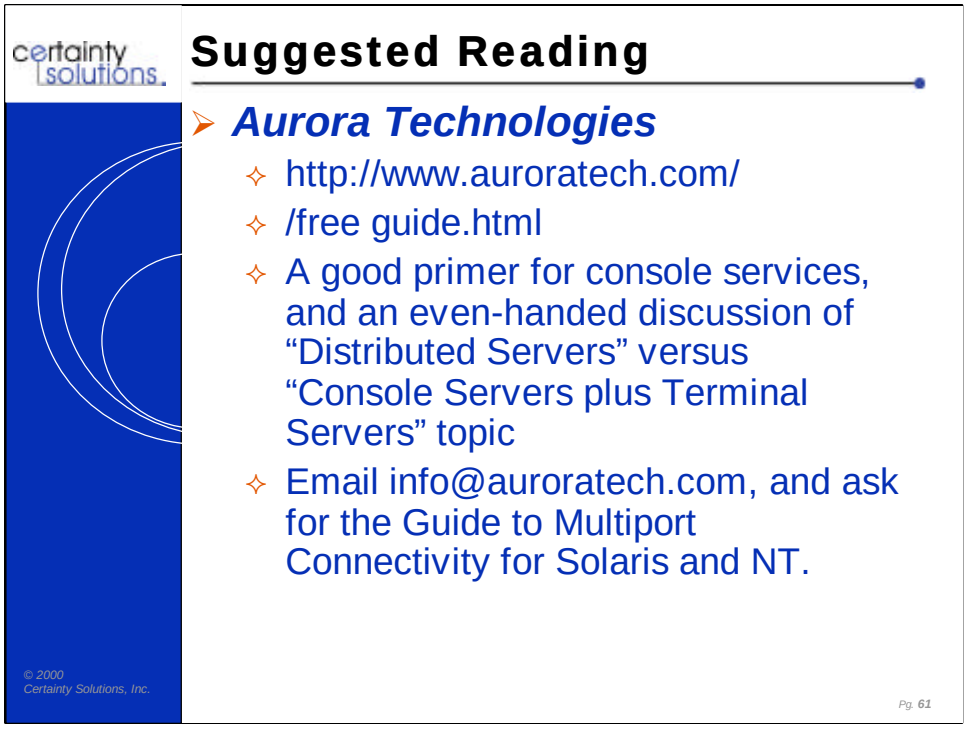
In the world of Serial Communications, the term BREAK is significant, as an out-of-band signal on the in-band path. Not every device can send a BREAK, and many devices ignore the signal.

However, there are cases where the BREAK signal can help, or hinder the flow of serial communications;

+ modem devices usually perform a warm-reset when the see it, which will interrupt any in-progress call, and set many of the registers to known states.

+ Sun computers will see this signal and effectively stop all processes, dropping to the OK prompt. At this point, everything the computer was doing is in a sort of stasis…typing "go" will let the machine continue as it was; typing "boot" will perform a reset of the CPU, losing all previous process state info, and possibly damaging any data structures that were in an opened state when the machine received the BREAK.

For us, a BREAK is a chance to stretch, eat, drink, and chat!

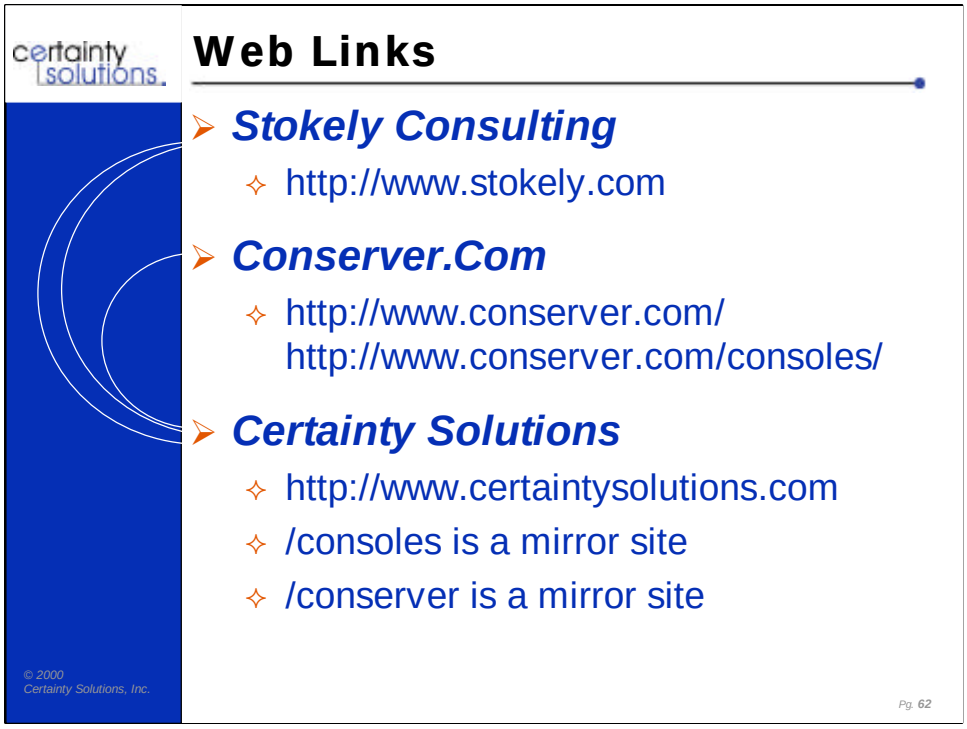


There are very few comprehensive works, explaining how to set up serial console services, which is why we developed this tutorial. However, Aurora Technologies has provided a good discussion of the topic, called "Guide to Multiport Connectivity for Solaris and NT"

The Aurora folks lean more towards a commercial server application (with support!), and they favor using many servers with multiple-port async cards installed in the servers.

While Conserver can support multi-port cards in the servers, we also feel that Terminal Servers are probably a better solution for large sites, or for sites with needs to connect many distributed ports.

Rather than just taking our word for everything we have described here, we would encourage you to contact Aurora Technologies, and ask for a copy of their guide. Read through it, and get their side of the story as well.

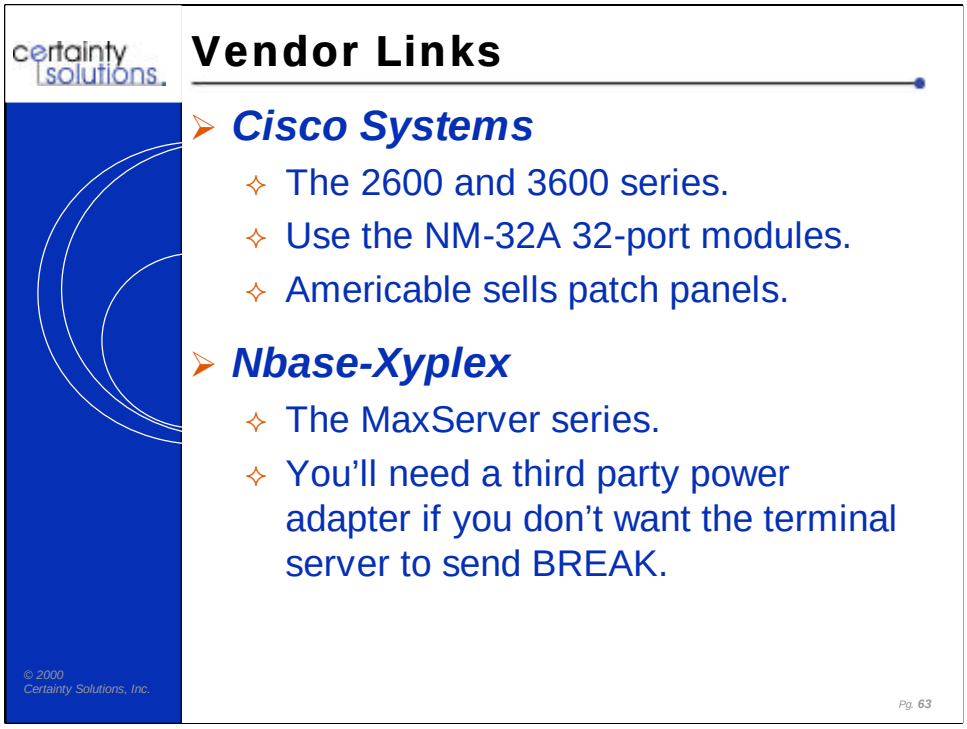


Celeste Stokely has been providing a lot of useful information on her website, and you can find a good deal of information about the problem that Serial BREAK can cause a Sun CPU, as well as a vast array of systems administration topics and tutorial pages. (There are other useful pages as well. Check it out!)

Bryan Stansell has put Conserver.Com on the Internet, to make it easier for folks to find the most recent version of the Conserver code.

David K. Z. Harris has been working on the serial console guides for 5+ years, and has recently moved them to the conserver.com. These include Console connection guides, as well as some basic serial tutorial pages.

Both Bryan and David work for Certainty Solutions, and have established mirrors for their web pages under the “tech-advice” section of the website



Terminal Server Vendors

Cisco Systems http://www.cisco.com

http://www.cisco.com/univercd/cc/td/doc/pcat/2600.htm

http://www.cisco.com/warp/public/cc/pd/rt/2600/prodlit/2600_ds.htm

http://www.cisco.com/warp/public/cc/pd/rt/3600/prodlit/seral_ds.htm

http://www.cisco.com/warp/public/cc/pd/rt/3600/prodlit/3600_ov.htm

Xyplex

http://www.nbase-xyplex.com/misc/adobe.cfm

http://www.nbase-xyplex.com/products/access_server/maxserver_series.cfm

Adapter Vendors

Americable 800-328-7954 http://www.americable.com

http://www.conserver.com/consoles/ciscokit.html

(also annexkit, iolankit, and xyplexkit)