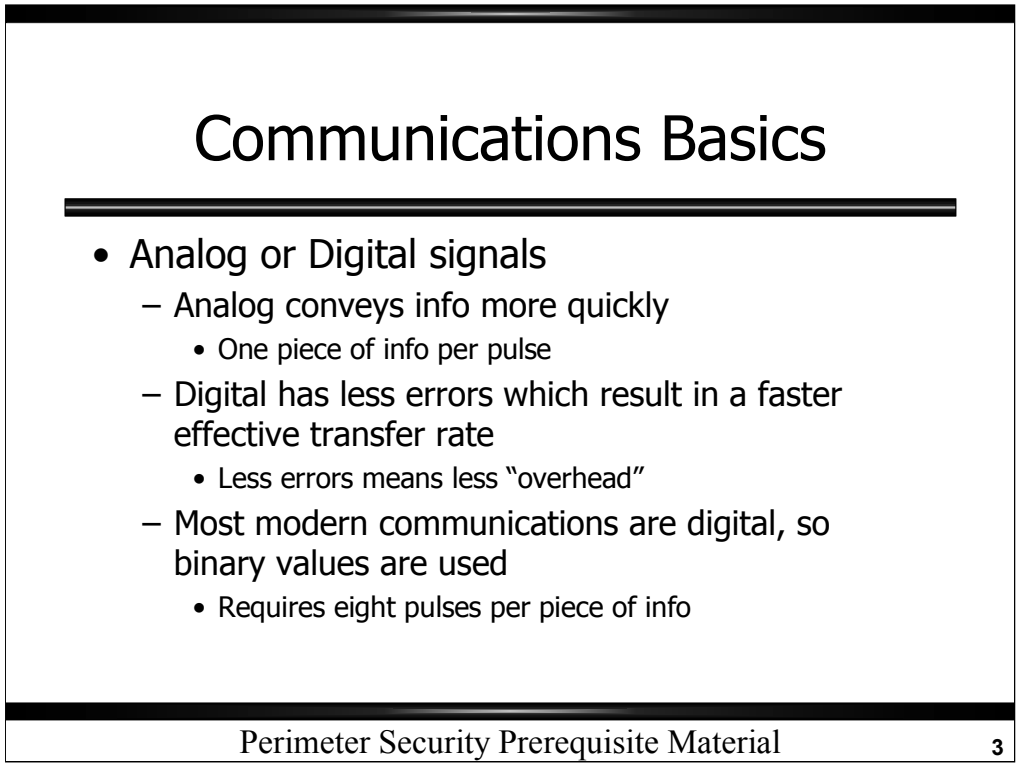
Perimeter Security Prerequisite Material

Material Overview

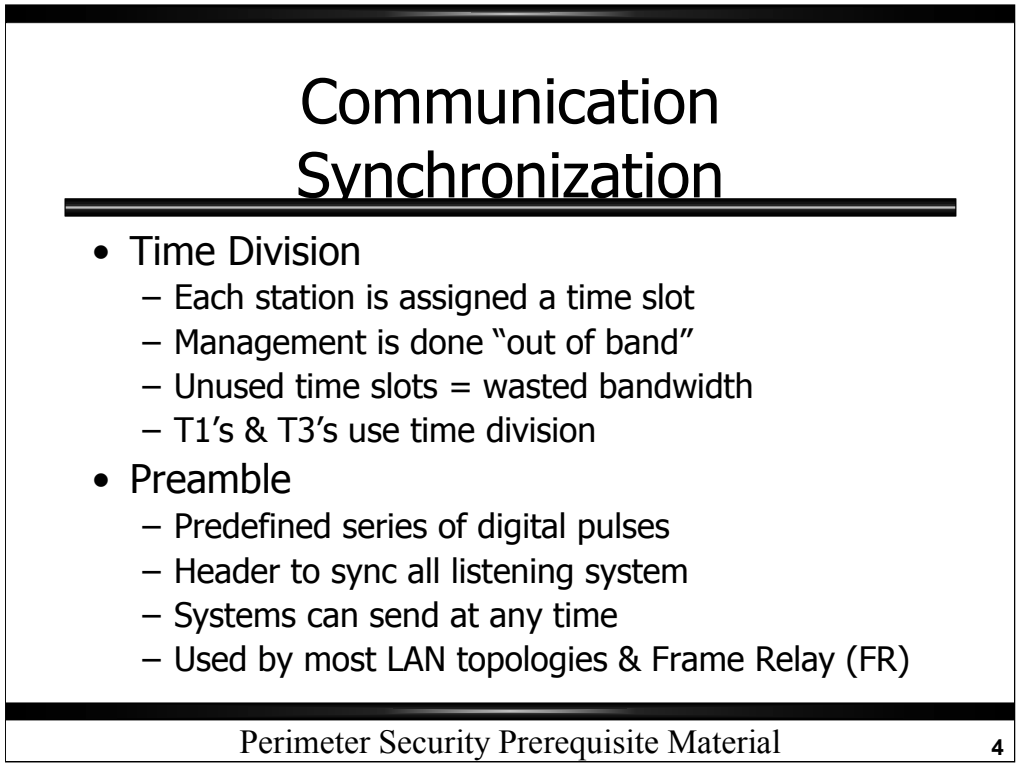
- Communication Basics
- Physical & Logical Topologies
- Performance Metrics
- Protocols
- VPN Basics



There are only two real options when transmitting network information, analog and digital transmissions.

Analog is the most efficient of the two. Since digital is based on binary, it takes eight digital pulses to convey a piece of information like the letters “A” or “a”. An analog signal is capable of transferring information on every pulse or cycle. For example the letter “A” could be an analog pulse with an amplitude of 3 volts while an “a” is expressed as 3.1 volts. This allows us to express these two letters with two analog pulses. To do the same in the digital world would require 16 pulses (two sets of 8 bits).

So why use digital? One of the problems with analog is its susceptibility to noise. For example a small amount of electromagnetic interference (EMI) may change the analog voltage to 3.05 volts. Is this an “A” or an “a”? You have no way to tell. The circuit would have to retransmit the information with the hope that noise would not alter the signal a second time. Since digital is based on binary, everything is a one or a zero. There are no values in between. If I express a zero as zero volts and a one as 5 volts it becomes far more difficult to interfere with the circuit. Not impossible, but far more difficult. For this reason most modern network communications rely on digital transmissions.

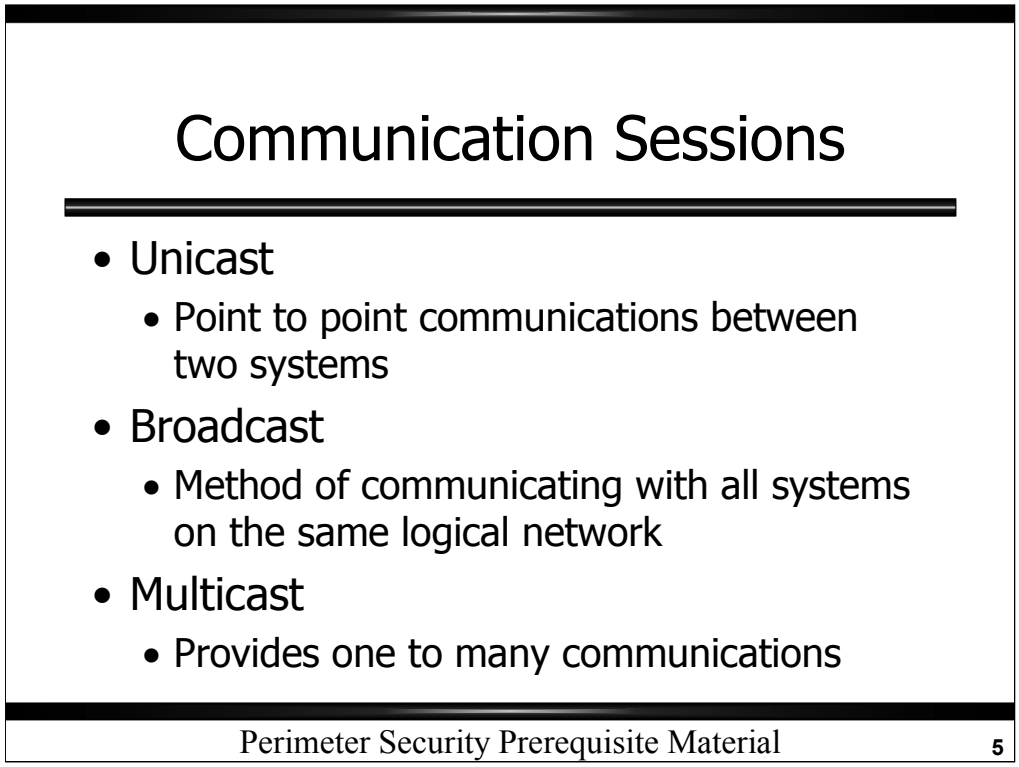


No matter what type of transmission is used, there needs to be some way of synchronizing communications. Think of a drummer laying down the beat at the beginning of a song and you'll get the idea. This is done to ensure that all the band members begin the song at exactly the same time. Network communications require the same type of synchronicity to ensure that a station does not mistake the middle of one transmission for the beginning of another.

Time division is one way of synchronizing communications. With *time division* every host is allocated a time slot in which to communicate. When their time slot arrives they are allowed to communicate for as long as their time slot lasts. If they still have data to transmit when the time slot expires, they must wait for the next one to arrive before continuing. Also, if a system has nothing to say that time allocation goes to waste.

Because of its wasteful nature, time division does not scale well. Think of having 200 systems on the wire, 195 of which have nothing to say and you'll get the idea. Time division does not play well when you have a large number of hosts.

When many hosts are involved, a preamble is the way to go. Think of our drummer analogy again. The *preamble* is a predefined set of pulses that let other systems know that a transmission is about to take place. This creates an ad hoc environment where systems are allowed to transmit whenever they need.

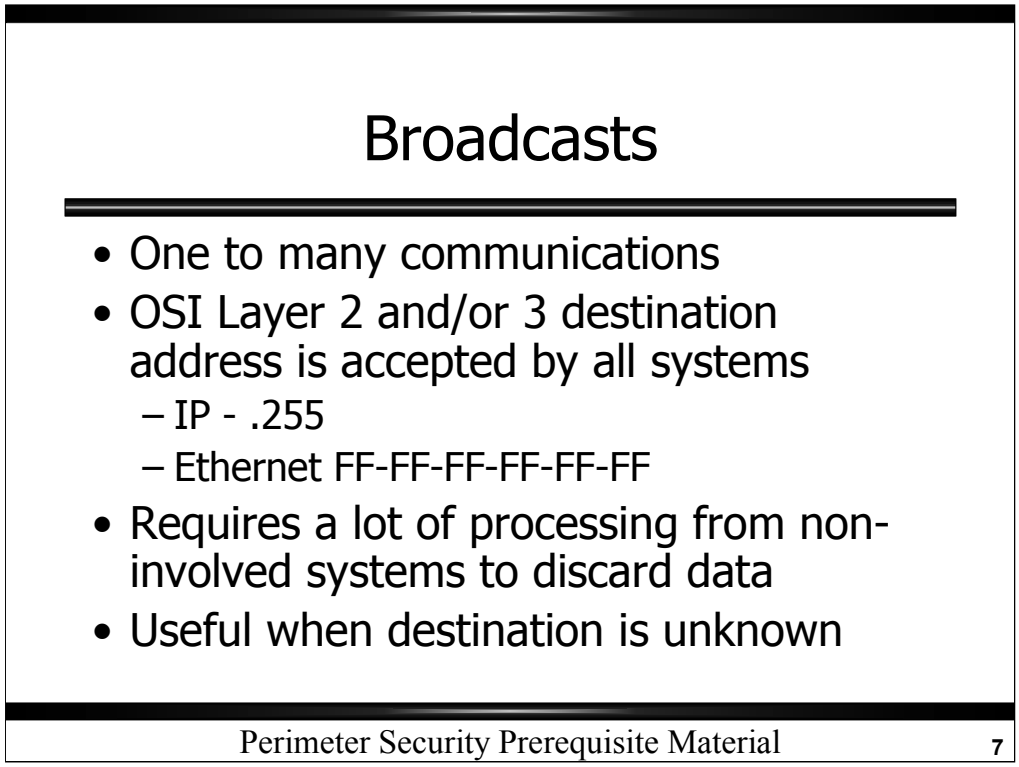


The gurus who defined how networked systems should communicate actually created a few different methods of exchanging information. Each has its own strengths and weaknesses and is applicable only during certain kinds of communication sessions. For example, if the conversation is between two systems, unicast may be the best choice. If communication is to a local system but you don't know its address, broadcast is a better choice. If you need to send data to multiple systems at the same time, multicast is the way to go.

Unicast Communications

- Point to point communications between two systems
- Destination address is a single system
  - At both Open Systems Interconnection (OSI) layer 2 and 3
- Minimal processing from non-involved systems to discard data
- Low noise but doesn't scale well

Unicast is the most common method of communication used in networking. When two systems are exchanging data, they will usually use unicast transmissions. This allows all other hosts to ignore the conversation and go about their business. One small bump with unicast is that the systems first need to know how to find each other before the conversation can begin. This means that most unicast conversations start off with one or more broadcasts.



Broadcasts are the common method of communication when the location of a remote machine is unknown. For example, if I need to talk to "Bob" but I do not know his network address, I may broadcast a query for Bob to tell me where he is located. Once Bob is found, I can then switch over to unicast in order to communicate more efficiently.

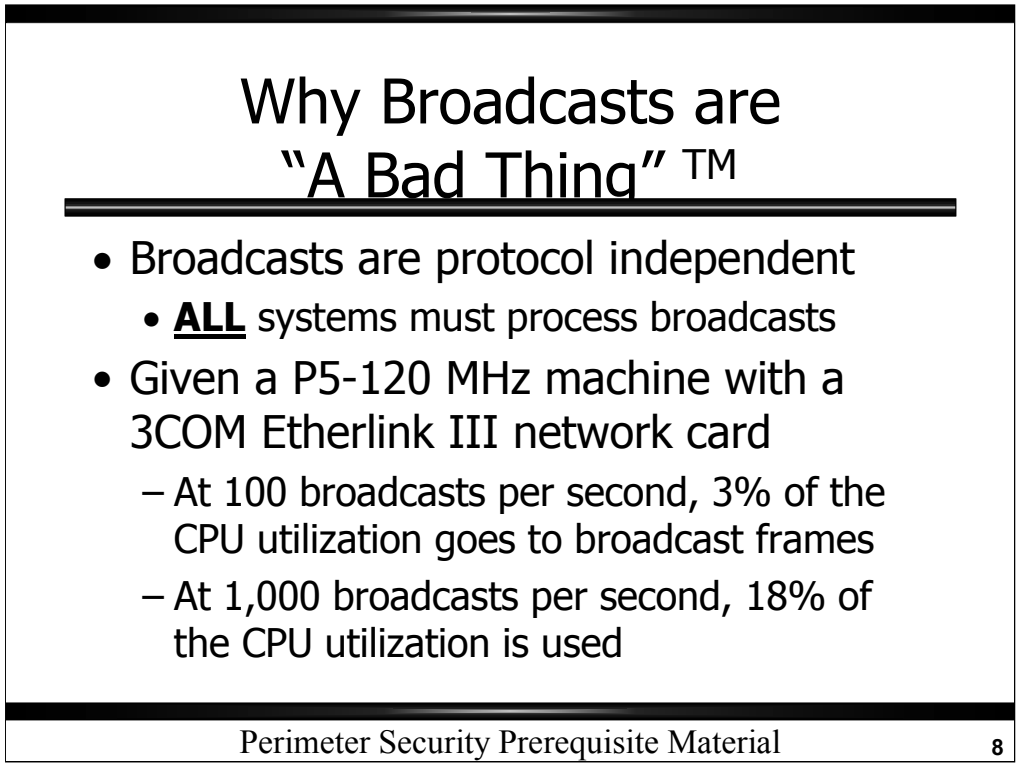
Some common broadcast based protocols:

Routing Information Protocol (RIP) on IP and Internet Packet Exchange (IPX)

Bootp & Dynamic Host Configuration Protocol (DHCP)

Network Basic Input/Output System (NetBIOS) name advertisements and name claims

Approximately 75% of all AppleTalk function calls



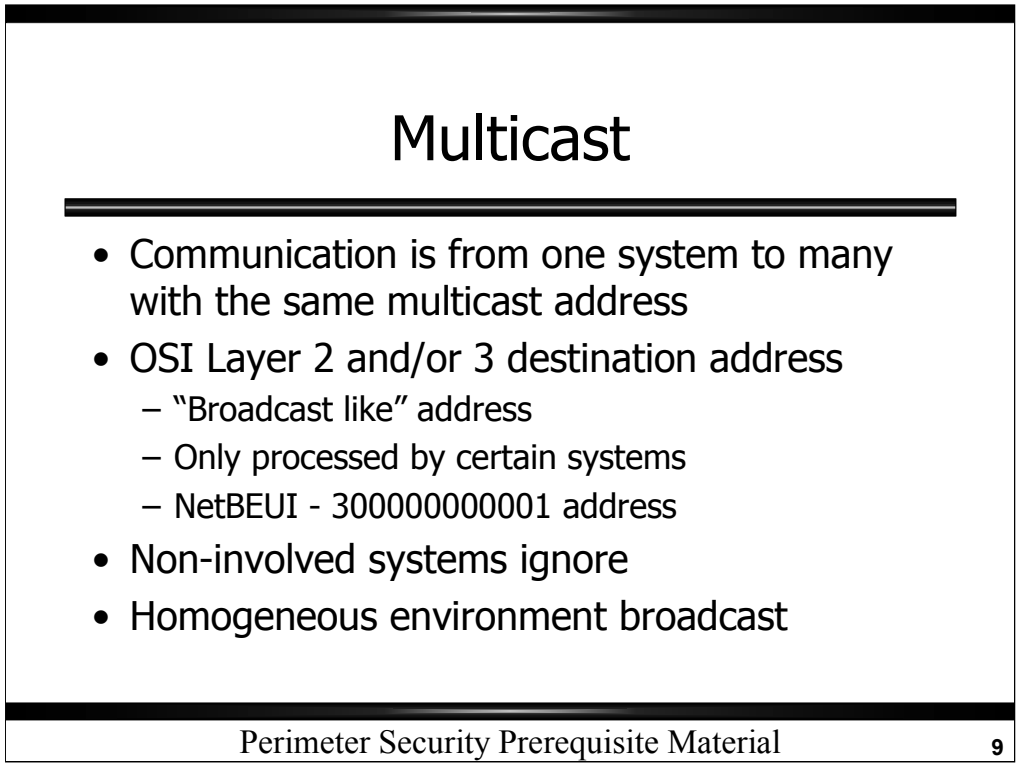
The slide above says it all. When a system reads in a frame from the network it needs to expend central processing unit (CPU) cycles to decode what the frame has to say. Since all systems must read in broadcasts, a broadcast can waste CPU time on every system on the wire. One of the best things you can do to improve network performance is take control of the number of broadcasts that are being transmitted.

It's important to note that network broadcasts get mapped to the Ethernet level. This means that all systems, regardless of the network protocol they are using, see all broadcasts.

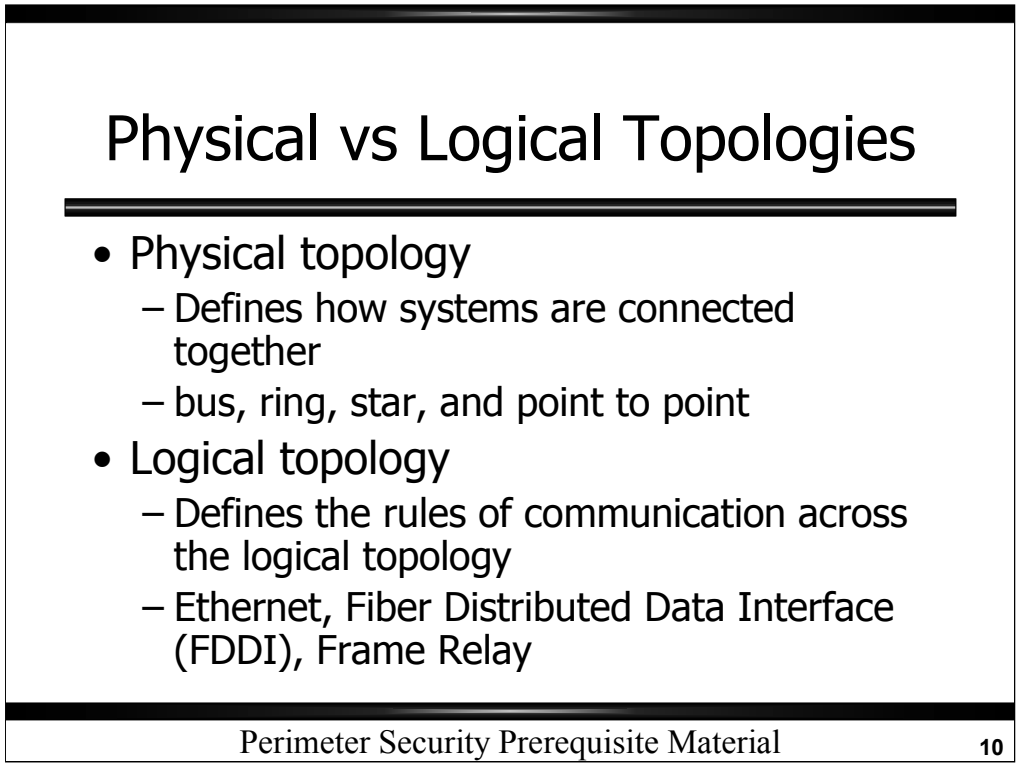
For example, let's say I have two Mac system on the wire that are using AppleTalk to communicate. Each system is generating broadcasts every 30 seconds. Let's also assume that I have 50 Windows system on the same network segment which communicate via IP.

Since the AppleTalk broadcasts get mapped to an Ethernet broadcast address, every Windows system will need to process this traffic. For each AppleTalk broadcast each Windows system will see that the frame has an Ethernet broadcast address and buffer the frame into memory. The system will then perform a cyclic redundancy check (CRC) check to ensure the frame is valid. Next, the system will strip off the Ethernet frame and pass the packet up to the networking layer.

At this point the packet will be discarded but we've already spent quite a bit of CPU time dealing with a packet we really didn't need to see.



Mix together unicast and broadcast transmissions and you get multicast. Multicast is a bandwidth efficient method of "broadcasting" to only specific systems. Typically this is done by using a special multicast address. Systems which are part of the same multicast group listen on this address and handle any transmissions they see in a similar fashion to a broadcast. Systems which are not part of the multicast group ignore the transmission. Multicast addressing can be done at the hardware (OSI layer 2) or software (OSI layer 3) level.

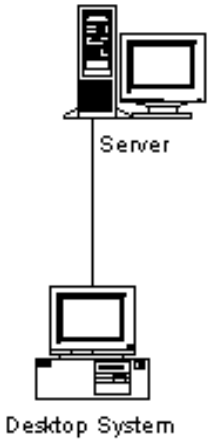


There are two types of topologies, physical and logical. A physical topology describes the way the network is wired together. This is independent of the logical topology which describes the communication rules which are to be used when systems exchange data on the logical topology.

Physical and logical topologies are independent of each other and you can actually mix and match. For example a logical topology can be wired using different physical topologies.

Point to Point

- Single connection between two networked systems
  - Commonly used for WAN connections
  - Can be used for home networks
  - May need cross cable for systems to communicate



Perimeter Security Prerequisite Material 11

Point to point is the most simple wiring configuration. String two networked systems together and let them talk to one another. The problem is this topology does not scale, as there cannot be more than two systems wired together. Hence, point to point is typically only used for WAN connectivity when only two devices need to be connected. Of course the two devices may be acting as doorways and allowing multiple systems to communicate through them. A good example is two routers connected by a WAN link.

Here's a tip for home networking people. If you want to directly wire together two systems which both have an Ethernet card installed, you need to modify the cable. On one end of the cable you need to switch the wiring on the following pins:

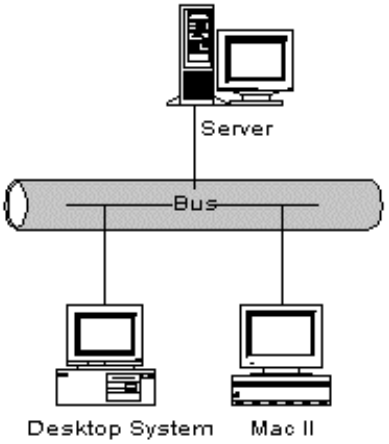
1<- - >3

2<- - >6

This is called a cross cable and insures that "transmit" on one side is connected to "receive" on the other.

Bus Topology

- All systems connect to the same segment of wire
  - Poor scalability
  - Poor traffic isolation
  - Low fault tolerance
  - Troubleshooting nightmare

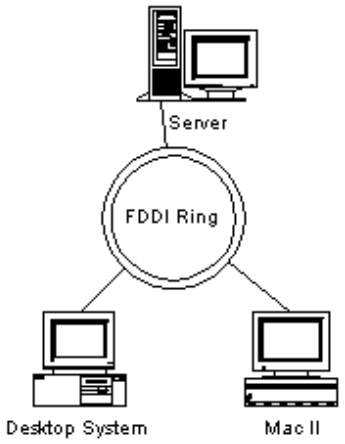


Perimeter Security Prerequisite Material 12

This topology is dated and used very little today. The slide describes all the reasons why bus topology has gone the road of the pet rock. Bus had its place in the days when hubs cost $150 per port. Today, many network card vendors do not even support bus as it sees very little use on the modern network.

Ring Topology

- Multiple point to point connections forming a ring
- Systems transmit on one side and receive on another
  - Dual ring can provide fault tolerance



Perimeter Security Prerequisite Material 13

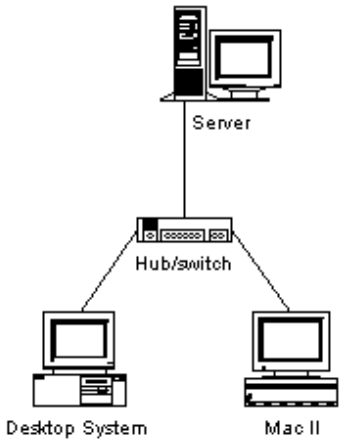
Think of a bunch of kids standing in a circle playing the telephone game and you'll get a good idea of how ring topology works. Each station transmits on one side while receiving on the other. So if I have something to say to a person on the other side of the ring I tell the person on my left and they pass it along. Eventually, I will get a reply to my message from the person standing on my right.

There are two major reasons why you do not see ring topology in wide use today. The first is that it is not supported by Ethernet. Since Ethernet came out on top in the logical topology wars, people don't use rings because you can't run Ethernet on them. The other major factor was cost. Each system's network card acts like a repeater (discussed in greater detail later) which increases the cost of hardware.

If you run two sets of rings you can increase the fault tolerance of the circuit. This was used by FDDI to increase network availability. For example if the MAC II in the slide dies a fiery death, the two other systems could isolate the MAC by looping back on the second ring. This would allow the network to remain operational.

Star Topology

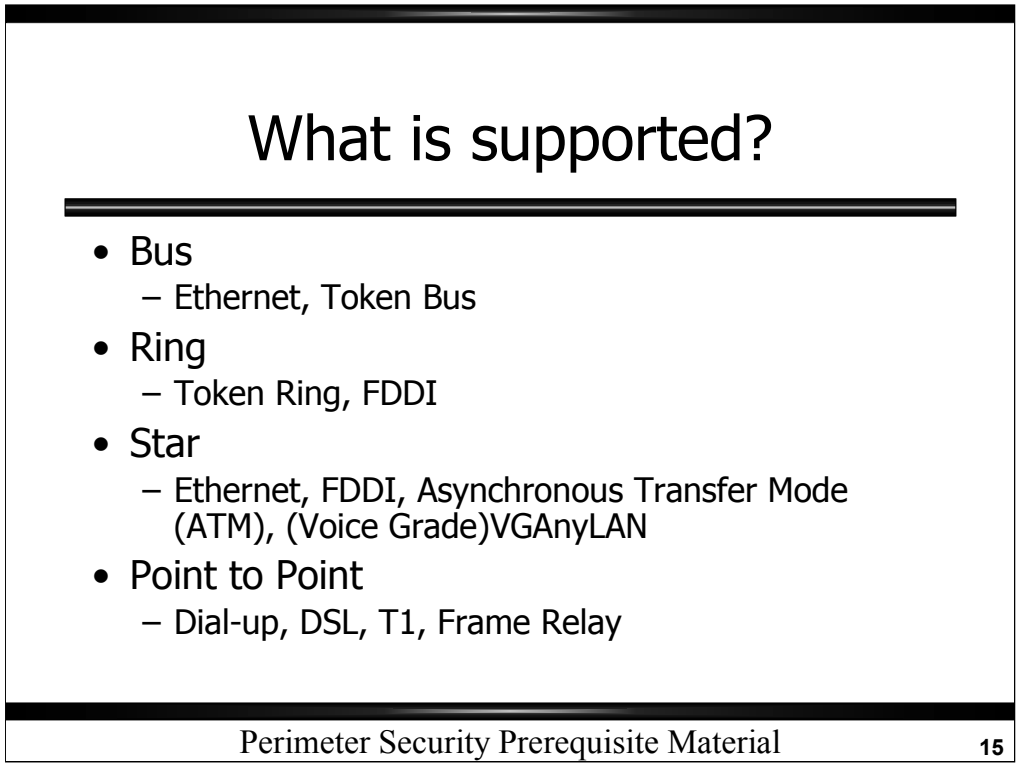
- Multiple point to point connections to a central device (hub or switch)
  - Good fault tolerance
  - Certain hardware can provide traffic isolation
  - Scales well



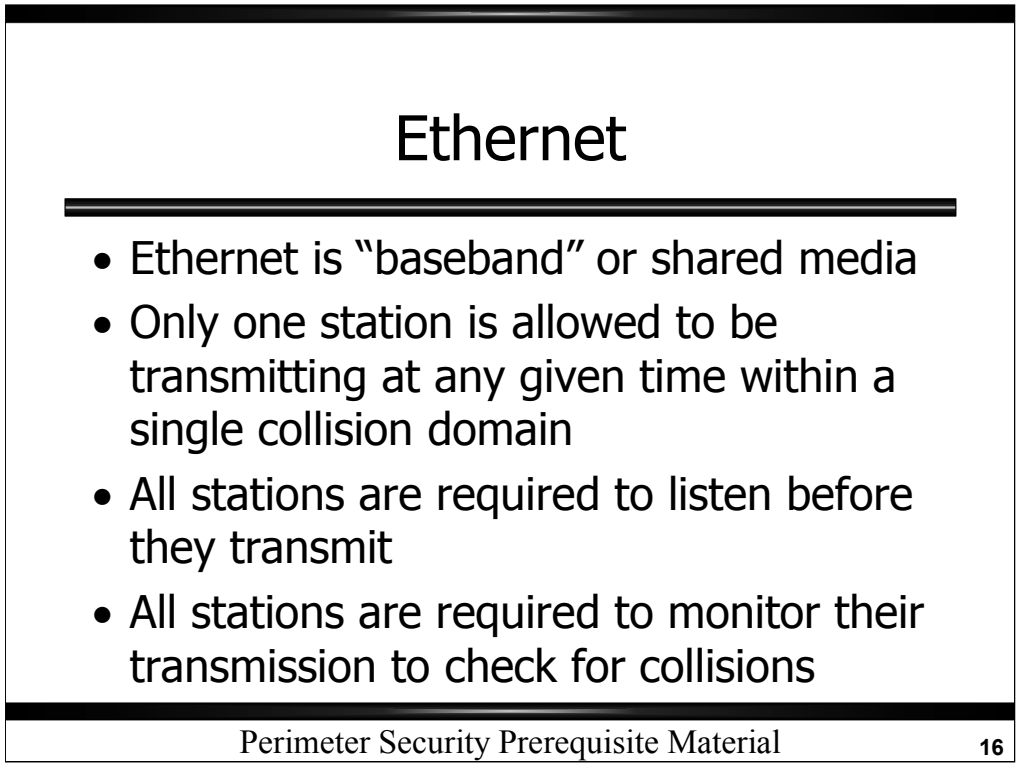
Perimeter Security Prerequisite Material 14

Star is the most common physical topology used today. With a star topology, all system are connected together though a central device such as a hub or a switch. While the central device is a central point of failure, star is usually resilient enough to deal with any one circuit or system failing. For example if the cable leading to the desktop system is cut, the MAC and the server would be unaffected.

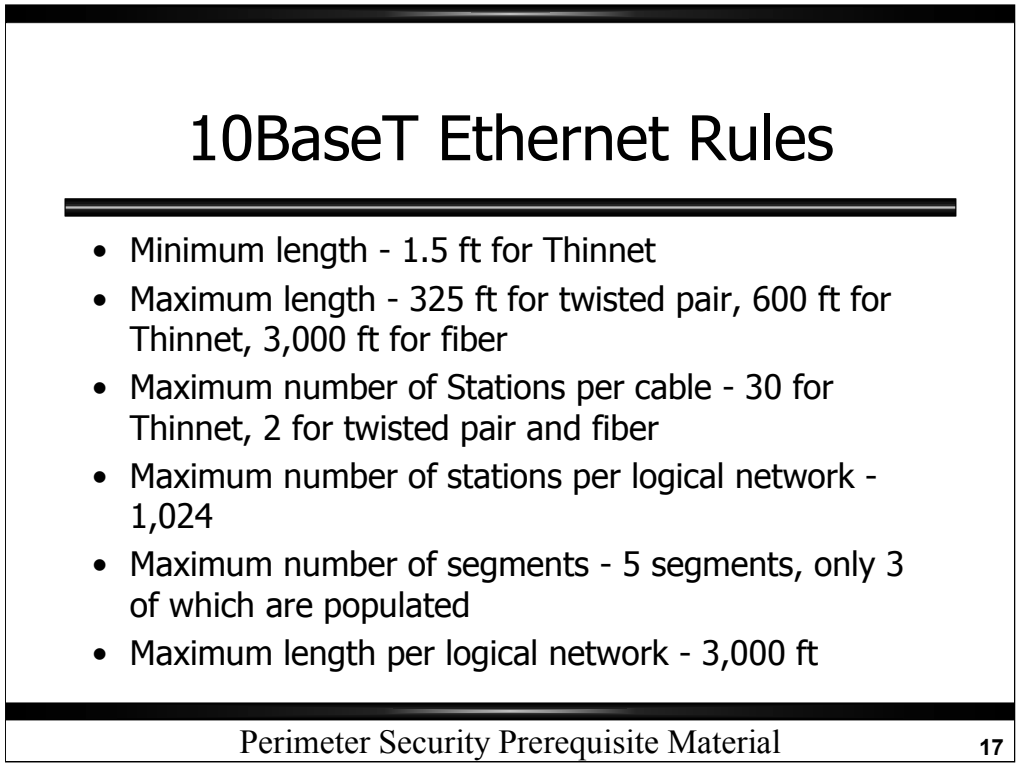
Traffic control is also improved. Since all circuits are tied to a single device I can build intelligence into that device in order to control traffic flow on my network.



As mentioned not all physical topologies are supported by a logical topology. For example Ethernet can be wired as a bus or a star but not as a ring. Ethernet can be wired as a point to point but this will only support two systems.



Ethernet is by far the most popular logical topology. Ethernet is cost effective and scales fairly well. While Ethernet itself is not fault tolerant, other technologies can be leveraged to increase Ethernet's availability.

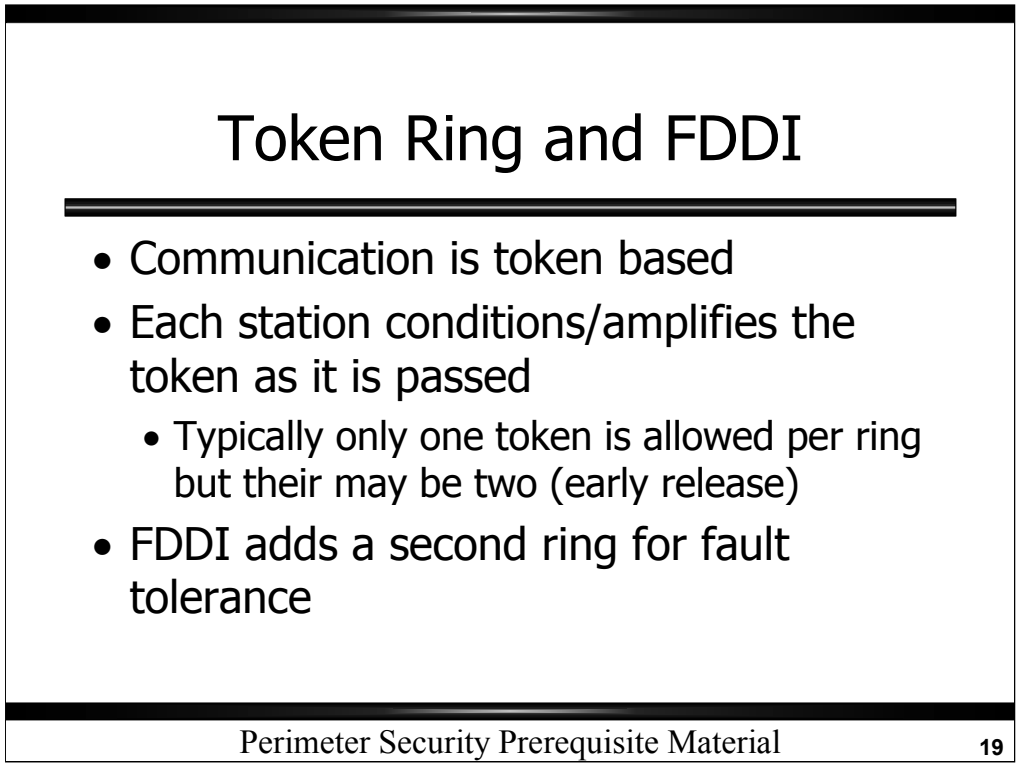


When wiring a 10BaseT Ethernet network, here are the guidelines to follow.

100BaseTx Ethernet Rules

- Minimum length - none
- Maximum length - 325 ft for twisted pair, 3,000 ft for fiber
- Maximum number of Stations per cable - 2 for twisted pair and fiber
- Maximum number of stations per logical network - 1,024
- Maximum number of segments - 3 segments, only 2 of which are populated
- Maximum length per logical network - 3,000 ft

These are the rules for wiring up a 100BTx network.

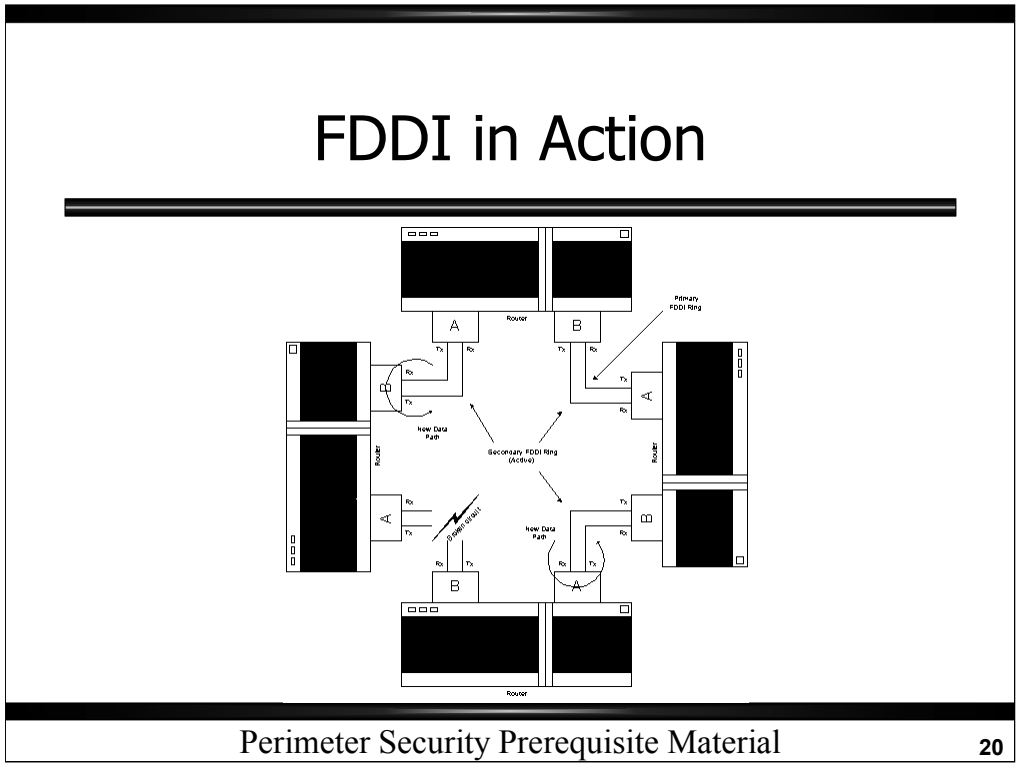


Token Ring and FDDI have a certain level of fault tolerance built right in. When a Station does not see a token within a specific window of time, it will begin to send a beacon. A beacon is simply the stations way of saying "hey I have not seen the token in a while, there may be a problem down stream from my location". This allows systems on the network to automatically isolate the problem area and attempt to take some form of corrective action.

For example in a token ring environment, a system which is identified as having a problem, will pull itself out of the ring and perform a self diagnostic. If it finds a problem, the system will remain out of the ring allowing other systems to continue communicating normally. If the self check passes, the station jumps back into the ring.

One of the nice things about FDDI is that it has a fail safe against this self check. Think about this for a moment, a card that may be faulty is running a check to see if it is faulty. Does this sound like a good idea? In the case of FDDI there is a backup plan. If the upstream and downstream systems realize that the ring fails every time the faulty system jumps in, they can take it upon themselves to isolate the offending system. This greatly increases the availability of the network.

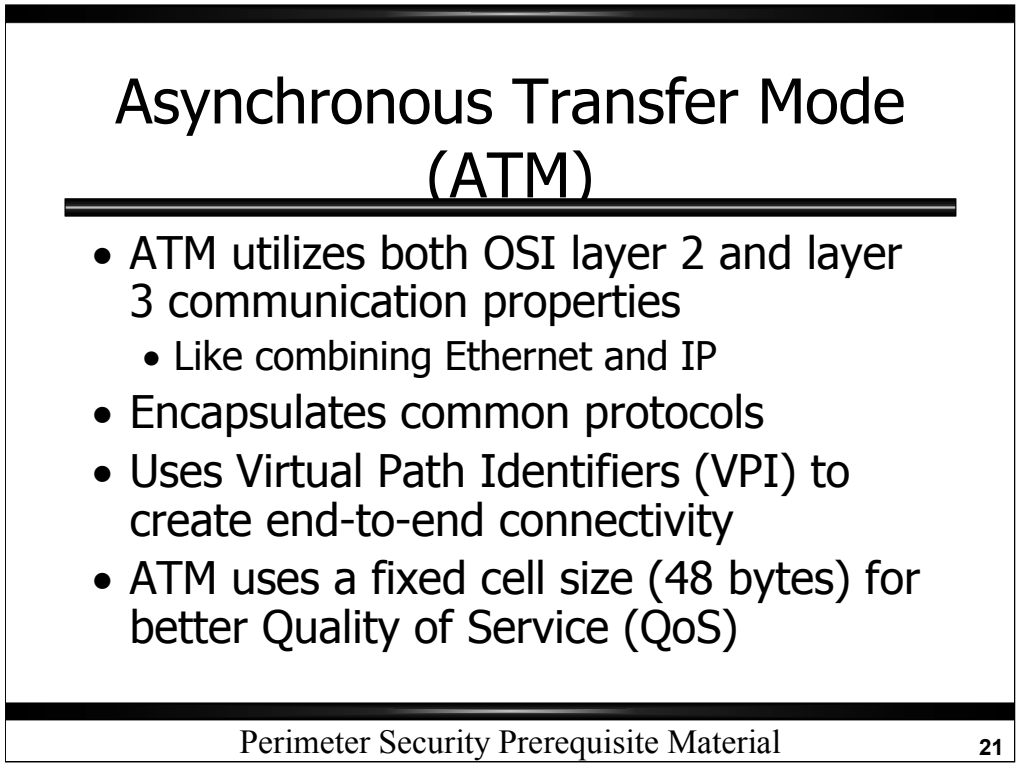
There is a strong push away from Ring technology. Ring's high cost and the stability of Ethernet has lead to it's demise over the last few years. Some people are however still using FDDI for its high availability functionality.



This slide shows FDDI in action. Note that each FDDI device has two connections, one labeled “A” and the other “B”. Each of these connectors has both a transmit and a receive port. When “A” and “B” are connected around the ring, two transmission circuits are created. During normal communications, only the primary ring gets used. Remember that FDDI communications are token based so there is a token spinning around the ring in a clockwise direction carrying data from one system to another.

Now, let’s say that one of our circuits gets broken. When the left hand system realizes that the token is late in arriving, it transmits a diagnostic packet called a *beacon*. The beacon lets the other systems know that a problem has been detected. This allows the systems to dynamically activate the second ring in order to circumvent the fault area. This fault could be a bad cable or even a faulty system.

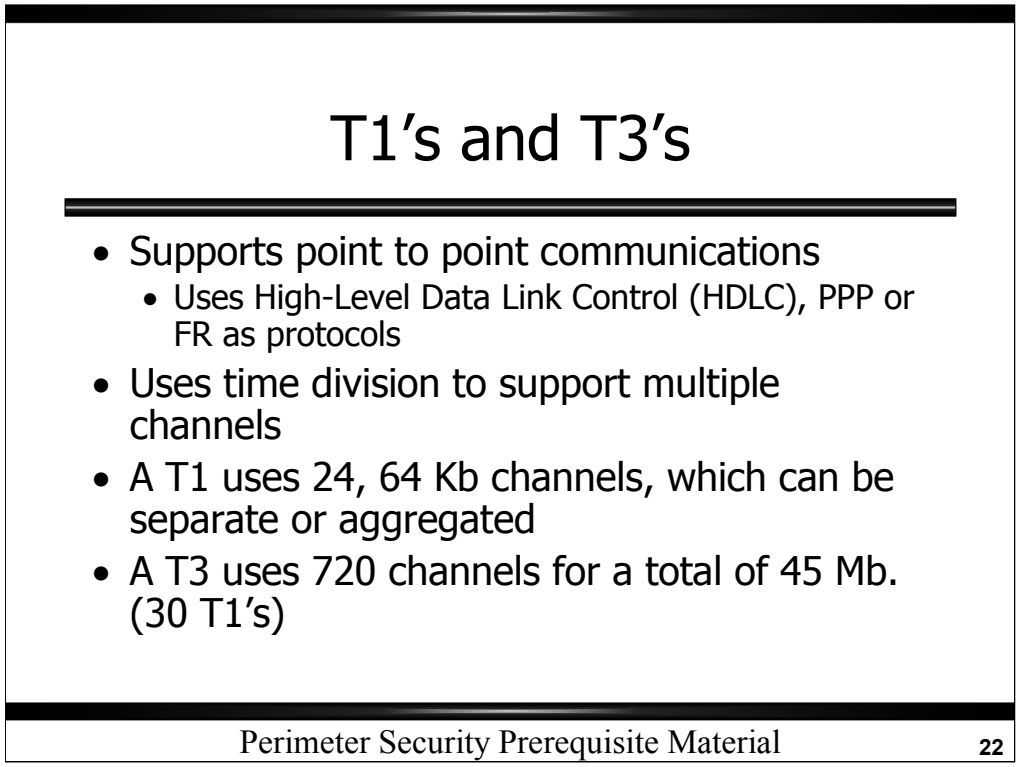
Two caveats to watch. First, watch the distances. As you can see in this slide cable distance increases 2X when a fault occurs. Second, modern FDDI devices support full duplex mode which doubles the communication speed by running the secondary ring full time. The trade off is that the circuit is no longer fault tolerant.



ATM is a wee bit of a strange beast. It has properties that make it a logical topology like Ethernet or FDDI as well as properties that make it a protocol like IP or IPX. Unless you have a pure ATM environment, all traffic must be encapsulated.

ATM stations create permanent virtual circuits (PVC) and switched virtual circuits (SVC) across shared media. Think of making a phone call and you'll get the idea. When you dial a number the phone company sets up a circuit from your phone to the phone you just dialed. When the call is complete, the circuit is torn down. This is useful because the network can provide a quality guarantee for your traffic. For example when the VPI sets up the PVC it can state to each switch along the way "I need 512 Kb/s to support a video conference, do you have that much bandwidth available?" If the answer is yes the PVC is setup through the ATM switch. If the answer is no than the VPI will attempt to create a path through some other switch.

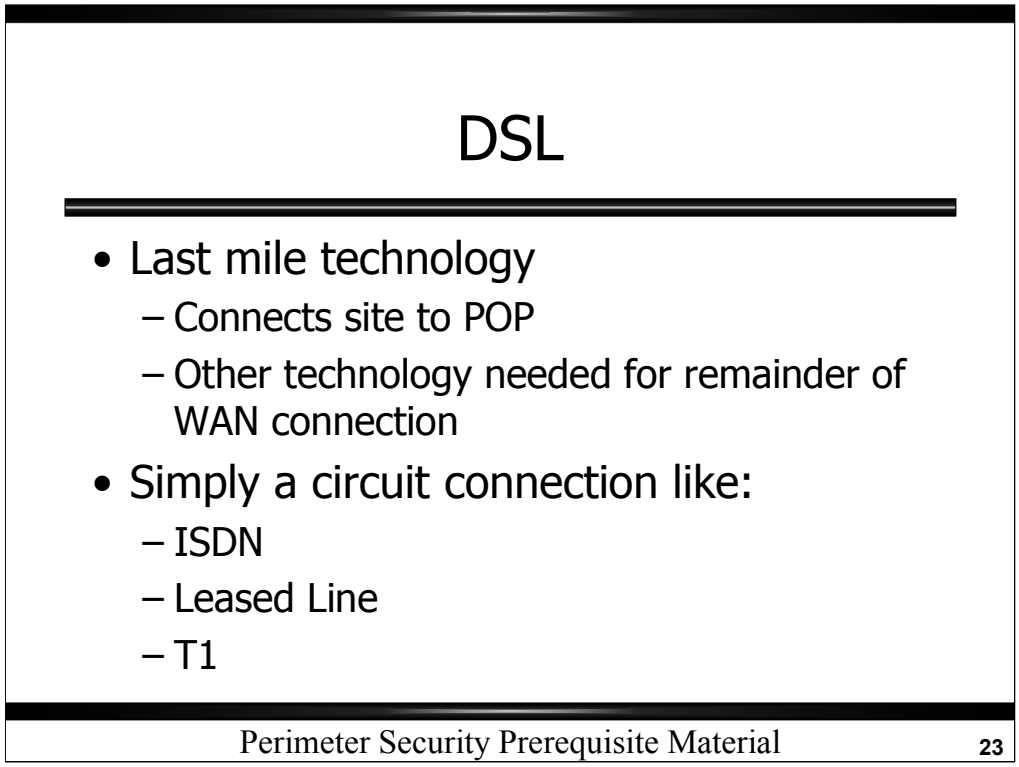
There has been a lot of talk of ATM's speed, this is due to pipe size rather than efficiency. As we will discuss later ATM's runty cell size kills potential transfer rates when working with raw data. I've personally found that when passing typical network data, a 100 Mb Ethernet network will greatly out perform a 155 Mb ATM network at about 1/4 the price.



Digital transmission rate 1 or T1's and digital transmission rate 3 or T3's are the transport which allows you to connect distant networks together via a WAN. They simply provide the physical connection so actual communications rely on some other protocol. For example if you have a T1 that goes from one site to another, you would probably use HDLC or PPP to communicate across this circuit. If the T1 is simply to wire you to your local central office (CO) so you can connect into a frame cloud, you would want to use the Frame Relay protocol to communicate.

T1's and T3's use time division which creates multiple communication channels. These channels can be aggregated to increase available bandwidth or used separately for different purposes.

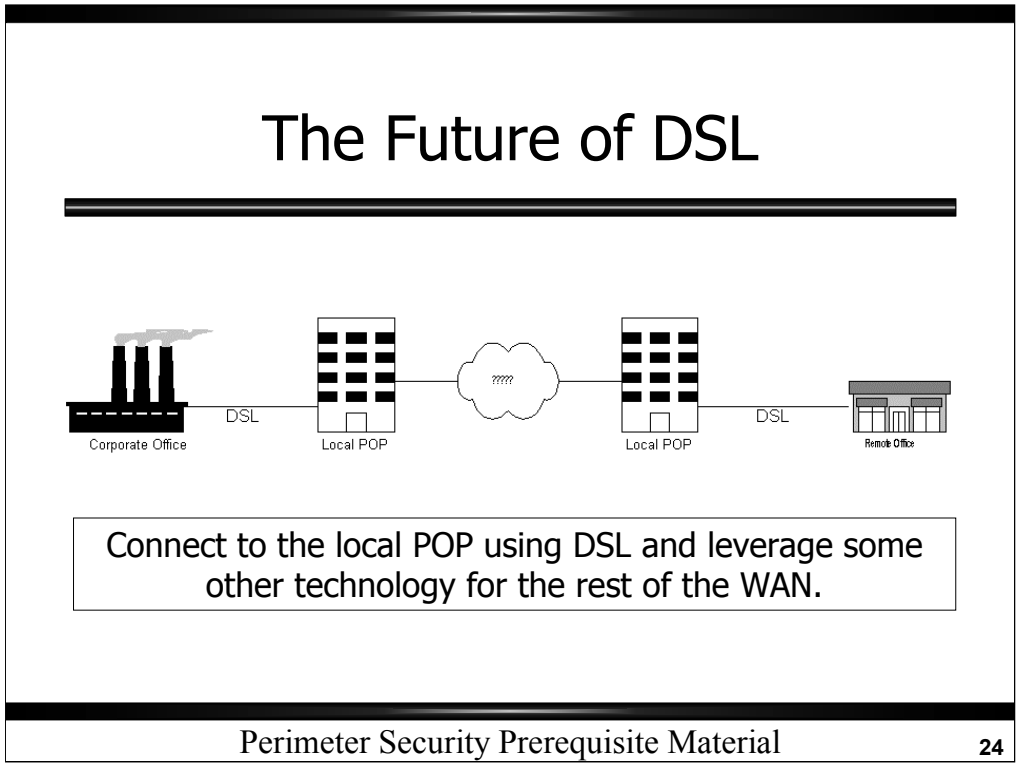
For example let's say I have two sites connect with a T1. I could choose to combine the first eight channels to pass network data between the two sites. This would give me 512 Kb of available bandwidth. I could then use six channels for voice by using them to create six separate circuits connecting my two private branch exchange (PBX)'s together. This would allow my organization to conduct six separate voice calls between the two sites at the same time. The remaining ten channels could then be left unused until they are needed.



DSL is an acronym for Digital Subscriber Line technology. It is used to connect a customer site to the local telephone company's Point Of Presence (POP) or Central Office (CO). The term "last mile technology" is used to describe DSL as it is only used for the connection between the connecting site and the POP.

DSL is currently being targeted at small offices and/or home offices (SOHO) for Internet connectivity. Its biggest competitor is considered to be integrated services digital network (ISDN) and cable modems. A DSL circuit can be used to replace ISDN, a leased line or even a T1. DSL is deployed at speeds ranging from 56Kb to 50 Mb.

The term DSL is actually a combination of a number of standards. This is why you may see it referred to as "xDSL" from time to time as each of the different standards uses a different first letter. Asymmetrical Digital Subscriber Line (ADSL) is the most widely implemented standard. Very High Bit Rate Digital Subscriber Line (VDSL) is new but gaining in popularity.

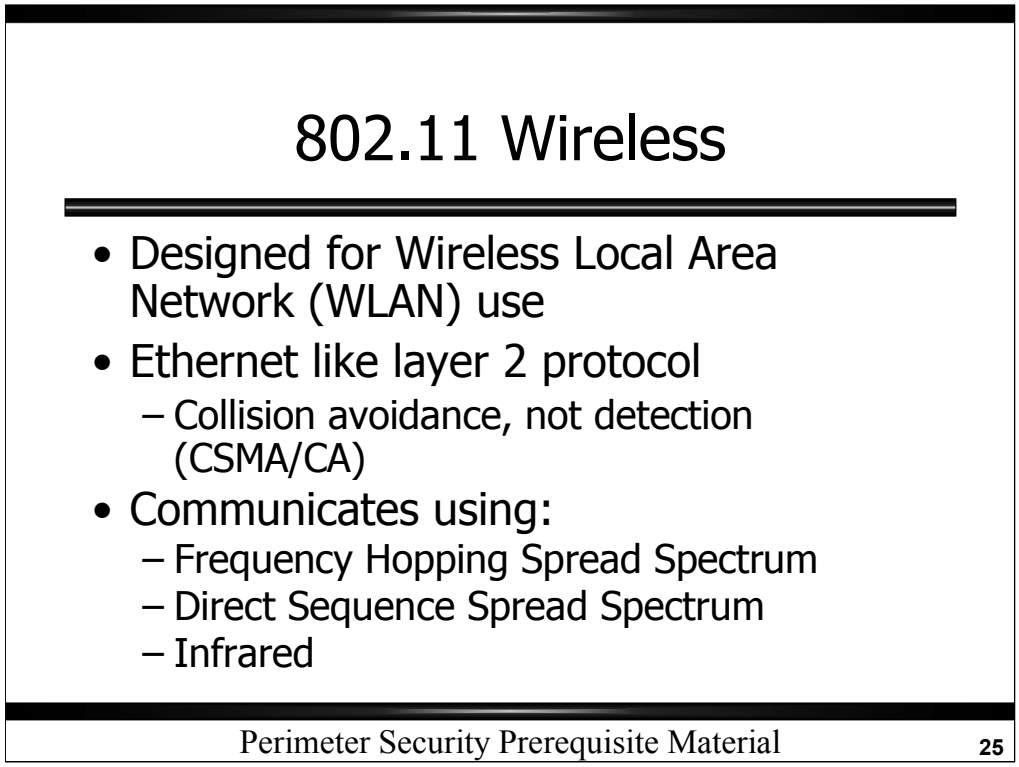


The slide shows an example configuration of how we may end up using DSL in the years to come. DSL is used to tie each end of the connection to the local POP. The connection between POPs needs to be supplied through some other technology (Frame Relay, T1, vendor network, etc.). This could be a third party network which is not subjected to government tariffs (such as Vitts). The result is WAN connectivity at a fraction of the cost you see today.

Note that you are not required to use DSL on both ends. The example in our slide the corporate office could be using a T1 or T3 to connect to its local POP while the remote office uses DSL.

Note that DSL is still in its infancy so other applications for DSL will be targeted once the offering matures. While DSL is not really being targeted at medium to large business as of yet, the potential exists.

The biggest limitation of DSL is availability. This is a factor of POP equipment as well as distance limitations. Typically to have any chance of deploying DSL you must be within three miles of a POP or CO. Also, the further away you are located from the POP or CO, the slower the connection speed will be.



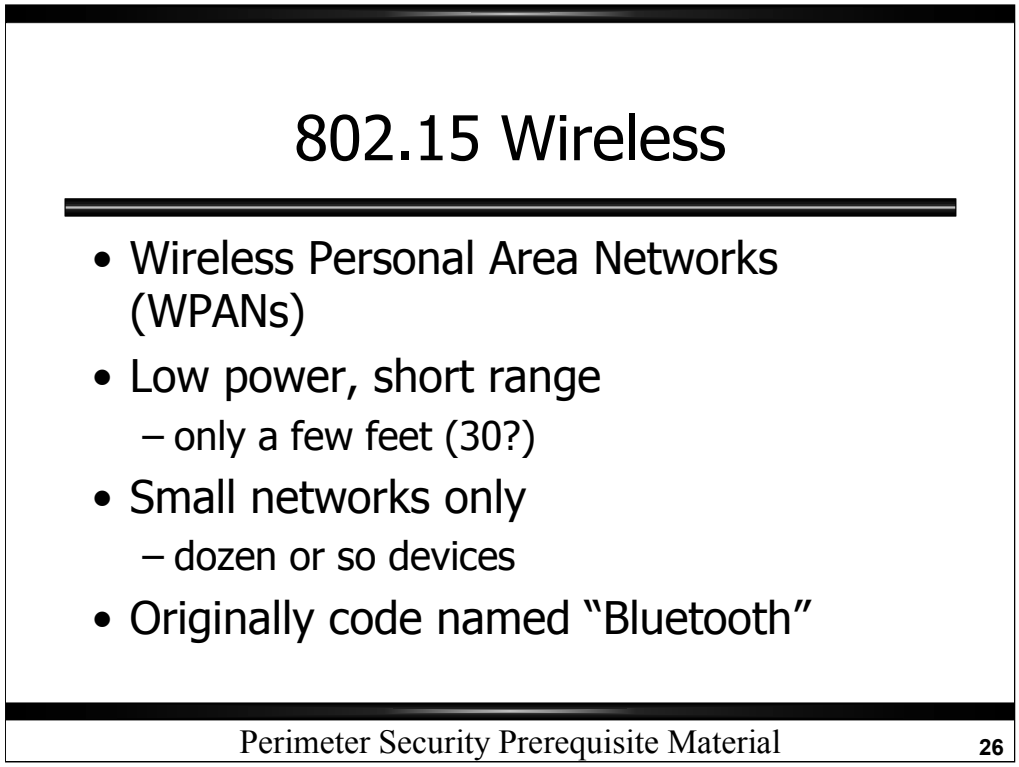
The driving force behind the adoption of wireless networking is it's flexibility. Need to walk down the hall to participate in a meeting? Simply carry down your laptop and keep working away. Many people would be lost today without their cellular phone. The same type of functionality is now being adopted for use on the network.

802.11 is the specification for use of wireless devices in a local area network type of environment. Computers are outfitted with a wireless transmitter/receiver (usually a PC card) and communicate with a base station that is hard wired to the network. The base station functions similar to a network bridge or switch.

Power is usually minimal at 1 Watt which limits the reception range to 100 ft or so. Range is greatly dependent on wall and floor thickness as well as what kind of devices may be generating noise in the area.

Bandwidth is usually limited to 1 to 2 Mb although the specification outlines speeds of 11 Mb in at 2.4 GHz. Jump up to a 5 GHz unit, and bandwidth increases to 54 Mb but at the cost of support for shorter distances. The higher frequency has a more difficult time penetrating solid objects.

Wireless can also be a capacity planning and security nightmare. For example what if everyone is in the same meeting running through the same base station? That system that broke into the HR server is within 100 ft of the base station, but where exactly is it located?



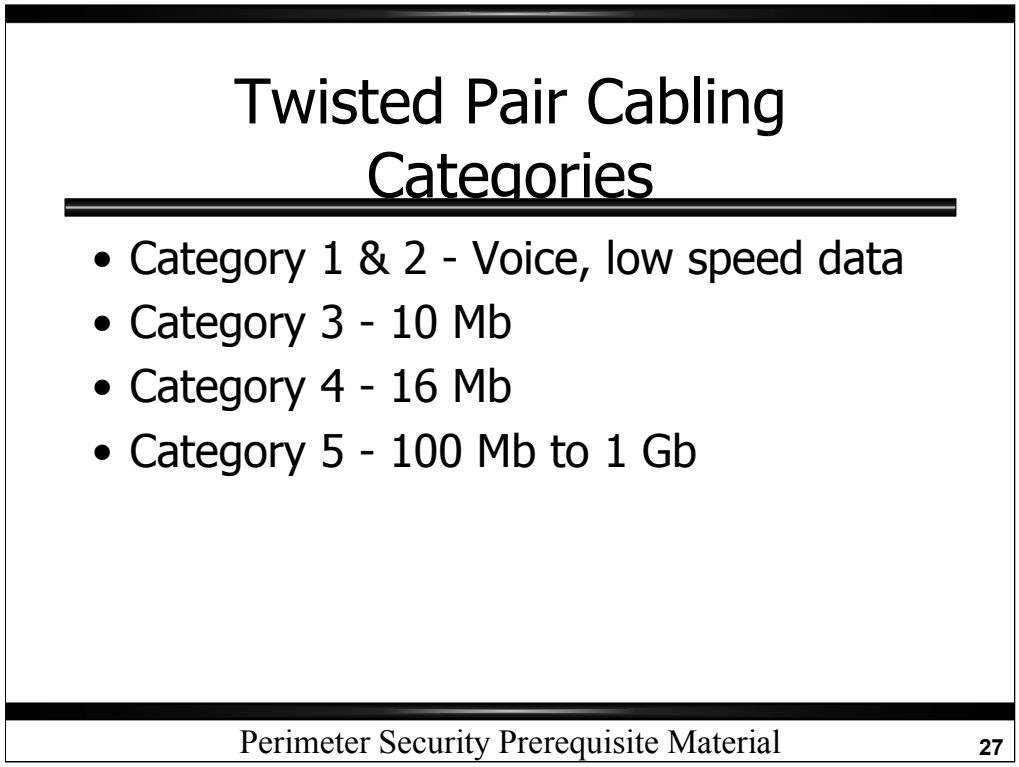
The 802.15 specification is still very much under development but has the possibility of moving us into a truly "completely networked" world.

The idea behind 802.15 is to allow devices within a close physical proximity to create an ad hoc network and exchange information. This could be as basic as letting your PIM talk to your laptop or as esoteric as smart cards which identify you to all devices within your proximity. Think about walking up to your car and having the door open because the car knows it's you, the seat adjusts to your requested position, the climate control adjusts to your perfect preference and the CD player loads up that favorite "morning driving CD". You get the idea. 802.15 has the potential of bringing networking to a whole new level.

Current development work is attempting to provide 802.15 with at least 20 Mb worth of bandwidth. This should be sufficient for modern day requirements for only a couple of devices. Expect to see this pushed up over time however.

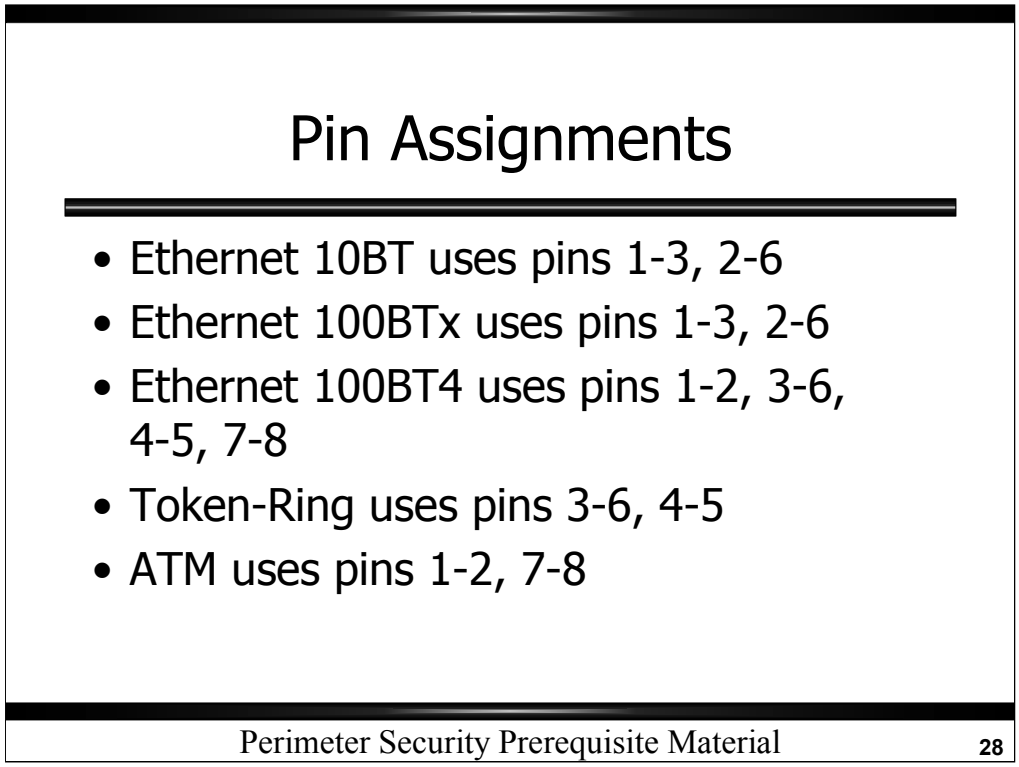
One of the problems with 802.15 is conflicts with 802.11. Both currently use the same 2.4 GHz frequency. This can cause interference problems between these devices.

So why not just change the frequency? Part of the reason is that you really need these devices to be compatible. For example you may want your PIM to use 802.11 for cruising the Internet but may also want your PIM to be able to communicate with your stove using 802.15. Certain devices need to be able to speak using both specifications.



Cabling, particularly twisted pair, is designed to meet the criteria of a specific category. The supported category indicates what level of bandwidth can be pushed through the cable without error. While you can use category 3 (CAT3) cabling on a 100 Mb network you would probably end up having intermittent communication problems and failures.

Remember that your cabling is only as good as your weakest link. For example I've seen environments using CAT5 cables and connectors but the punch downs were only rated for CAT3. The best way to verify compliance is to use a cable tester and verify the entire circuit, including patch cables.



This slide shows the pin outs for some common network cabling. Note that 10BT and 100BTx Ethernet only uses two of the four wire pairs found in CAT5 twisted pair cabling.

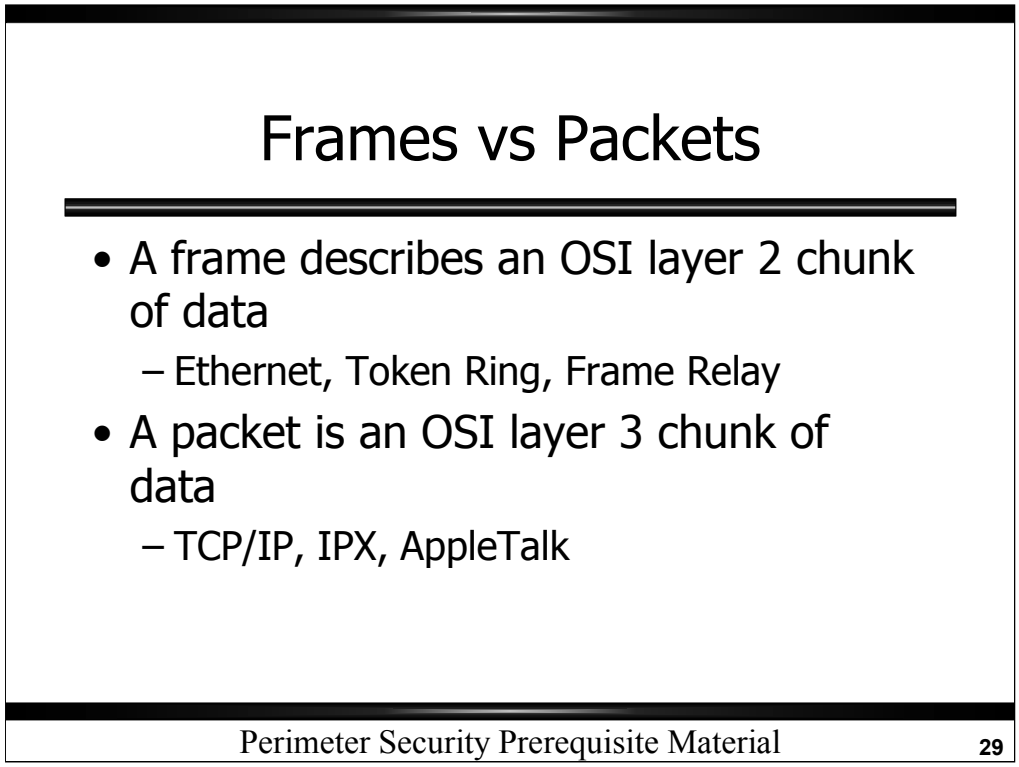
You may remember we stated that in order to make an Ethernet cross cable, you needed to switch pins 1 and 3 as well as 2 and 6 on one end of the circuit. Ethernet uses the following pin outs:

1 = Transmit +

2 = Transmit -

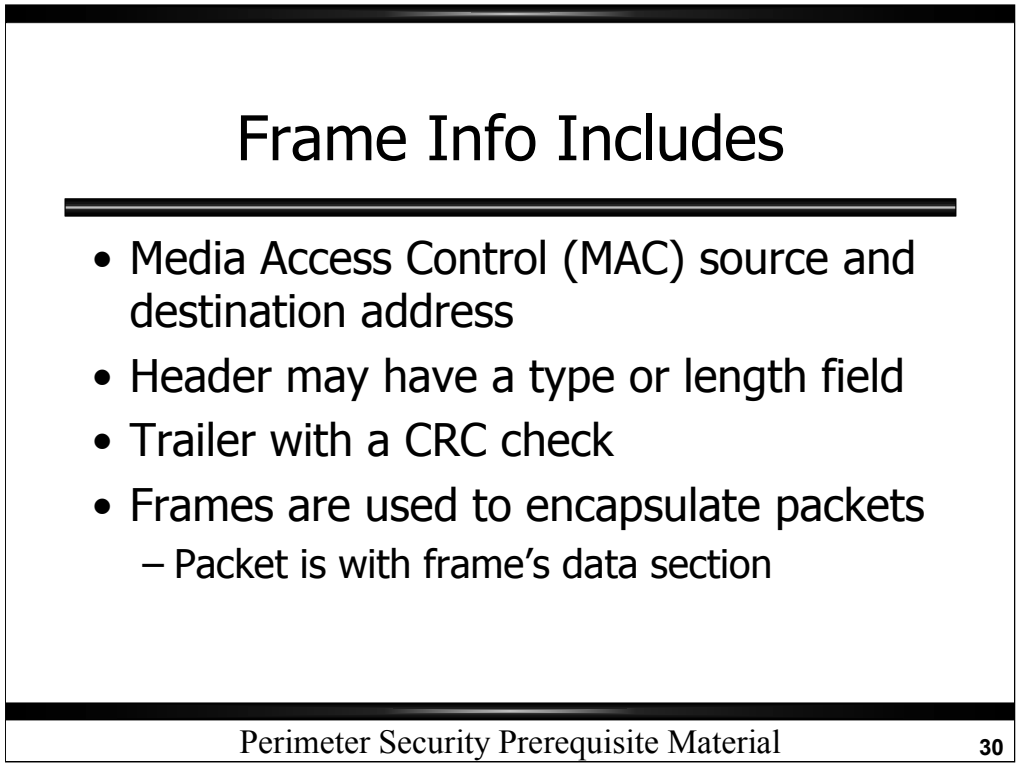
3 = Receive +

6 = Receive -



Just as a point of clarification, a frame refers to an OSI layer 2 chunk of data. For example when talking about a chunk of data which includes Ethernet delivery information, the proper term is “frame”.

A packet refers to an OSI layer 3 chunk of information. For example if you are looking at just the IP or IPX delivery information (i.e. without the Ethernet information) you would refer to the chunk as a “packet”.



A frame includes MAC source and destination information. At the end of a frame is a CRC trailer which verifies the integrity of the rest of the frame.

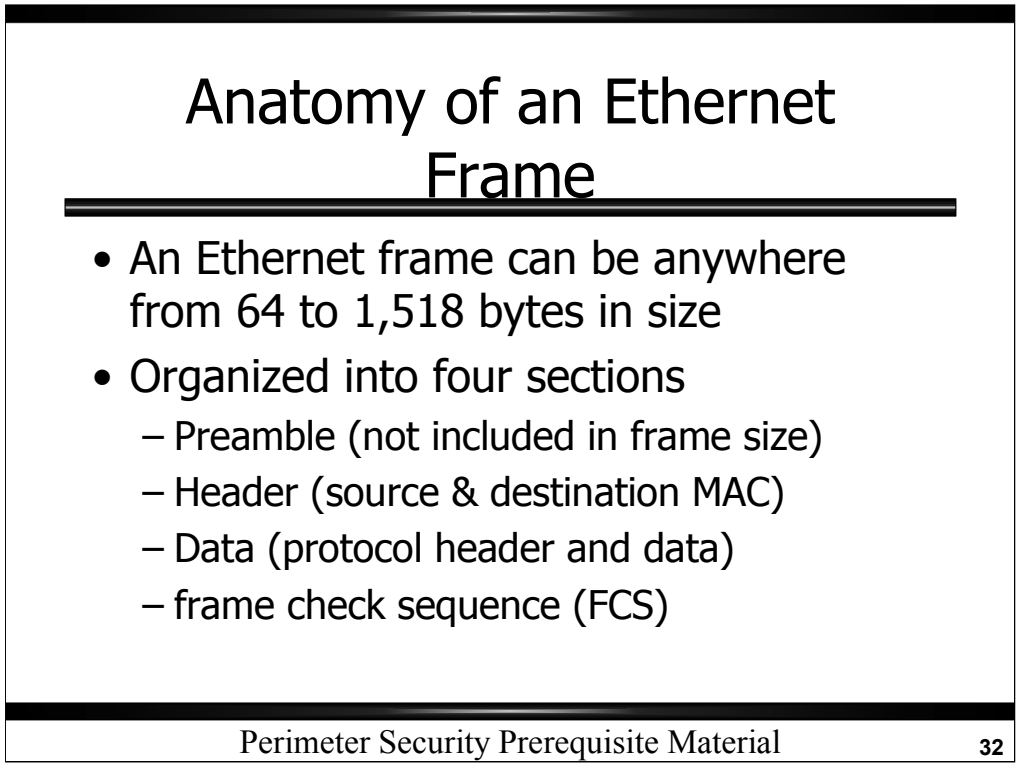
The data portion of the frame is actually an OSI layer 3 "packet". In other words, packets get encapsulated inside of frames.

Packet Info Includes

- OSI Layer 3 protocol header
  - Source and destination software address
  - time to live (ttl), CRC, ID for transport protocol
- OSI Layer 4 and up headers
  - Sequencing, translation, ports or sockets
- Actual data being transmitted

A Packet includes header information for OSI layers 3 and up. After the headers is the actual data being transmitted. A packet typically does not include a trailer.

As we will discuss in a later slide, the data portion of an Ethernet frame can be 46 - 1500 bytes in length. This means that some of this 46 - 1500 bytes gets lost to packet headers. The amount of bytes lost depends on the protocols being used. For example user datagram protocol (UDP) headers are smaller than transmission control protocol (TCP) headers so the amount of information that can be transferred with each UDP packet is slightly higher than TCP. This is smaller header on the UDP packet leaves more room for actual data.

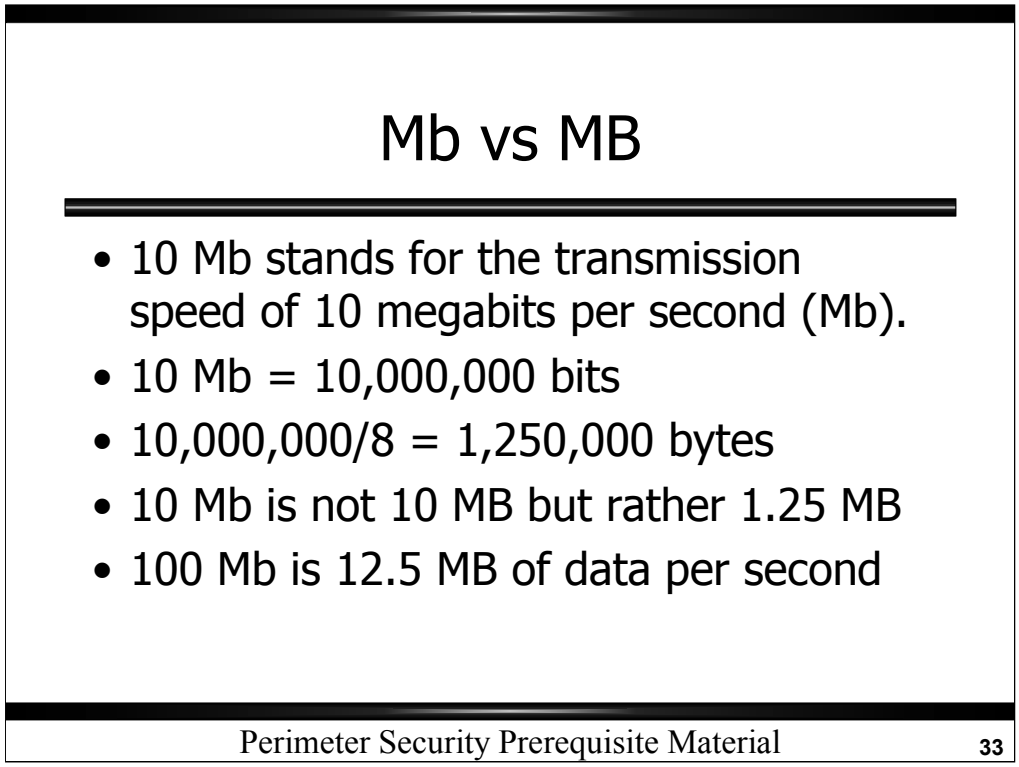


A *preamble* is a defined series of communication pulses that tells all receiving stations, "get ready—I've got something to say." The Ethernet preamble is always eight bytes long.The preamble is considered to be "overhead" in the communication process and is not included when measuring a frame's actual size. For example when we say an Ethernet frame is 64 bytes long, this does not include the 8 byte preamble.

An Ethernet header always contains information regarding who sent the frame and where they are trying to send it to. It may also contain other information, such as how many bytes the frame contains, or an indicator as to what type of Ethernet frame it is. For Example TCP/IP typically uses an Ethernet_II frame type. The Ethernet header size is always 14 bytes.

The data section of the frame contains the actual data the station needs to transmit as well as any protocol information such as source and destination IP address. As mentioned earlier, the data field can be anywhere from 46 to 1,500 bytes in size.

The frame check sequence is used to ensure that the data received is actually the data sent. The transmitting system processes the frame check sequence portion of the frame through an algorithm called a cyclic redundancy check or CRC. This CRC takes the values of the above fields and creates a 4-byte number. When the destination system receives the frame, it runs the same CRC and compares it to the value within this field.



A common mistake is that people confuse "megabits" (Mb) and megabytes (MB) when discussing topology speed. Remember there are eight bits to a byte so the conversion is:

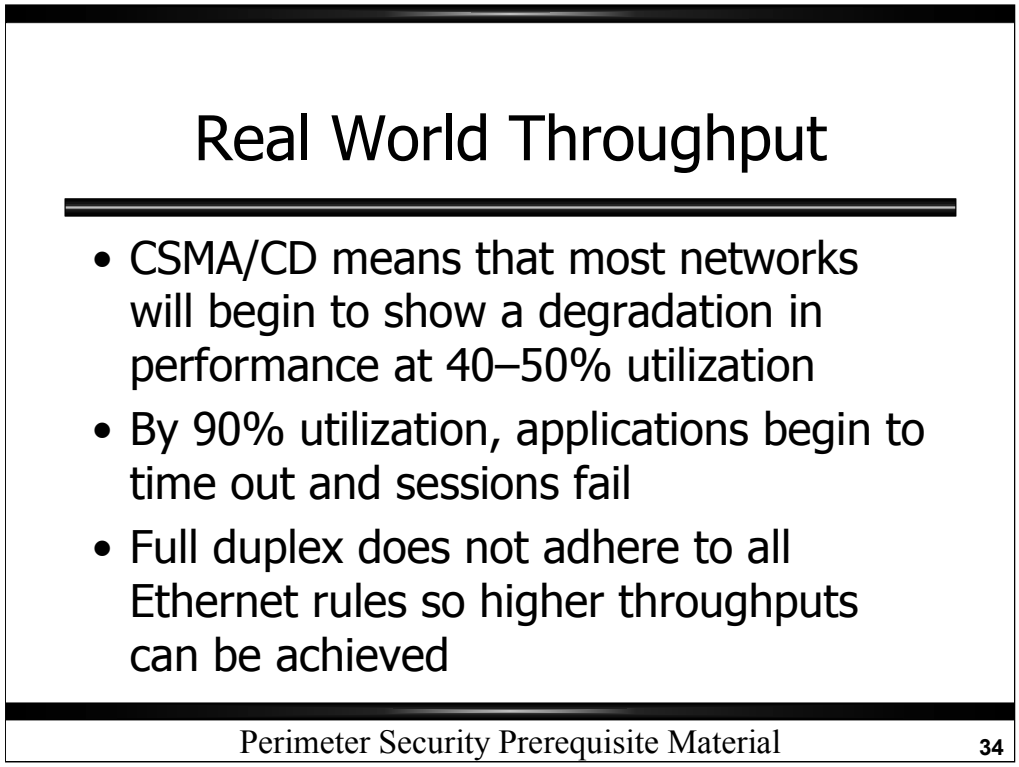
bits / 8 = bytes

So if I'm working on a 100 Mb network and my backup system needs to transfer 8 GB of data on a nightly basis, I would use the following formula to calculate transfer rate under perfect conditions:

8,000,000,000 bytes / (100,000,000 Mb / 8) =

8,000 MB / 12.5 MB = 640 seconds or 10.7 minutes

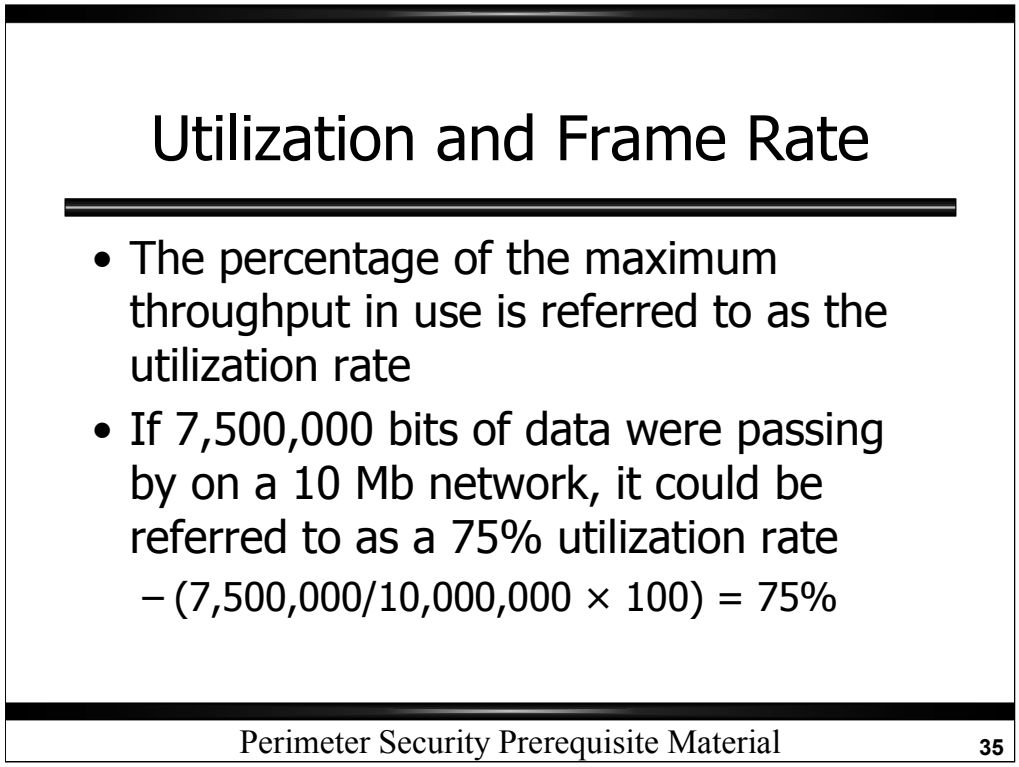
Of course ideal conditions are rarely achieved.



Ethernet uses Carrier Sense Multiple Access/Collision Detection (CSMA/CD) as a LAN access method. Ethernet stations are required to listen prior to transmitting. While listening, the wire must be free from transmissions in order for a station to decide it's OK to send a frame. This "quiet time" is overhead and can be considered wasted bandwidth as no data is actually transferred.

The more stations transmitting on the wire, the harder it is to find enough quiet time for a station to decide that it's OK to transmit. As a network reaches 50% utilization a station has a 50/50 statistical chance that the network will be busy and it will have to wait prior to transmission. This degrades network performance for that system. As utilization increases, performance degrades even further.

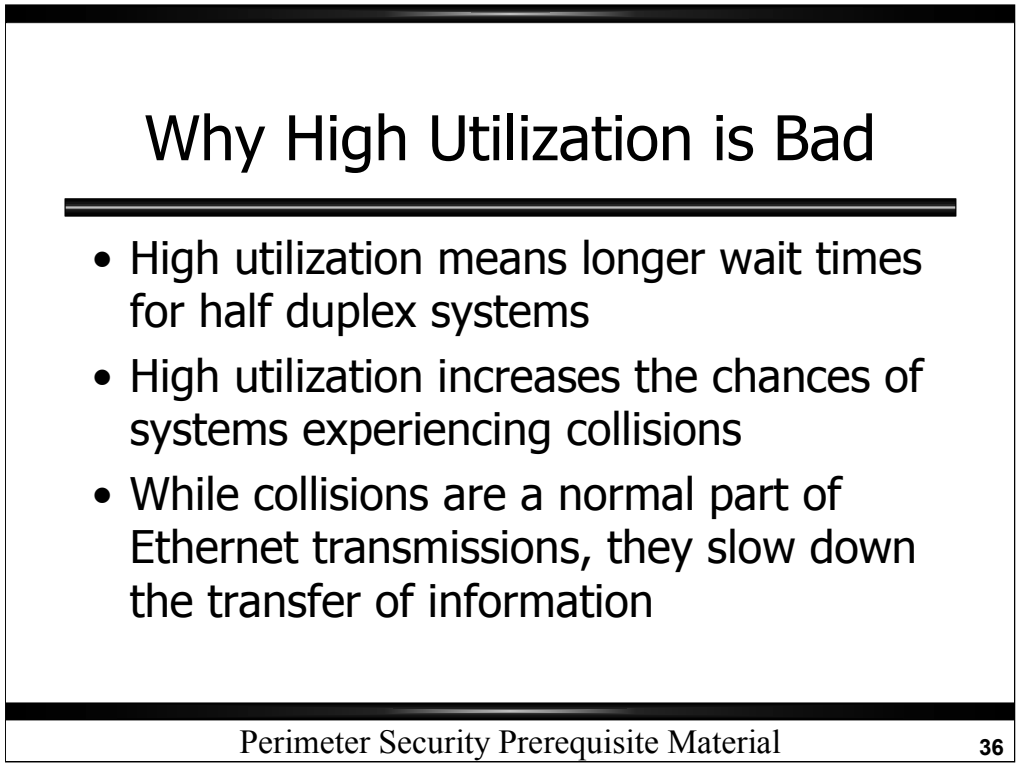
Full duplex, which we will discuss in greater detail later, does not adhere to Ethernet's "listen for a quiet moment prior to transmission" rule. This means that even at high utilization levels full duplex can be an efficient means of communication.



So what is utilization? Simply put, utilization is ratio of the current throughput rate in reference to the maximum throughput rate. For example 10 Mb Ethernet is capable of transmitting 10,000,000 bits per second. If in a one second time period you measure 7,500,000 bits being transmitted, the network is experiencing a 75% utilization level.

People like to look at average utilization to measure a network's load. This is fine but keep in mind the window of time over which you are taking your measurements.

For example let's say you measure the amount of traffic passing though your network in a 24 hour period of time. You run a few calculations and determine that you are experiencing a 30% utilization level over that 24 hour period of time. While this is below our point of concern, think about how your organization does business. What if the wire is only busy for 8 hours but nearly silent for the remaining 16 hours of the day. With this in mind, your average utilization during normal business hours may be closer to 88% utilization! Well above our level of concern.



So why is high utilization bad? We touched on this a few slides ago but it is an important enough concept to cover in detail here.

The higher the utilization level, the greater a chance that a system waiting to transmit is going to have to pause before finding a window of opportunity in which to transmit. Also, with so many systems waiting to transmit, the chances of a network collision taking place grows exponentially as well.

While collisions are a normal part of Ethernet communications, they do slow down performance. When a system determines it has been involved in a collision, it must pause for a random period of time before again attempting transmission. This pause as well as the time wasted transmitting the frame the first time prior to the collision translates into slower throughput and poorer performance.

When a station experiences two or more collisions while attempting to transmit a frame, this is referred to as multiple collisions. Multiple collisions are an indication that the network can not handle the load.

Frame Rate

- A legal Ethernet frame can be anywhere from 64 to 1,518 bytes in length.
- 10 Mbps Ethernet experiencing 100% utilization of 1,518 byte frames would have a frame rate of 813fps.
  - (10,000,000/8)/(1,518+8+12) = 813 fps
- What happens to frame rate when the frame size drops?

Perimeter Security Prerequisite Material 37

So far we have discussed utilization in great detail. Another yard stick of network load is frame rate. Frame rate is the quantity of frames passing through the network in any one second period of time (fps). Frame rate is directly related to utilization in that, as the frame rate increases, utilization does as well.

So how many frames can be transmitting on a 10 Mb Ethernet network? This quantity varies depending on the size of the Ethernet frames. The throughput rate divided by the frame size will tell us how many frames can be transmitted in any one second period of time.

First, we convert bits to bytes with 10,000,000/8. This gives us a throughput rate of 1.25 MB as we discussed earlier.

Next, we take the maximum Ethernet frame size of 1,518 bytes and add to it:

-The preamble at 8 bytes

-Signaling lost to listening prior to transmission at 12 bytes

We now divide the transfer rate by the maximum frame size to get:

-1,250,000 / 1,538 = 813 frames per second (fps)

So if all of the frames on the wire are maximum size frames, we can transmit a maximum of 813 fps under perfect conditions.

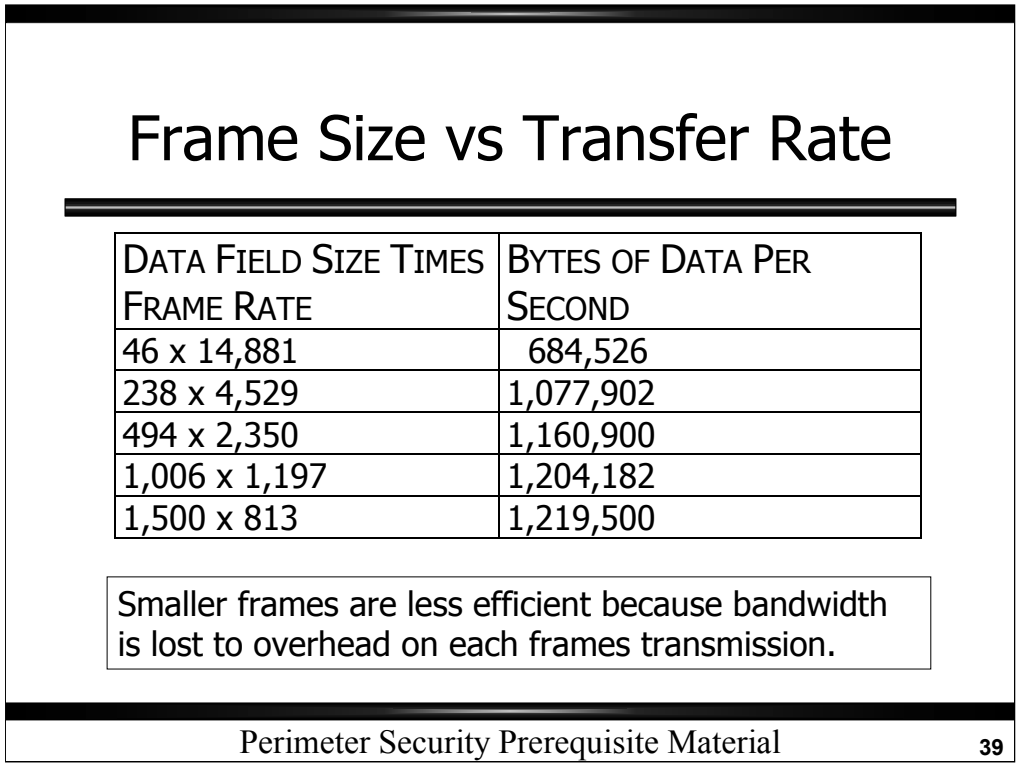
Frame Size vs Frame Rate

Frame Size in Bytes	Number of Frames at 100% Utilization
64	14,881
256	4,529
512	2,350
1,024	1,197
1,518	813

So what is more efficient, many little frames or fewer larger frames? Is the efficiency the same?

The chart above shows the correlation between frame size and the maximum number of frames that can be transmitted within a one second period of time. Since it takes less time to transmit a 64 byte frame than it does a 1,518 byte frame, it only stands to reason that you can push more of them out on the wire at any given time.

Of course this begs the question, does size really matter? Do we care if our frames are runty or large? Do the size have any effect on the amount of actual data we can push across the wire?



This chart says it all. Note that the minimum Ethernet frame size is only half as efficient as the full size frame in transferring data.

The reason is transmission overhead. Each frame transmitted uses:

12 bytes for listening

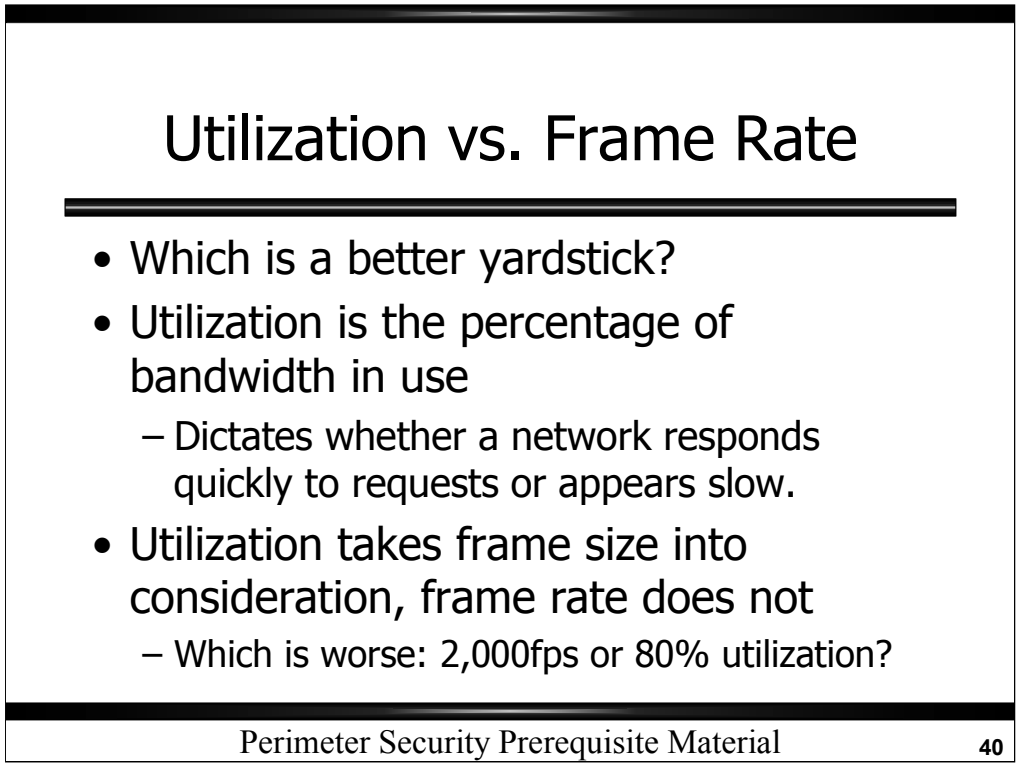
8 byte preamble

14 bytes for an Ethernet header

4 bytes for the trailer (CRC)

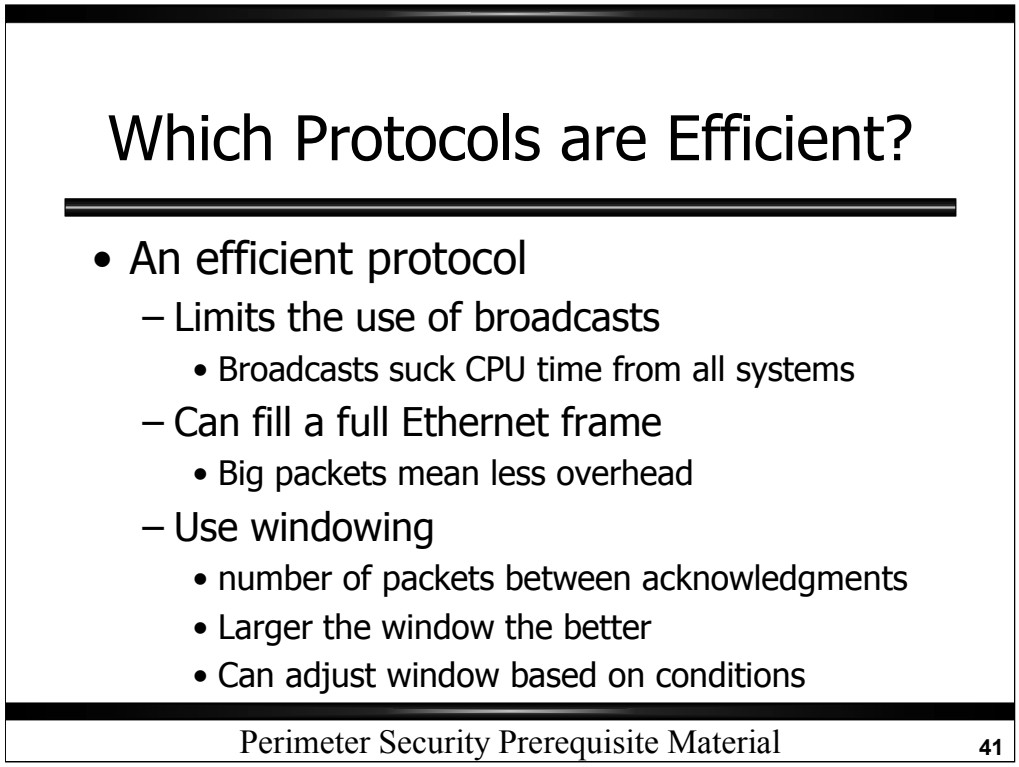
for a total of 38 bytes lost to overhead every time a frame gets transmitted onto the wire. Note that we do not even include the overhead due to an IP and TCP header! This would drop our efficiency even further.

With these calculations in mind, think about how efficient ATM communications are with its small 48 byte cell size.



So what's a better yard stick of network load, utilization or performance? Typically, utilization is the preferred metric as it's a better expression of how much bandwidth is in use at any given time.

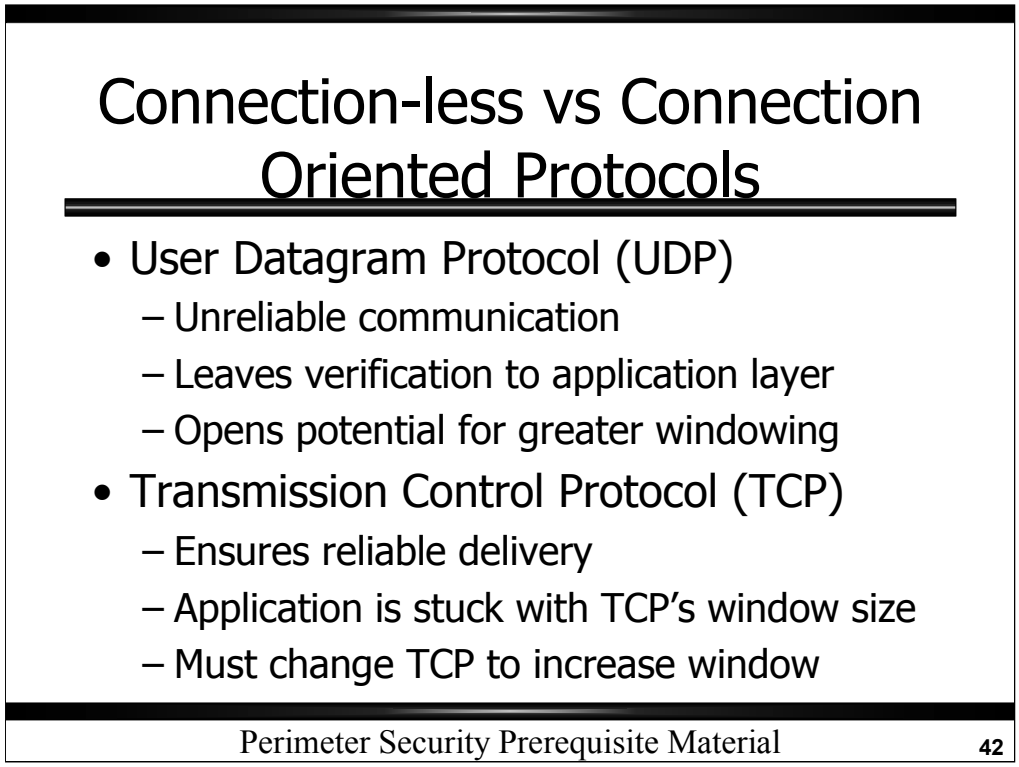
For example if I say to you "Our network is seeing 2,000 fps, this could be a good thing or a bad thing, you don't really know. You would be forced to ask "What's the average frame size?". If it's 64 bytes the network is not even breaking a sweat, if its 900 bytes the wire is saturated. So fps needs an additional metric to have any meaning. Utilization on the other hand tells the whole story as a single number. It's a combination of fps and frame size.



So what makes a protocol efficient? Limited use of broadcasts is extremely helpful. This keeps systems on the wire from constantly being interrupted in order to process data. For example the average AppleTalk server broadcasts a minimum of six times a minute. An IP network can be configured to generate zero broadcasts.

Another useful feature is the ability fill an Ethernet frame. Many older protocols limit data transfer to 512 bytes per frame. This means that nearly three times as many frames must be generated to move our data from point "A" to "B". Sometimes this is a function of the networking protocol while sometimes it's a problem with the application service. For example both Telnet and file transfer protocol (FTP) use TCP/IP as a transport. While FTP can fill an Ethernet frame, Telnet typically only uses minimum size Ethernet frames.

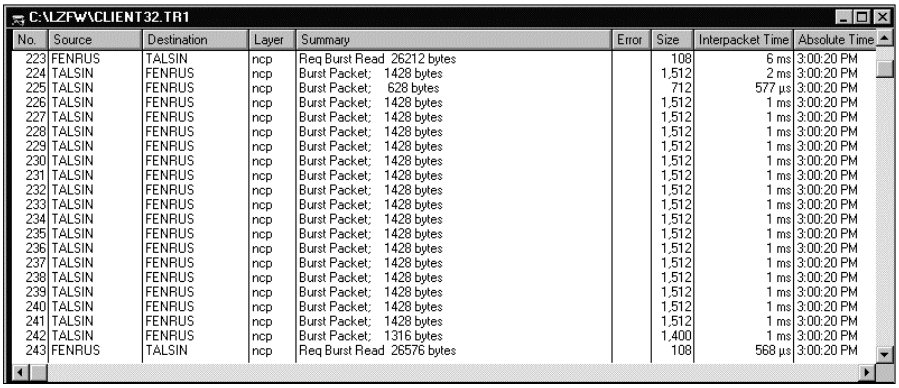
Finally, windowing is an excellent way to increase efficiency. Normally, transmitting systems like to see an acknowledgment for every packet of data transmitted. This helps to increase reliability. Unfortunately, this also decreases efficiency. *Windowing* allows a system to transmit multiple frames of data before receiving an acknowledgement. The larger the window size, the greater the efficiency. Of course windowing needs to be negotiated on the fly so if network conditions deteriorate, smaller windows can be used to increase reliability.



There are two places that windowing is typically implemented, at the transport layer or at the application layer. UDP is a connection-less protocol which means that it does not take care of acknowledging data delivery. This means the acknowledgement process must be built into the application using UDP as a transport. While this requires additional work, it also leaves the developers free to maximize windowing to create efficient communications.

TCP on the other hand is a connection oriented protocol. This means that TCP takes care of the acknowledgments and has windowing built right in. Unfortunately, TCP windowing is rather small allowing only 3-4 packets or frames between each acknowledgment.

NetWare's IPX With NCP



No.	Source	Destination	Layer	Summary	Error	Size	Interpacket Time	Absolute Time
223	FENRUS	TALSIN	ncp	Req Burst Read 26212 bytes		108	6 ms	3:00:20 PM
224	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	2 ms	3:00:20 PM
225	TALSIN	FENRUS	ncp	Burst Packet; 628 bytes		712	577 µs	3:00:20 PM
226	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
227	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
228	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
229	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
230	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
231	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
232	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
233	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
234	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
235	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
236	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
237	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
238	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
239	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
240	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
241	TALSIN	FENRUS	ncp	Burst Packet; 1428 bytes		1,512	1 ms	3:00:20 PM
242	TALSIN	FENRUS	ncp	Burst Packet; 1316 bytes		1,400	1 ms	3:00:20 PM
243	FENRUS	TALSIN	ncp	Req Burst Read 26576 bytes		108	568 µs	3:00:20 PM

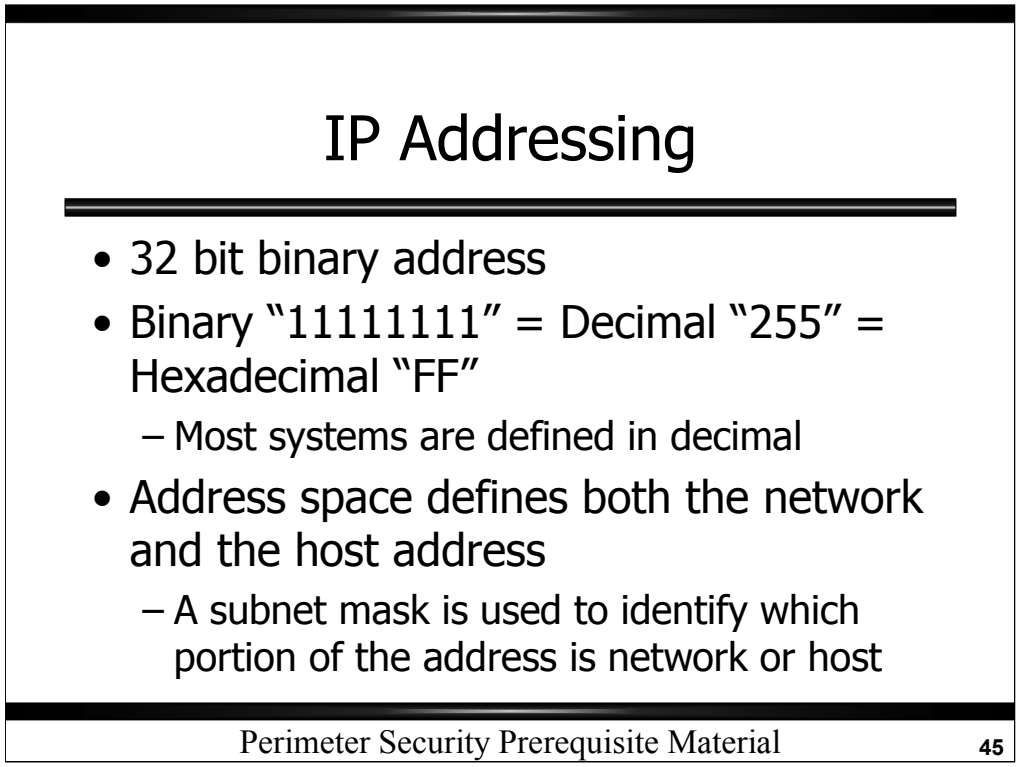
Probably one of the most efficient protocols around for transferring data is NetWare Core Protocol (NCP) running on top of IPX. As you can see from this screen capture, Fenrus requested 26,212 bytes of data from Talsin. Instead of acknowledging each packet, NCP windowing allows the whole thing to be transmitted before requiring an acknowledgment. If you look at the last line this acknowledgment comes in the form of an additional data request. Also note that since no errors were experienced in transmitting the 26,212 bytes, Fenrus is now asking for 26,576 bytes. The ability to adjust the window on the fly with favorable network conditions makes this protocol very efficient.

Telnet under TCP/IP

No.	Source	Destination	Layer	Summary	Size
1	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	telnt	Data=l	64
2	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=l	64
3	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1036 --> TELNET ACK	64
4	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	telnt	Data=s	64
5	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=s	64
6	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1036 --> TELNET ACK	64
7	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	telnt	Data=..	64
8	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=..	64
9	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1036 --> TELNET ACK	64
10	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=install.log..	71
11	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1036 --> TELNET ACK	64
12	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=[cbrenton@thor /tmp]$	80
13	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1036 --> TELNET ACK	64
14	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	telnt	Data=l	64
15	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=l	64
16	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1038 --> TELNET ACK	64
17	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	telnt	Data=s	64
18	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=s	64
19	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1038 --> TELNET ACK	64
20	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	telnt	Data=..	64
21	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=..	64
22	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1038 --> TELNET ACK	64
23	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=install.log..	71
24	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1038 --> TELNET ACK	64
25	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=[cbrenton@thor /tmp]$	80
26	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1038 --> TELNET ACK	64
27	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	telnt	Data=l	64
28	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=l	64
29	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1039 --> TELNET ACK	64
30	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	telnt	Data=s	64
31	THOR.FOOBAR.COM	LOKI.FOOBAR.COM	telnt	Data=s	64
32	LOKI.FOOBAR.COM	THOR.FOOBAR.COM	tcp	Port:1039 --> TELNET ACK	64

In contrast to NCP/IPX, take a look at a Telnet session using TCP/IP. Note that the window size never grows to more than two frames. This increases the number of frames on the wire, thus decreasing the efficiency of the transmission.

Also look at the right hand column at the frame size. As mentioned Telnet uses runty little frames as typically one character is transmitted at a time. While it would take very many Telnet sessions to fill a 10 Mb circuit, any sessions that do exist are making poor use of the available bandwidth.



Chances are you have a pretty good idea of how to work with IP addresses. With this in mind we will only hit the highlights.

IP addressing is based on the binary numbering system. While we use decimal numbers for convenience, all background processing is done in binary.

A unique feature of IP addressing is that a portion of the IP address expresses the network address while another portion of the address defines the specific host. For example, if you see the address 10.100.15.73, the possibilities are:

network	host
10	100.15.73
10.100	15.73
10.100.15	73

To figure out which possibility is correct, the IP address needs to be associated with a subnet mask. The subnet mask value is used to define which bits are network and which bits define the host.

Subnetting

Mask	# of Subnets	Hosts per Subnet	Available host Address ranges	Network Address value	Broadcast Address value
255.255.255.128	2	126	1 - 126 129 - 254	0 128	127 255
255.255.255.192	4	62	1 - 62 65 - 126	0 64	63 127
255.255.255.224	8	30	1 - 30 33 - 62	0 32	31 63
255.255.255.240	16	14	1-14 17 - 30	0 16	15 31
255.255.255.248	32	6	1-6 9 - 14	0 8	7 15
255.255.255.252	64	2	1 - 2 5 - 6	0 4	3 7
255.255.255.254	128	0	0	0	0

Perimeter Security Prerequisite Material 46

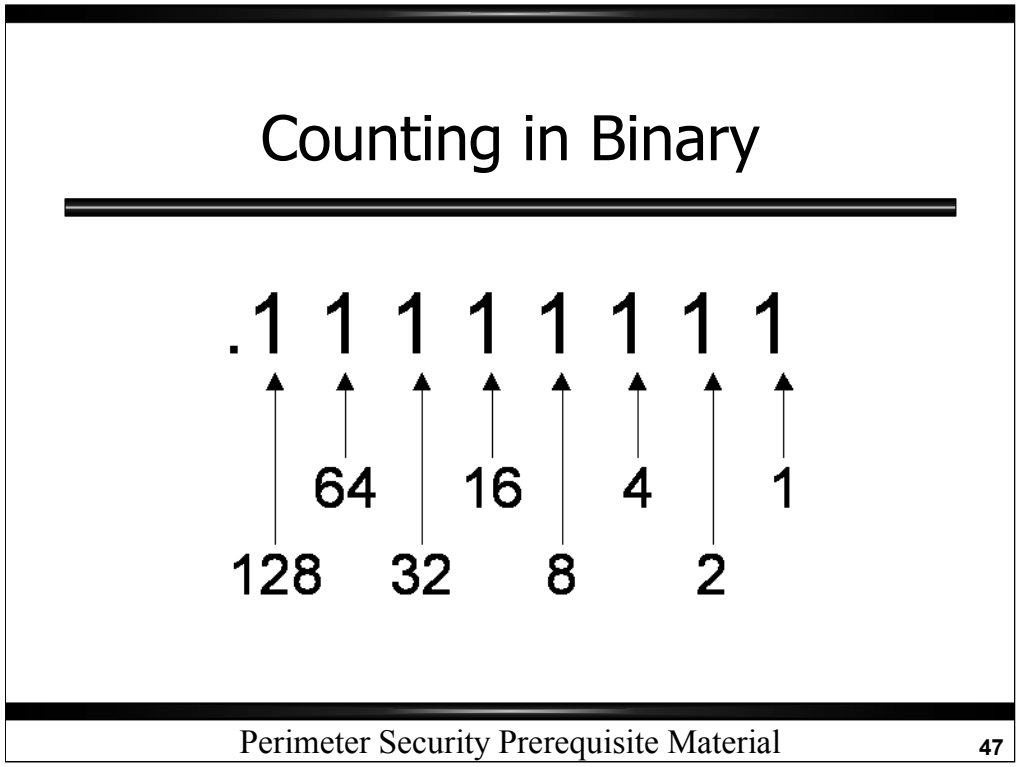
One of the neat things about subnetting is that you can adjust the size of the address and host space in order to fit your needs. This slide shows the first two subnets that get created on a class C network when you vary the number of host bits set to "1" in the last octet. Note that as the number of available networks increase, the number of supported hosts decreases. Also note that every time we create a new network we lose potential addressing for two hosts. As one address is lost to the new network address and one is lost to the broadcast address.

When subnetting was first introduced in Request For Comment (RFC) 1123, the rules stated that the first and last subnet should not be used. This means that a 255.255.255.128 subnet mask would create no useful subnets. These "rules" were later updated however in RFC 1878 which permit the use of the first and last subnet as shown in the chart above.

Some devices need a bit of coaxing to operate correctly. For example Cisco routers running IOS 12.0x and prior follow the rules of RFC 1123 by default. You must enter the command:

ip subnet-zero

from global configuration mode to make the device RFC 1878 compliant and allow the router to recognize first and last subnets as valid.



So where did the mask values come from that where shown in the last slide? Each decimal mask value is based on a corresponding binary value. The mask value is based on how many bytes are "borrowed" for the networking address.

This chart shows the breakdown:

.10000000 = .128 = /25

.11000000 = .192 = /26

.11100000 = .224 = /27

.11110000 = .240 = /28

.11111000 = .248 = /29

.11111100 = .252 = /30

.11111110 = .254 = /31

Note the “/XX” notation shown above. This is a short hand method of describing the subnet mask by identifying the number of bits used for networking. Written out completely you get:

255.255.255.192 = 11111111.11111111.11111111.11000000 = /26

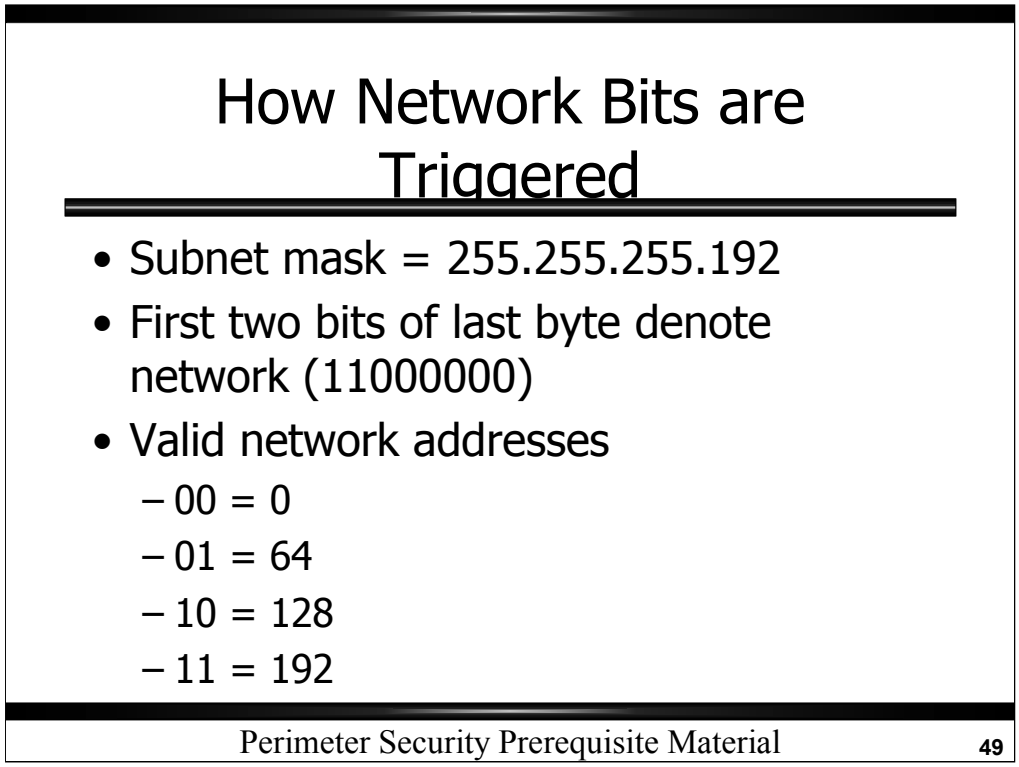
So a 255.255.255.192 subnet mask can be expressed as a “/26” because 26 of the bits are set to “1” and used to identify the network portion of the address.

Calculating Hosts & Subnets

- Last octet of 255.255.255.248
  - .11111000
  - Five "1's" mean five bits for mask
    - 2^5 = 2x2x2x2x2 = 32 networks
  - Three "0's" means three bits for hosts
    - 2^3-2 = 2x2x2-2 = 6 hosts per subnet

The mask value can also be used to determine the number of networks and hosts that get generated. For example the number of valid networks is 2 to the power of the quantity of "1's" in the mask. The number of valid hosts is 2 to the power of the quantity of "0's" in the mask -2.

So .128 would be 2^1 or 2 network segments. The number of valid hosts would be 2^7 - 2 or 126 host addresses.



So we've figured out how bits are borrowed to create additional address space. Now we need to know how to figure out the network address for each of our newly created networks.

This process is as simple as counting in binary. For example if we convert 255.255.255.192 to binary, we find that the two high order bits will be used for networking and the remaining bits will be for addressing hosts. With this in mind we simply trigger each of the two high order bits sequentially in order to get each of our valid network addresses.

Host bits are derived in the same fashion using the lower order bits. For example on the .64 network, the first few hosts would be:

01000001 - 65

01000010 - 66

01000011 - 67

01000100 - 68

Finding the Network Address

- Find the network and broadcast address for 192.168.1.121/29

```
        01111001 = host
    +   11111000 = mask (.248)
        01111000 = 120
Network address = 192.168.1.120

Set all host bits high to calculate the broadcast address
network  =      01111000
broadcast =     01111111  = 192.168.1.127
```

Perimeter Security Prerequisite Material 50

Many people do not realize it, but its possible to calculate the network and broadcast address for a specific host provided you know its IP address and subnet mask value. The network address is derived by performing a logical "AND" between the host address and the subnet mask When performing a logical AND:

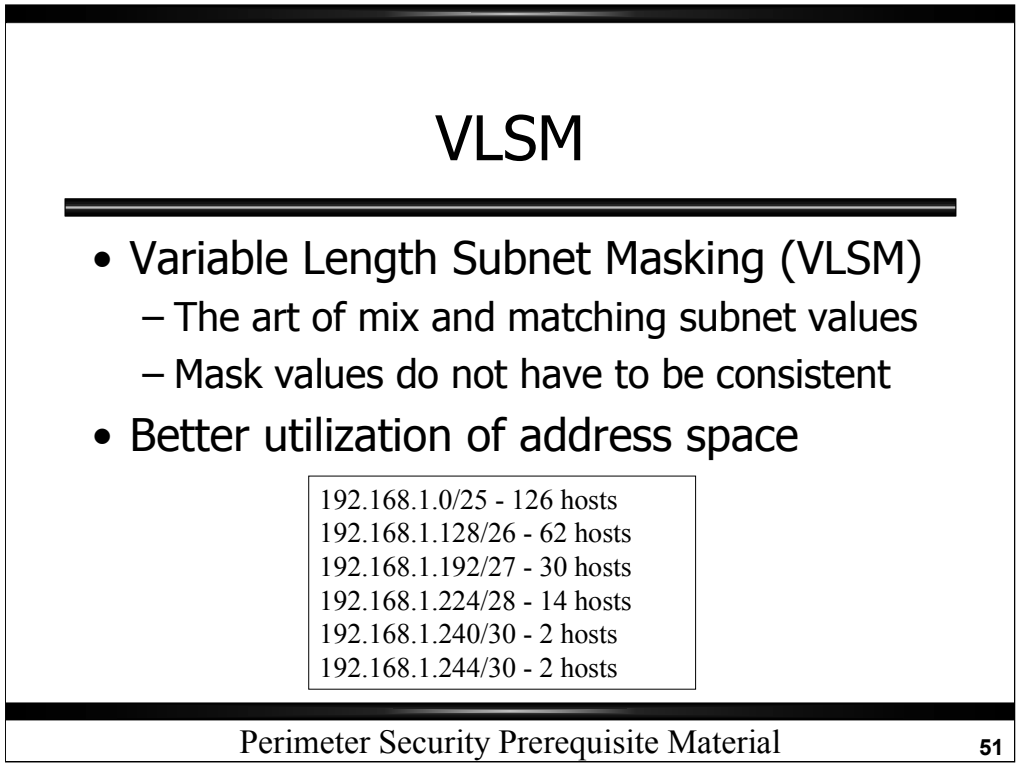
1 + 1 = 1

1 + 0 = 0

0 + 1 = 0

As shown in the slide above we converted the last octet of the IP address and the subnet mask to binary. We then perform a logical AND between the two values. The result is the valid network address for this host.

The broadcast address is even easier to derive. Now that we know the network address, we simply set all the host bits high to arrive at the broadcast address.



An important concept to understand is that you do not need to keep the subnet mask value consistent throughout your network. Variable Length Subnet Masking (VLSM) allows you to mix and match mask values to suit your needs.

For example, let's say you need to break up a single class C address space to support the following:

100 hosts

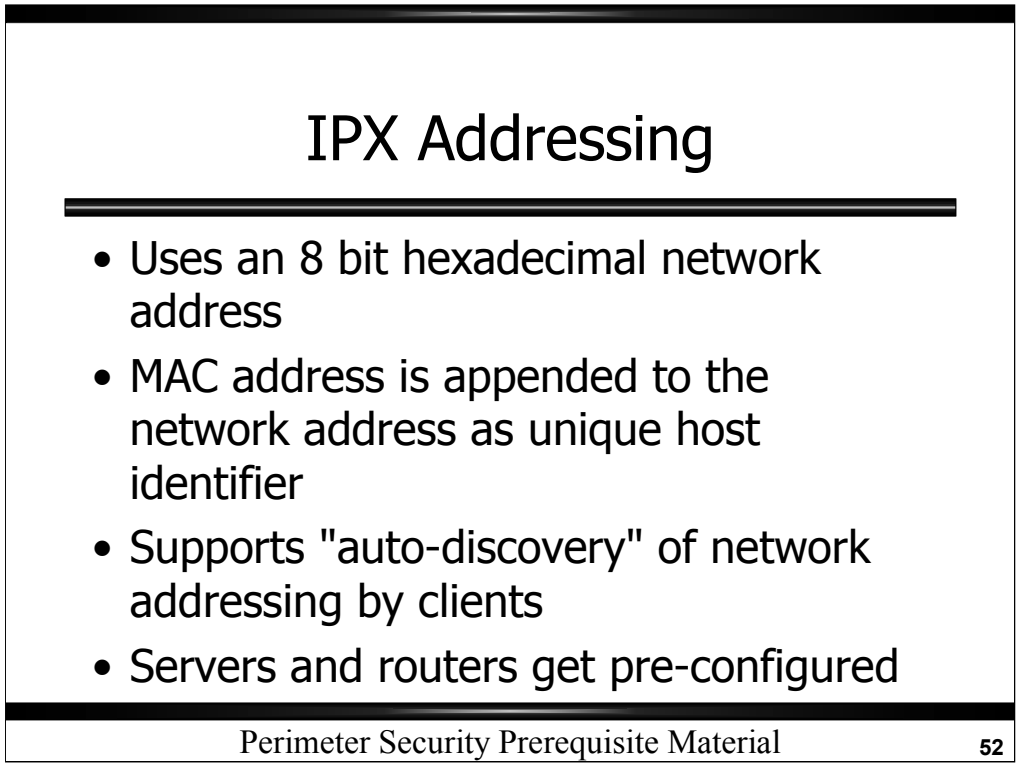
50 hosts

20 hosts

11 hosts

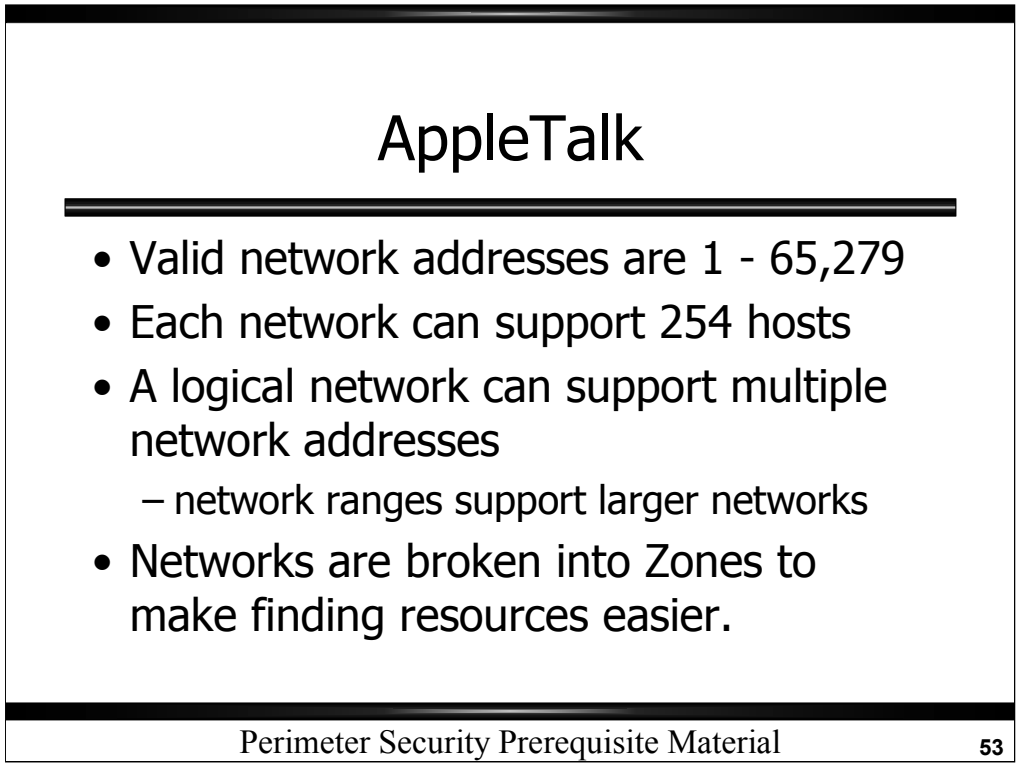
2 point to point WAN links (2 hosts on each)

Clearly there is no single subnet mask that would fit the bill. By breaking the network up as shown in the slide above you could easily address each of these network segments.



If addressing IP networks has left your head spinning, IPX is going to seem like a walk in the park. IPX uses hexadecimal addresses rather than binary. So while 192.168.1.0/24 is valid in the IP world, in IPX the network addresses look more like “ba5eba11” and “deadba55”.

Unlike IP, only IPX routers get configured with IPX network addresses. This includes IPX servers as well, as technically, they also act as routers. When an IPX client comes on-line they simply transmit a single broadcast packet in order to learn the local network address. This is similar to having DHCP built directly into the protocol except you have fewer address conflicts because the host address is taken from the network card’s MAC address.

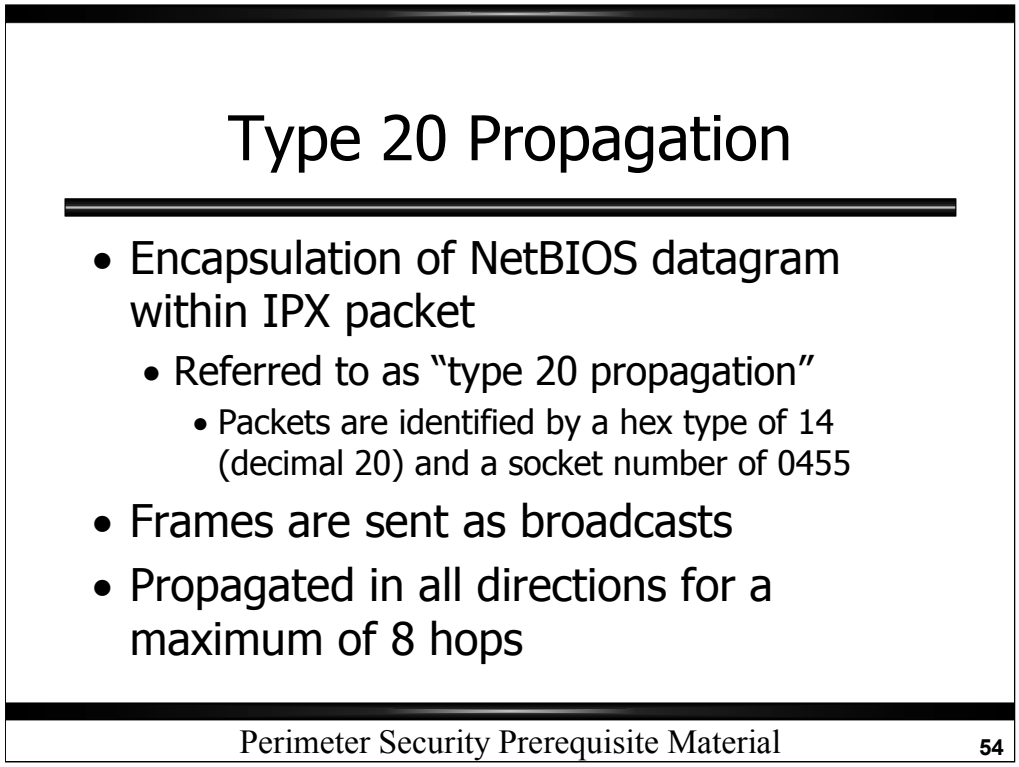


Following in the typical Apple fashion, AppleTalk is arguably one of the easiest protocols to setup and configure. Network addresses are expressed in decimal format with ranges being permitted to support a larger number of hosts. For example, a network address of 1 can support 254 hosts, 1.1 through 1.254. If I define a network range of 1-3, I can now support 762 hosts, 1.1 though 3.254.

AT uses what is referred to as "seed nodes" to pre-configured the network address range. Typically only a single router is designated as the seed. All other hosts are referred to as "non-seed nodes" and learn the address range on startup. Non-seed nodes remember their last address and will attempt to use it on startup. All stations will complain if they think a device is advertises the wrong address range.

Pay close attention to this fact when making address changes. It means that in order to change a network's address you have to power down all the non-seed devices, reprogram the seed and then start everything back up again.

Each network range gets a primary Zone Name assigned to it. These are human labels such as "Corporate" or "Sales" to make finding resources easier. Secondary Zone Names can be assigned to further identify resource locations. So while the primary zone may be Corporate, there may be secondary zones of "First Floor", Second Floor" and so on.

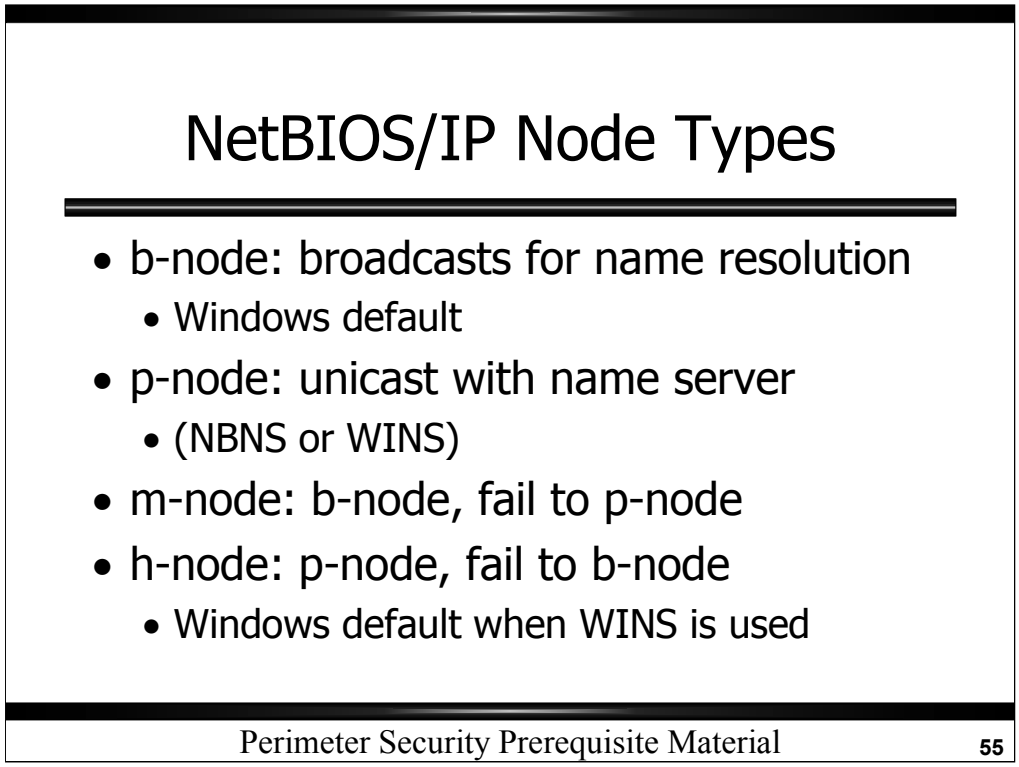


If there were ever an "evil" protocol, NetBIOS/IPX would be it. Known as type 20 propagation NetBIOS/IPX is a NetBIOS packet using IPX as a transport. One of the things that make this protocol so evil is that transmissions are all broadcast based. This means that all conversations suck CPU time from every system on the wire. NetBIOS/IPX was the default for many years for quite a few client/server applications such as Lotus Notes.

While NetBIOS/IPX is block by default on most routers, you can allow it to pass by enabling type 20 propagation. This is the only way to get NetBIOS/IPX from one logical network to another. Unfortunately, there is no real routing involved so this traffic gets propagated in every direction for a maximum of 8 hops (it will cross 8 routers and then be discarded).

So you have broadcast based traffic propagating in every direction for a maximum of 8 hops sucking CPU time from every device it encounters. Sounds like evil to me.

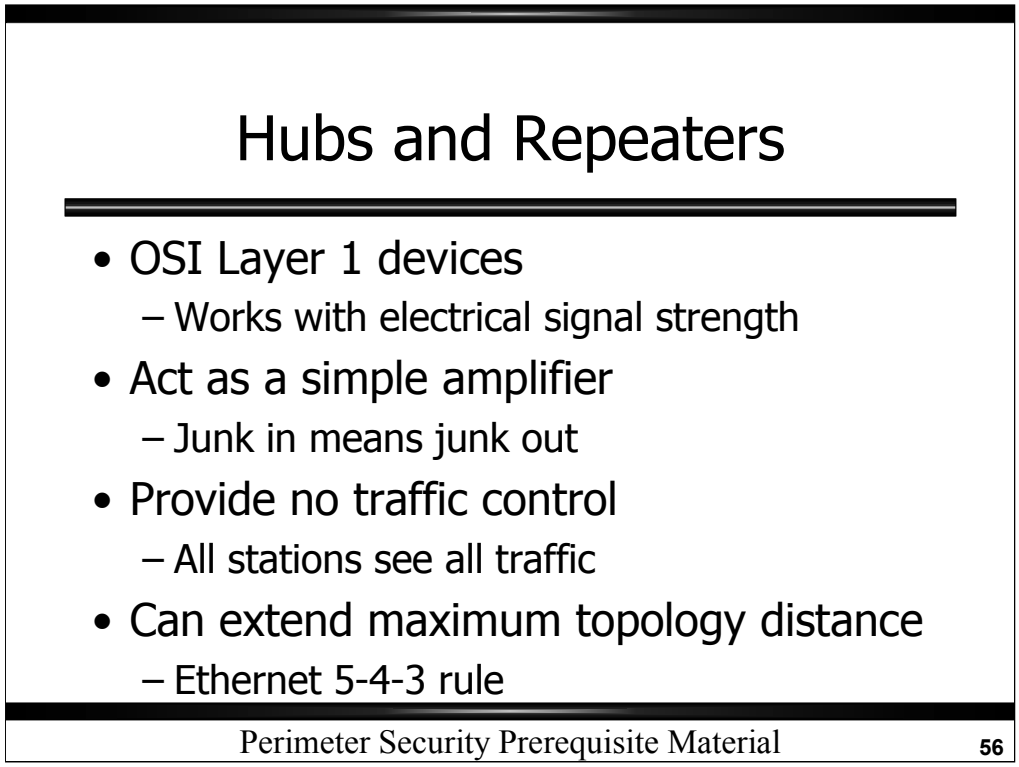
Because of the hit on network performance, sites that need NetBIOS for communications (such as NT shops) prefer to use IP to encapsulate NetBIOS.



NetBIOS/IP is not quite as evil as its NetBIOS/IPX counterpart. NetBIOS/IP uses different node types to define how NetBIOS/IP systems will communicate with each other. The default for all Windows systems is to communicate as a "b-node". This means that the Windows system will advertise itself as well as find other systems by transmitting broadcasts. If data needs to be transferred, unicast communications are used. So while the actual communication sessions are efficient, Windows systems waste a lot of bandwidth by broadcasting once or more per minute just to provide name resolution. The other problem is that since b-node name discover is broadcast based, it cannot cross a router. So you must take special steps to get one Windows system to find another located on the other side of a router.

In the p-node communication sessions, the NetBIOS/IP systems do not advertise their names or use broadcasts to find other systems. Instead, all name resolution is provided using a central database. In the case of Windows systems this would be a WINS database. When a system comes on line it registers itself with WINS. When the system needs to find another host it queries the WINS database. Since all communications are unicast, no bandwidth is wasted to broadcasts.

M-nodes and h-nodes are simply a combination of b-node and -node functionality. The preferred configuration is p-node as it starts off using unicast communications. If WINS fails to generate a successful lookup, the host can fall back on broadcasts. This provides a level of fault tolerance.



Simply put, hubs and repeaters are dumb amplifiers. Think of them as being like the amplifier in your stereo and you'll get the idea. Pump in classical, you get classical out the other end. Pump in garbage and noise, and the hub or repeater will happily amplify the garbage and noise and pump it out the other side.

One important aspect of repeaters and hubs is that they provide no traffic isolation. Most of the examples up till now have assumed a repeated environment where all systems can see the transmissions of everyone else.

While hubs and repeaters can help to expand the size of you network, remember the *Ethernet 5-4-3 rule*:

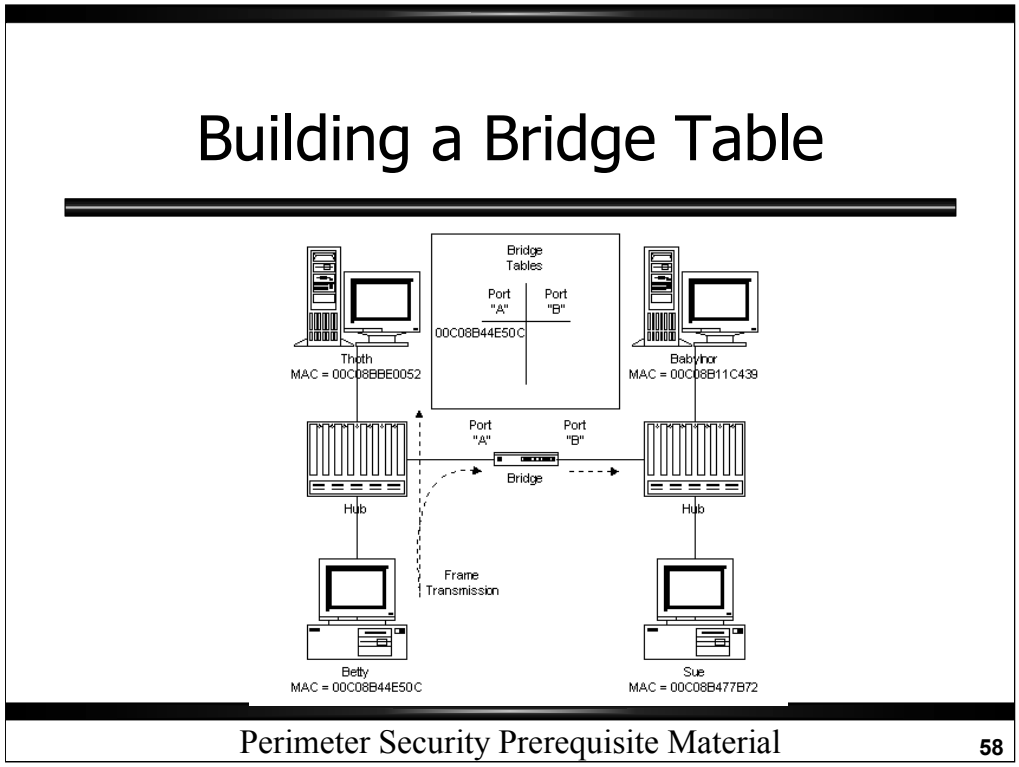
"No more than 5 segments with 4 repeaters. Only 3 of the 5 segments may be populated".

By *populated*, the rule means a segment that is not a point to point connection and has more than two systems attached. For example if hubs are used to extend an Ethernet topology no more than two of them may be cascaded together, because each hub is considered to be a populated segment.

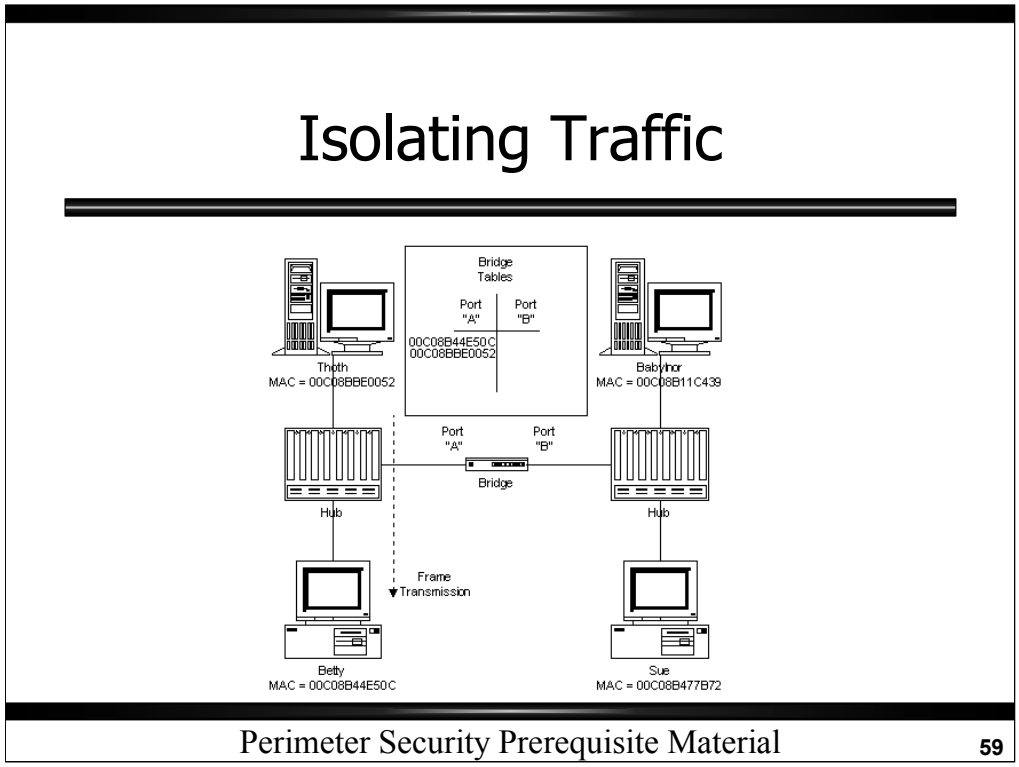
Bridges

- Monitoring the source and destination MAC address on each frame of data
- Learn where all the network systems are located
  - Construct a table, listing which MAC addresses are off of each port
- Will use the table to play traffic cop and regulate the flow of data

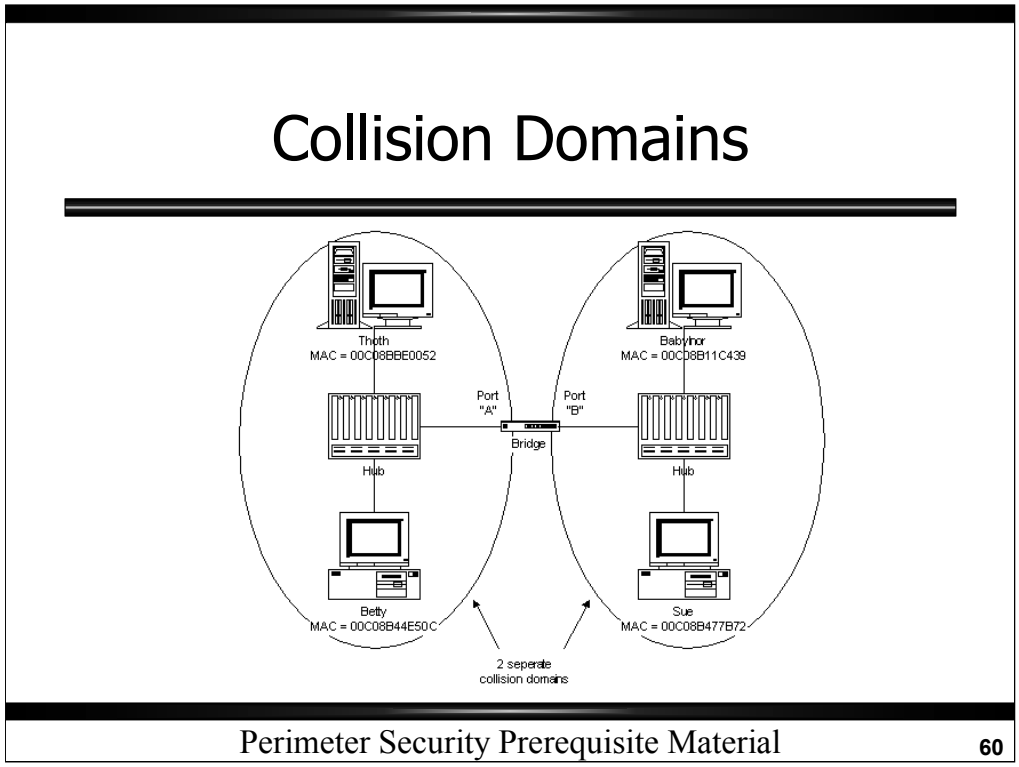
Unlike a repeater, bridges introduce a layer of traffic control into the network. As mentioned, the Ethernet header contains the MAC address of the destination system as well as the MAC address of the system that transmitted the frame. A bridge can use this header information to make intelligent choices when forwarding traffic.



This figure shows how a bridge goes about learning where systems are located on the network. In this example, Betty transmits a frame of data to Thoth. Since the bridge does not know where Thoth is located, it forwards the traffic out port B. The bridge then looks at the source MAC address and makes a table entry indicating that Betty is located off of port A.



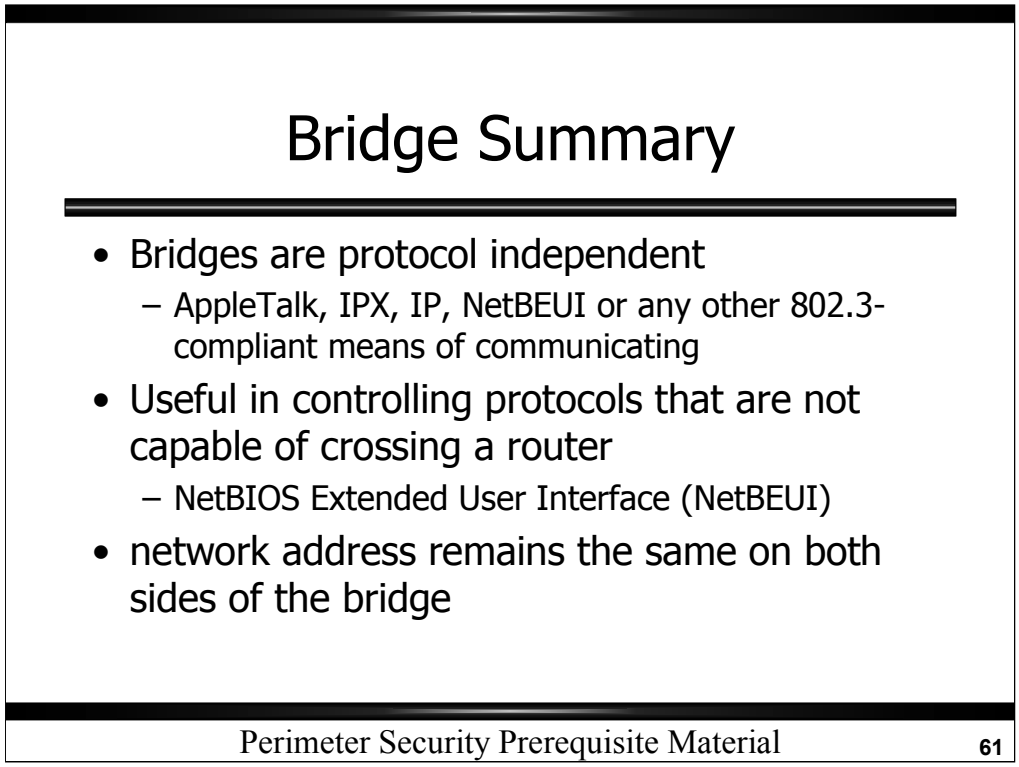
When Thoth replies to Betty, the bridge again looks at the header information. This time there is a match. The bridge sees that the frame is headed to Betty and that Betty is located off of port A. With this in mind the bridge will not forward the frame out port B since the target system (Betty) is located off of the same port. This means that at the same instant Thoth is transmitting information, Babylnor or Sue is free to be transmitting as well. The effect is a potential doubling of the available bandwidth.



The doubling effect comes from the fact that the bridge creates two separate network segments called *collision domains*. We are allowed to have one system per collision domain transmitting at the exact instant of time. If this was a repeated network, only one of the four systems would be allowed to transmit at any given time. By introducing a bridge, we can potentially have two systems transmitting in parallel.

Of course the operative word is “potential” as it relies on one of the conversations being between Betty and Thoth, and the other conversation being between Sue and Babylnor. If any other combination is used (such as Betty talking to Sue) we lose the ability to have two sets of parallel communications as both sessions must cross the bridge.

Also note that bridges propagate broadcasts so if any one of the systems transmits a broadcast this would eat into the available bandwidth as well.



One slick bridge feature is that they are protocol independent. Since the upper layer protocol is ignored, a bridge will work equally well with AppleTalk as it will with IP. So long as the framing is Ethernet, the bridge is happy.

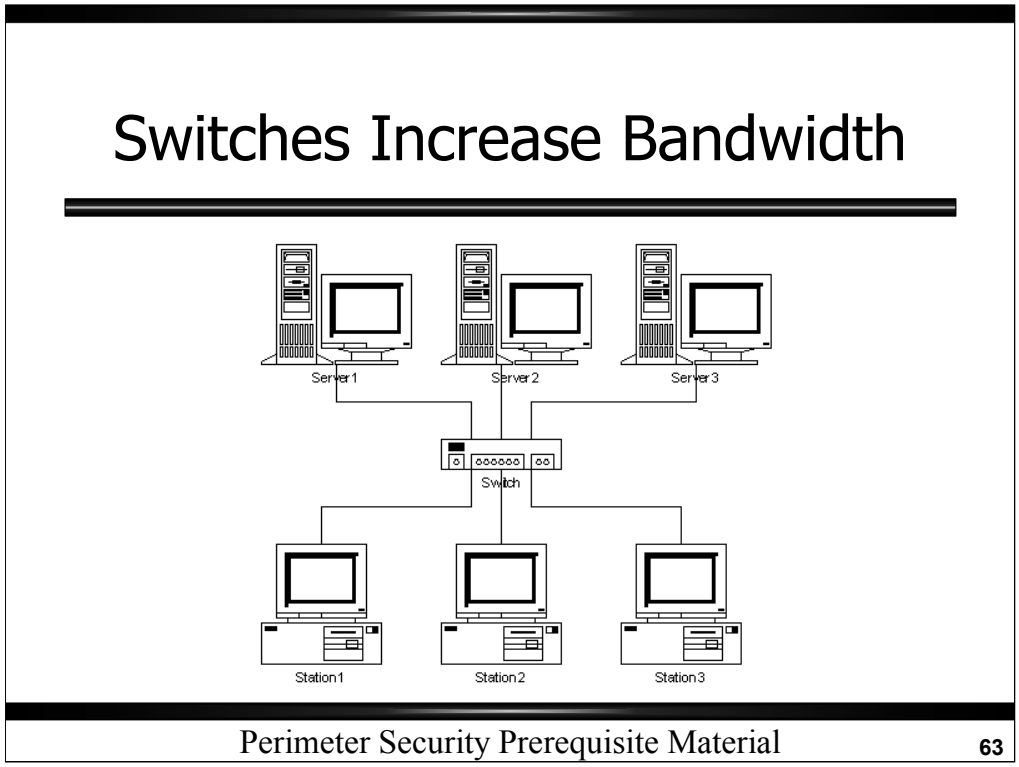
Also, bridges come in handy when you need to control non-routable protocols. For example NetBEUI is unable to cross a router and pass between logical network segments. Since the logical network is the same on both sides of the bridge, a bridge can be used to control NetBEUI traffic.

By “same logical network on both sides of the bridge”, I mean that the network addressing does not change. For example if the network located off the bridge’s "port A" is 192.168.1.0, then the network off "port B" would be 192.168.1.0 as well.

What's a Switch?

- Mixture of hub and bridge technology
- Instead of being a dumb amplifier like a hub, switches function as though they have a bridge built into the backplane
- A switch will keep track of the MAC addresses attached to each of its ports
  - transmit frames on only the needed ports

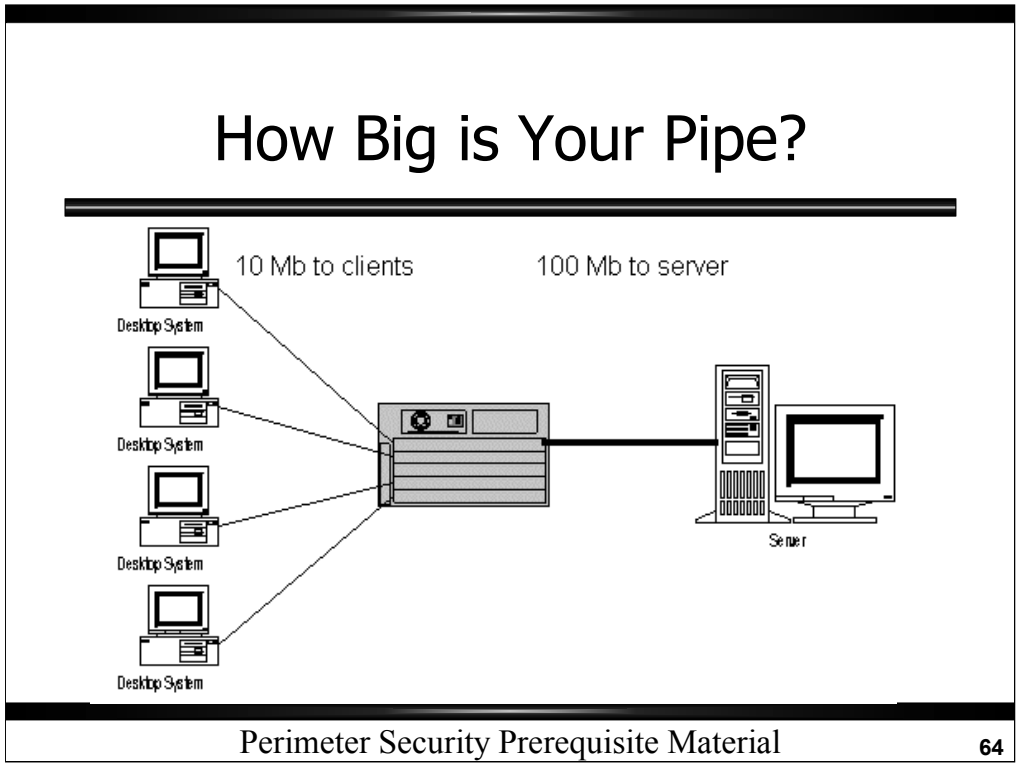
A switch is the marriage of hub and bridge technology. Think of a bridge as having more than two ports and you'll get the idea. A switch introduces the same type of traffic control that we saw in a bridge, the difference is that a switch has more ports.



The additional ports can come in handy and have a dramatic effect on potential bandwidth. For example, assume the above topology is 10 Mb Ethernet. What's the potential throughput?

If the above network used a hub, the potential throughput would simply be 10 Mb. Since we are using a switch however, the device's traffic control ability raises the bar on how much traffic we can pass at any given moment.

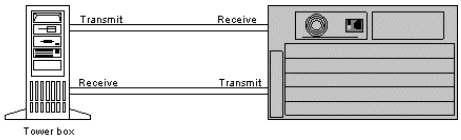
For example, let's assume Station 1 and Server 1 are carrying on a conversation, Station 2 and Server 2 are communicating, and Station 3 and Server 3 are exchanging data as well. Once the switch learns where everyone is located, it would be able to create three separate collision domains. This means or potential bandwidth is 10 Mb x 3 or 30 Mb. As we add more systems to our switch, the potential bandwidth increases even further.



Another neat switch trick is the ability to support multiple topology speeds. For example, in the above diagram we have connected each of our clients to the switch at 10 Mb. The server however is connected at 100 Mb. The fatter pipe to the server helps to balance the network load and keep the server from being overloaded.

Full Duplex Mode

- Each system has their own transmit pair
  - No worries of collisions
  - No need to listen before transmitting
- Doubles the potential bandwidth
  - Your mileage may vary



Perimeter Security Prerequisite Material 65

When we discussed bridges we introduced the concept of a collision domain. With a switch, every system directly connected to a switch port ends up being part of a two point collision domain. On one end is the switch, and on the other is the system.

If you look at the wiring in the diagram you'll see that each system connected to the switch has exclusive access to both a transmit and receive pair of conductors. The only system capable of transmitting on one wire pair is the server while the only system that can transmit on the other wire pair is the switch. The result is a condition in which collisions will never occur because there is no bandwidth contention within each collision domain.

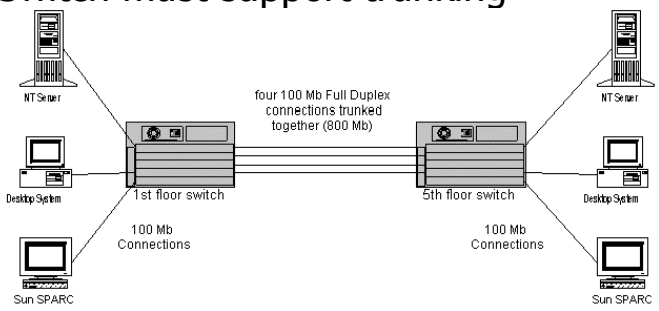
Since we can't have a collision, we don't have to worry about them. The Ethernet rule "listen before transmitting and don't send data if you see information on the wire" can be safely ignored.

This where full duplex mode comes into play. We are still using Ethernet except we can now safely ignore all of the rules which are used to prevent and recover from collisions.

Another benefit is that systems are free to send and receive data at exactly the same time. This effectively doubles the potential bandwidth of each switch port to 20 Mb. With this in mind our earlier example with 3 workstations and 3 servers would actually have a potential throughput of 60 Mb.

Trunking

- Allows you to aggregate switch ports for additional bandwidth
  - Switch must support trunking

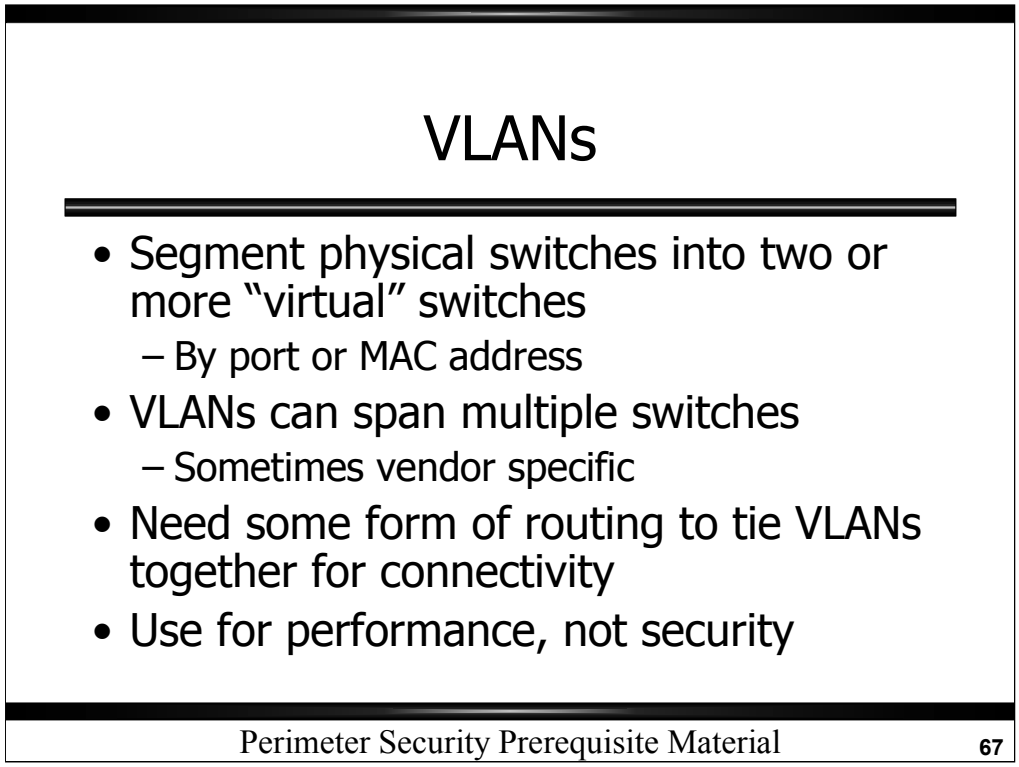


Perimeter Security Prerequisite Material 66

Most high end switches support trunking. With all of this potential bandwidth available to each system connected to our switches, we need some method of increasing the potential bandwidth between interconnected switches themselves.

This is where trunking comes into play. *Trunking* allows you to bind multiple switch ports together and have them look like a single circuit. The more ports you bind together, the greater the available bandwidth.

In the past trunking was a very vendor specific activity. It was not uncommon to find that it was impossible to trunk together switches from vendor “A” and “B”. This is starting to get better with the adoption of a universal specification referred to as 802.1Q. Adoption of this specification is still an on going process however, so don’t just assume that one vendor’s switch will trunk with another.

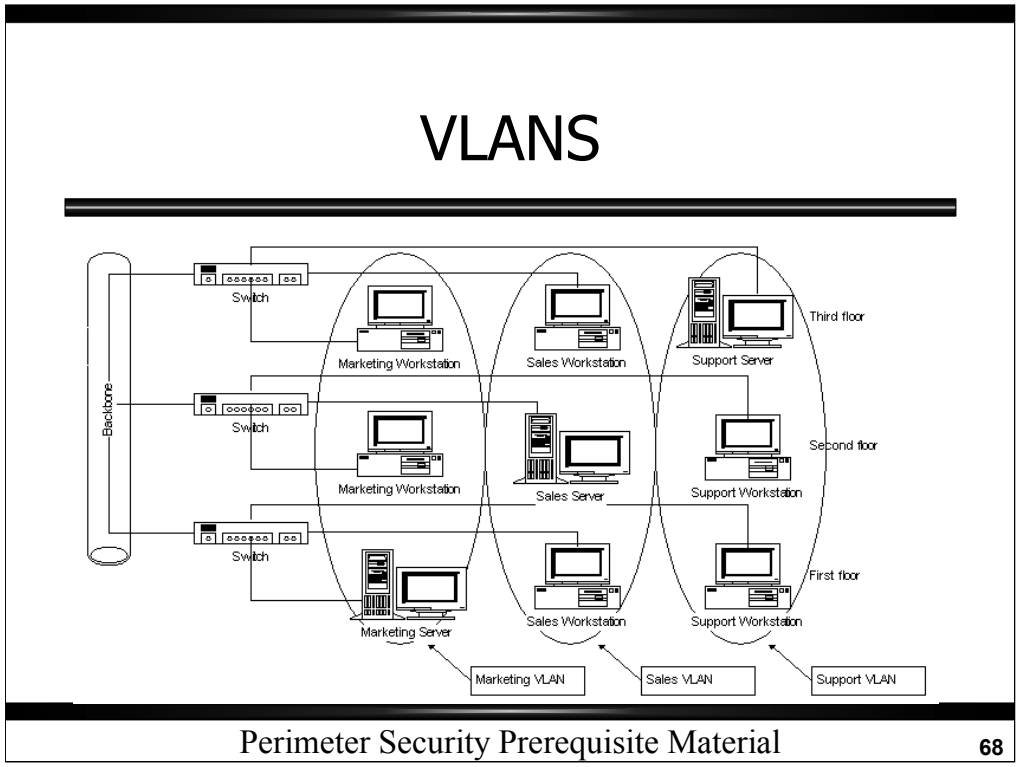


Virtual LAN's (VLAN) allow you to take physical switches and carve them up into multiple logical switches. For example think about taking an axe to a 24 port switch right between ports 14 and 15. This would give you two switches, one with 14 ports an the other with 10. VLANs allow you to do the same thing but without the mess.

Many high end switches allow you to trunk them together and span VLANs across switches. For example VLAN2 could consist of ports 2-5 on the first switch and ports 16-23 on the second switch. This allows you to group devices which may be located in geographically separated areas.

When you create VLANs, you need some way of tying them together as traffic on one VLAN will be isolated from traffic on another. Typically routers are used to connect VLANs. This could be an integrated router (discussed later) or a separate device.

Note that VLANs are a management/performance issue, they are not a security solution. Do not deploy VLANs on both sides of your firewall. You are creating a situation where someone could potentially circumvent your firewall.



This figure shows an example of how VLANs can be used to group geographically separated systems. VLAN technology would allow us to create a different logical network for each of our workgroups. This would provide traffic isolation and better performance. For example, traffic going to and from the Support server would not be seen by the two workstations on the same floor because they are each part of different logical networks. Of course a device such as a router would be needed to tie these three logical networks together.

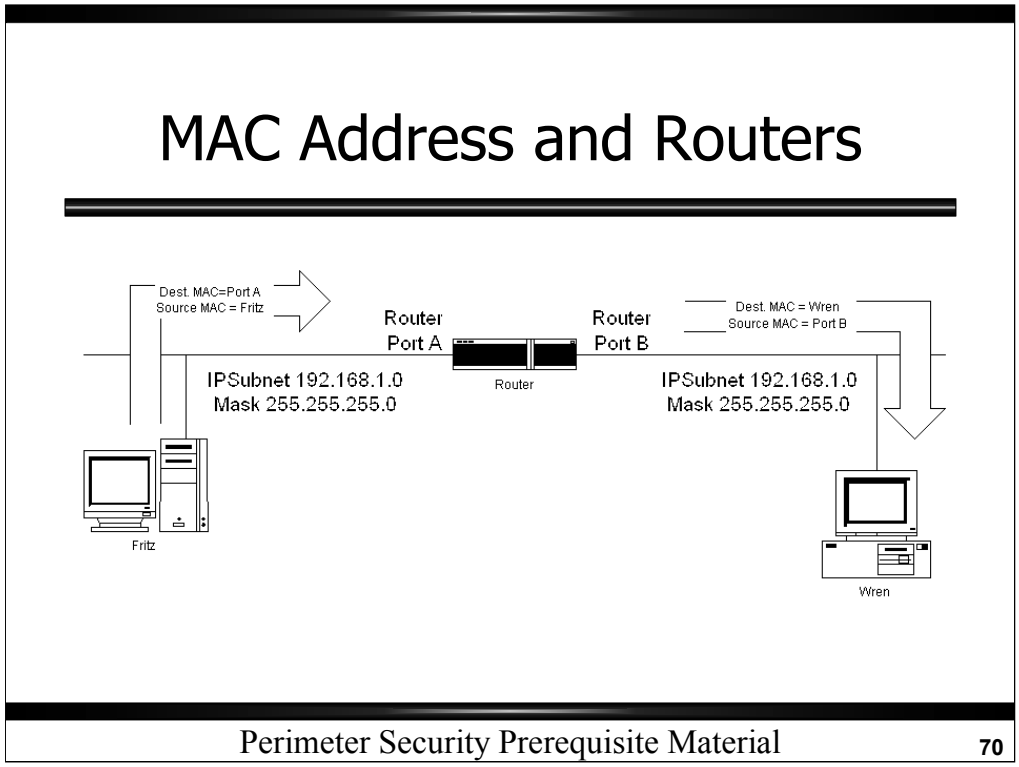
The only problem with VLANs is that administration can become difficult if you do not keep accurate records. For example what VLAN is wall jack 53A connected to? Without accurate record keeping, you may find yourself reviewing the switch's configuration files just to figure out where you can plug in a system.

Routers

- Communicate just like any other station
- Maintain tables with routing information that point to all reachable networks
- Forward or block traffic based on the destination network address (software)
  - Drop all frames to an unknown destination
- Block all broadcasts

Routers are considered to be perimeter devices as they connect together logical networks. A router resets the maximum distance rules for a topology so they can be used to extend networks over great distances. Much of the Internet is built on router technology.

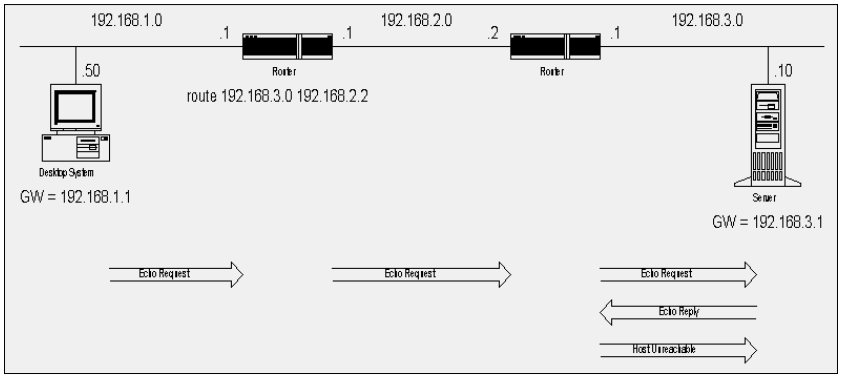
Routers are excellent at traffic control. Unlike a bridge which will forward traffic to an unknown destination by default, a router drops traffic to unknown destinations. You need to specifically program them to do so (default route). Also, routers block local broadcasts providing excellent traffic isolation.



Some people become confused when they try to wrap their brain around OSI layer 2 addressing versus OSI layer 3 addresses.

MAC addresses are a local phenomena. When a frame is received by a router, the Ethernet header and trailer gets stripped off. This packet then gets a new Ethernet header and trailer and is transmitted back out onto the network. The packet header information (such as the source and destination IP address) is not changed.

Routing is Two Sided



Perimeter Security Prerequisite Material 71

While this is more troubleshooting than performance, remember that routing is a two sided process. Not only must the data request make it to its intended target, but the reply or acknowledgment must have path information to make it back to the original host.

Take a good look at the traffic flow in this diagram. To the transmitting host, it appears that it is unable to reach the target server. The truth is the traffic is making it to the server, the replies are just not making it back. From the perspective of the transmitting host, you can not determine if the problem is with the data request or the returning reply. The symptoms look identical.

When troubleshooting connectivity problems, remember to look at the whole picture.

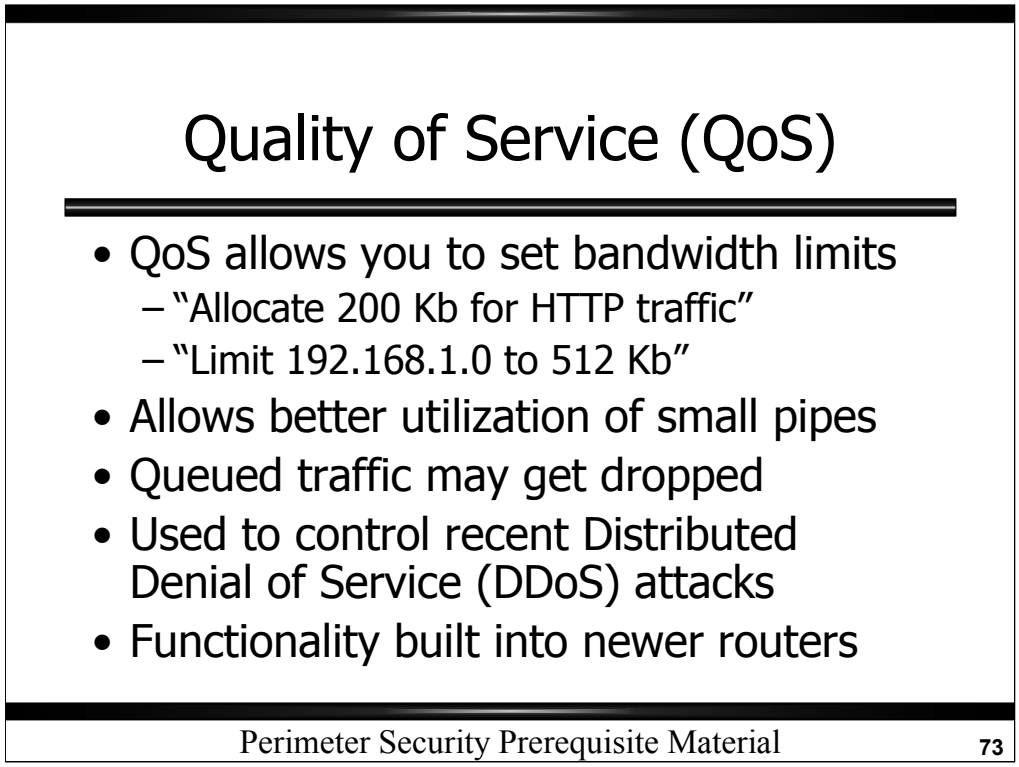
Bridge vs Router

A bridge or switch:	A router:
Uses the same network address off all ports	Uses different network addresses off all ports
Builds tables based on MAC addresses	Builds tables based on network addresses
Forwards broadcast traffic	Blocks broadcast traffic
Forwards traffic to unknown addresses	Blocks traffic to unknown addresses
Does not modify frame	Creates a new header and trailer
Can forward traffic based on the frame header	Must always queue traffic before forwarding

This chart shows a quick summary of the differences between bridging and routing technology. The two most important points are:

1) Routers block by default while bridges forward

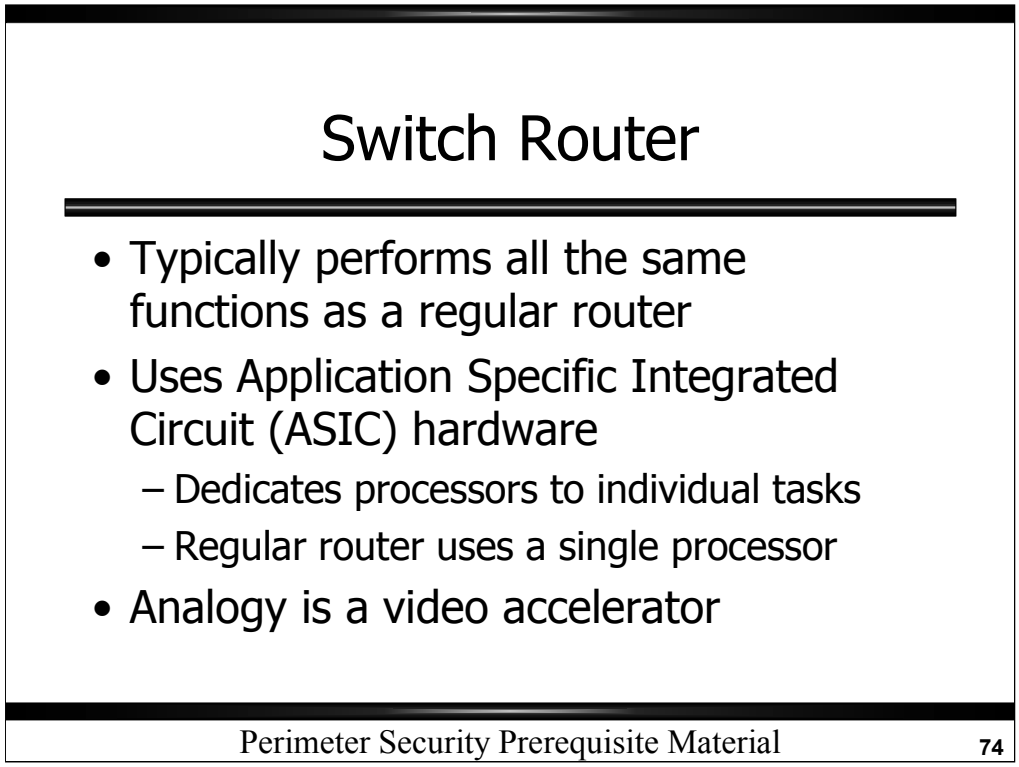
2) With a bridge, all ports connect to the same logical network (192.168.1.0/24). With a router, each port typically connects to different logical networks (port A to 192.168.1.0/24, port B to 192.168.2.0/24)



A useful feature implemented on many modern day routers is quality of service (QoS). QoS allows you to define how much bandwidth different traffic patterns are allowed to consume.

For example Yahoo used QoS to help recover from the recent DDoS attacks. The attacks came in the form of Internet Control Message Protocol (ICMP) traffic. Yahoo's Web server serve up pages using TCP port 80. What Yahoo did is ask their Internet Service Provider (ISP) to limit ICMP so that it could never consume more that a certain percentage of the available bandwidth (say 2% just to throw out a number). When ICMP traffic reached the 2% mark, additional ICMP traffic was discarded. This left 98% of the bandwidth free for TCP 80 requests thus allowing their customers to connect to the Web site with little to no degradation in performance.

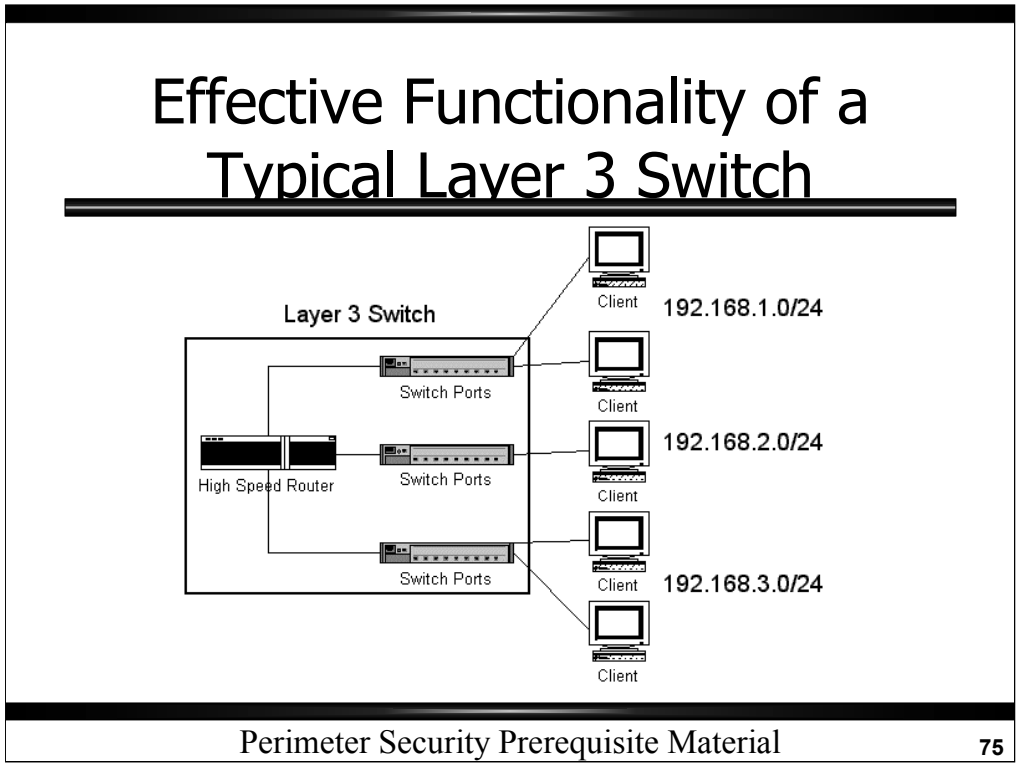
If you have a small WAN connection (say 56K), QoS can be used to prioritize traffic to keep performance at acceptable levels.



There has been a lot of marketing hype as to what exactly is a switch router. In fact the term "switch router" is a bit of a misnomer. Switch routers are more evolutionary than revolutionary.

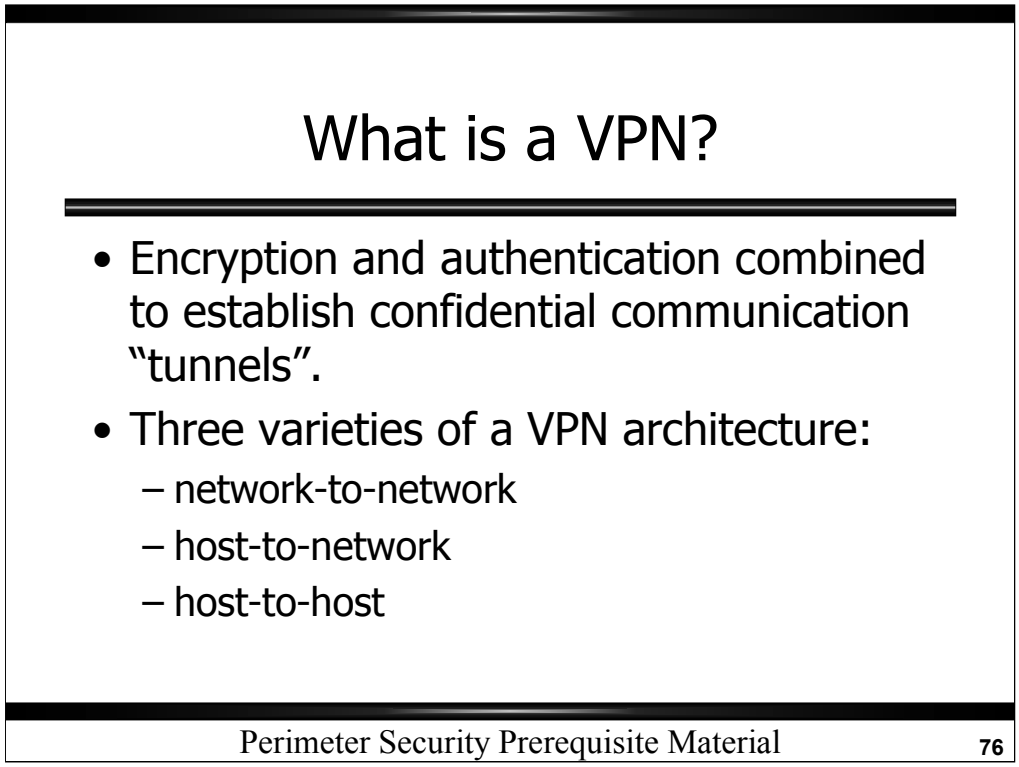
A *switch router* is simply a very fast router. Legacy routers typically use a single processor to perform all routing functions. Switch routers leverage ASIC technology to speed up the routing process. For example a vendor implementation may dedicate one processor to reading and writing frames to the wire, one to stripping and building topology frames and yet another to making routing decisions. All this processing power greatly reduces the lag time that is introduced by the routing process.

As an analogy, think of the modern day video card. Back in the old days all video processing was supplied by the system's CPU. The modern video accelerator card has its own processing chips in order to speed up the display. The result is that the system itself runs faster because the CPU is no longer required to shed clock cycles to video tasks.



To date, I have not seen a switch router which has been deployed as a stand along unit. So far vendors have integrated ASIC routing into their switch products perhaps as an integral part of the switch, or perhaps as an optional expansion card.

In either case, the functionality is the same. The ASIC router acts as a backbone to multiple switch ports. Each port group is defined as a unique VLAN and then these VLANs are tied together using the ASIC router. The result is higher performance as frames of data are moved through the network. Because of its high speed (and price), switch routing lends itself to the network's core or backbone rather than the network's perimeter. For example it would be a waste of high speed parts to deploy a switch router on your perimeter to tie in a T1 WAN connection to a remote network. Legacy routers are more than capable of keeping up with T1 speeds. A 100 Mb or 1 Gb network is a different story.



Time to change gears a bit and talk about Virtual Private Networking (VPN) technology.

The explosive use of public networks (e.g. the Internet :) has made development of VPN a foregone conclusion. While trying to explain this (to someone higher in the management chain), I compared it to trying to speak privately across a city bus. You can whisper, but the confounded man-in-the-middle thing will get the better of you. You can both speak in Spanish, but someone will likely be able to decode it. Switch to Swedish and the chances of being decoded are less. VPN and its basis in crypto are all about keeping publicly transmitted messages private.

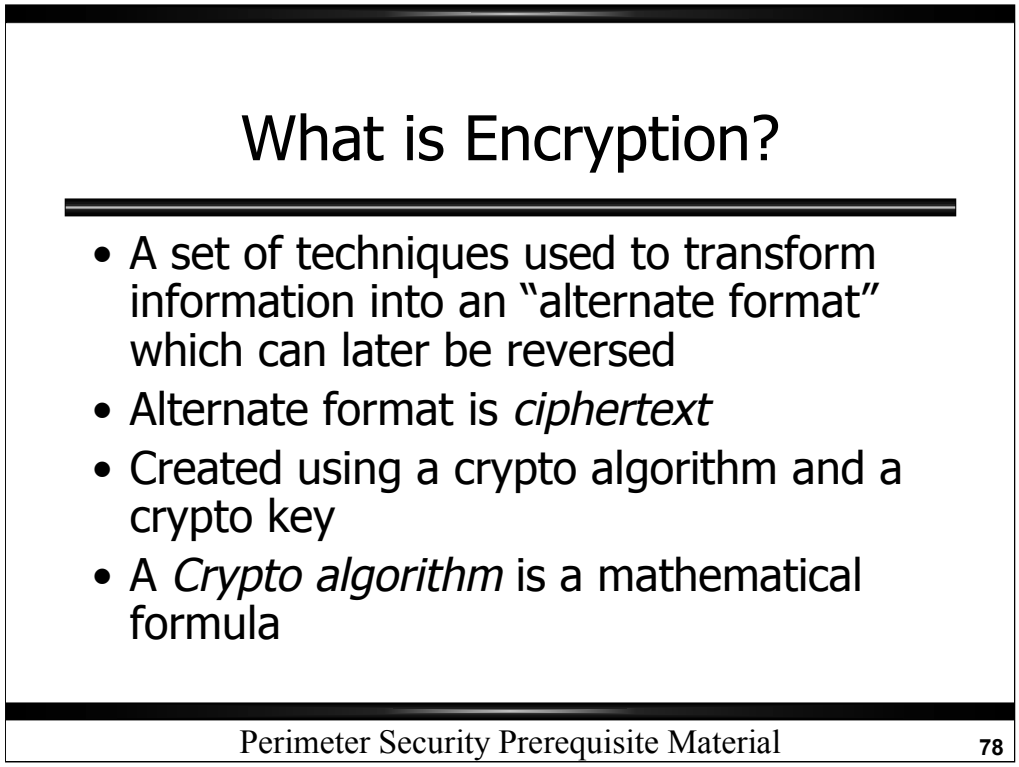
There's a lot more to it than that, and we'll get into specifics throughout the rest of the course.

Today there are three basic methods in which VPN's are deployed: network-to-network, host- to- network, and host-to-host.

Foundations of a VPN

- Cryptography is the VPN foundation:
  - Encryption: Scramble data into something difficult to read without a key.
  - Decryption: the opposite process of encrypting.
  - Authentication: How are you sure you're talking to the right person?

To architect and deploy a VPN, we need to understand how to apply these three tools. These concepts are easy to grasp at the conceptual level, but the devil is in the details as they say. Crypto has evolved from an abstract play ground for mathematicians to something with widespread public awareness (those little solid, gold keys in the browser have people asking the darndest questions). Likewise, authentication is a discipline in its own right.



Cryptography (or crypto for short) is a set of techniques used to transform information into an alternate format which can later be reversed. This alternate format is referred to as *ciphertext* and is typically created using a crypto algorithm and a crypto key. The *crypto algorithm* is simply a mathematical formula which is applied to the information you wish to encrypt. The *crypto key*, also called the *cipher key*, is an additional variable injected into the algorithm to insure that the ciphertext is not derived using the same computational operation each time the algorithm processes information.

So ciphertext is created using three variables:

The information you want to protect

The crypto key

The crypto algorithm

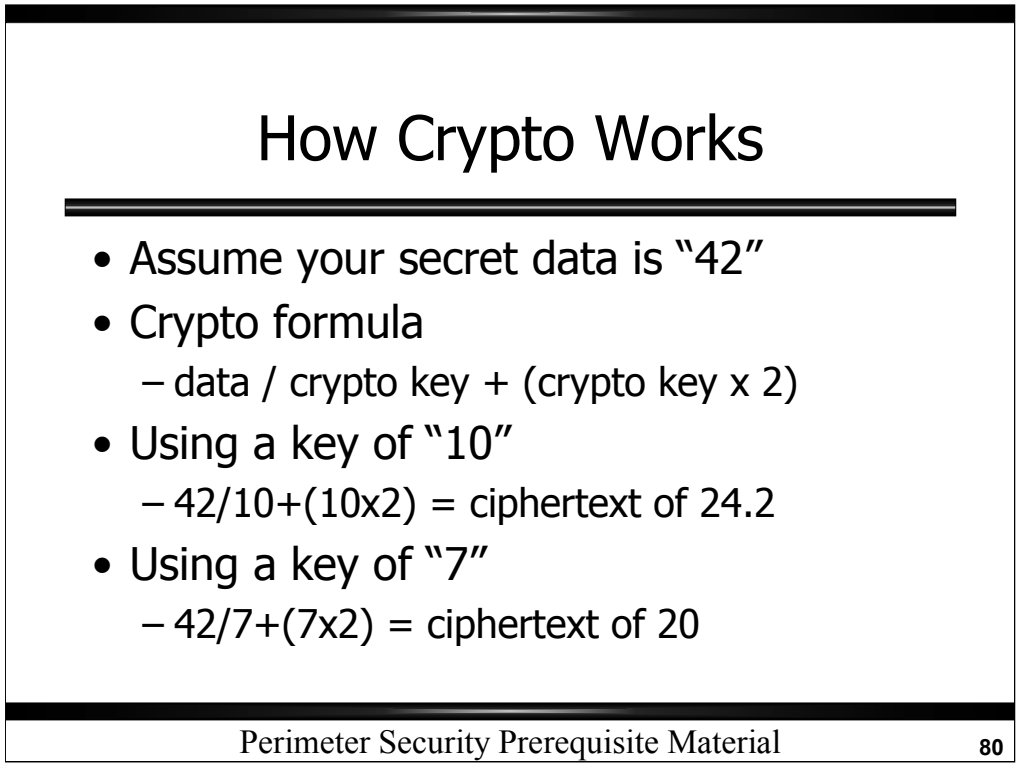
It's important to note that if someone knows two of these variables, they can use them to discover the value of the third variable. For example if I know the crypto algorithm you used and I know the key value, I now have all the info I need to decode the ciphertext and view your private information.

What Does Crypto Look Like?

```
  0  871e f3ba f7e4 34dc a9f5 07a7 0c6d 1934   ......4......m.4
 10  4eb8 da1b c04e 1b79 f6eb 8b7a f578 6a3a   N....N.y...z.xj:
 20  c8c1 5fd5 9c40 4125 35b8 9a41 4df5 7a61   .._..@A%5..AM.za
 30  9c7b dc76 7610 44ca fac8 1476 b659 4194   .{.vv.D....v.YA.
 40  3dde def9 9882 1124 9ad0 efe9 0f28 1437   =......$.....(.7
 50  5d0a b7f3 8a66 be27 bbd2 6a00 e51d 7559   ]....f.'..j...uY
 60  2a1e d1a9 8633 8a5f d95f 52ea 7962 75a6   *....3._._R.ybu.
 70  37b2 2296 da00 957c f2b1 8527 395f 3807   7."....|...'9_8.
 80  8b7d 5b37 7aa4 ed6c 78a7 daaa d958 d9d3   .}[7z..lx....X..
 90  e6f6 7b43 839e a330 41e6 3238 90c2 ba1e   ..{C...0A.28....
 a0  931a 285b 827c 96f8 1995 ac75 2280 465b   ..([.|.....u".F[
 b0  43df fab6 1ab8 835e cd8d 29a7 86cc 1f91   C......^..).....
 c0  77ce cb7e 3870 9752 d8fd 61fb 04e8 d634   w..~8p.R..a....4
 d0  be5c 8666 30c3 c00a c1c1 b396 15d0 8315   .\.f0...........
 e0  37a4 3d1b 7d0a 7a18 6ebc 93c7 ca41 e9e0   7.=.}.z.n....A..
 f0  e9c5 a4d3 d796 4f75 cb9d 648e 9e0f 4b08   ......Ou..d...K.
100  8073 6e5c e70a 4e27 121e ada5 9378 c551   .sn\..N'.....x.Q
```


So what does the data look like when its been converted from plaintext to ciphertext? This slide gives you a pretty good idea. While it may not look like it, this is a partial listing of the files located in the /etc directory on a Red Hat 6.2 system.

The session is an Secure Shell (SSH) connection between two systems. The data was captured using tcpdump. Note that the data lends no indication of what the user is actually doing. They could just as easily be reading e-mail or working on their resume.

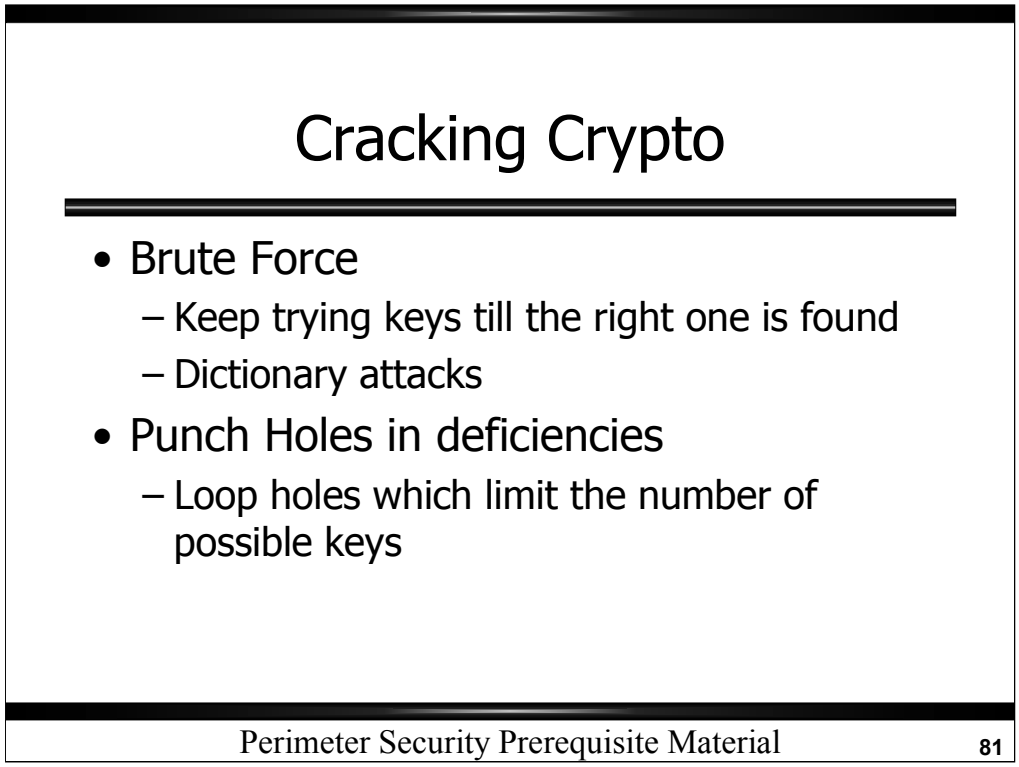


This is a very simplistic example of how cryptography works, but it is sufficient to help you understand the process. The crypto algorithm is simply a mathematical formula. In this case it is:

data / crypto key + (crypto key x 2) = ciphertext

So the Data Encryption Standard (DES), Rivest, Shamir, Adleman public key encryption (RSA) and all the other crypto algorithms you hear so much about are nothing more than a mathematical formula for scrambling information.

Our next component is the "crypto key". This is simply some value chosen to uniquely scramble the data. Since most crypto formulas are well known, you need some method of modifying the formula so people can not reverse engineer your ciphertext and get the value of your data. Note that in the example above changing the key causes the resulting ciphertext to be different.

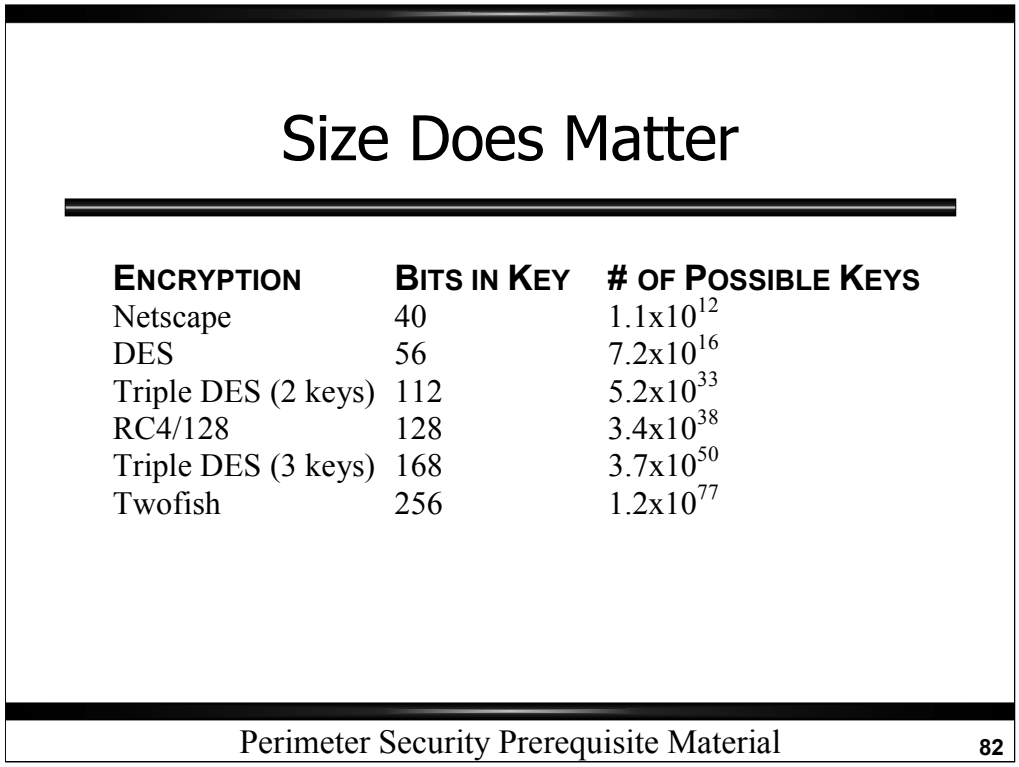


The most common method by which encryption gets cracked is through brute force attacks. While brute force is not elegant, it can be effective if the number of possible keys are small.

Brute force is simply the process of trying all possible key combinations. For example if you know that a given password is four digits in length, you would simple start guessing at 0000 and work your way up to 9999 until the correct key is found.

When a cracker is attempting to guess your password, they typically use what is referred to as a dictionary file. A dictionary file is simply a plaintext file which contains all the words the cracker wants to try as potential passwords. For example L0phtCrack ships with a dictionary file of approximately 35,000 words. Cracking goes much quicker if a pattern match can be found within a dictionary file rather than having to sequentially try different letter and number combinations.

Another option is to find holes in the encryption algorithm. This can be useful in limiting the scope of the number of potential password that the cracker needs to try. For example Windows NT has a weakness by which a cracker can determine if a user's password is less than or greater than eight characters. If the password is less than eight characters in length, the search time can be reduced exponentially.



When dealing with encryption keys, the size of the key plays a very important role in in the security of the data. If the key is too small, the ciphertext can be subjected to brute force attacks.

For example, consider a binary key that is only four bits in length. How many possible combinations are there?

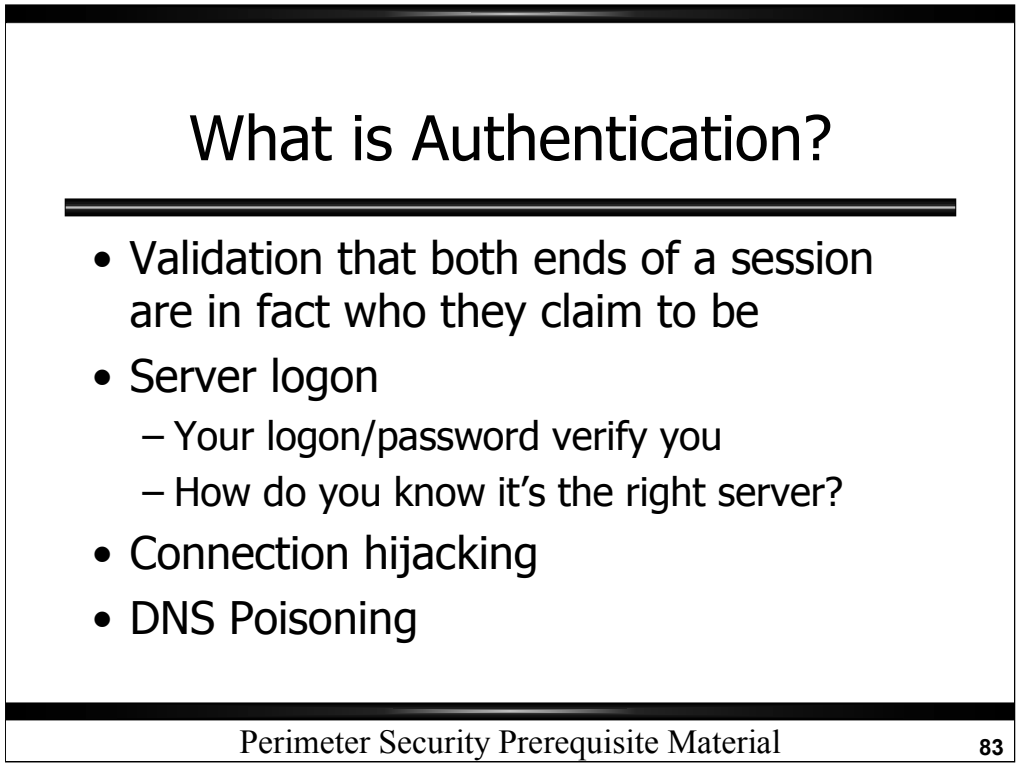
$$2^4 = 16$$

Having 16 possible key combinations would not be that hard to break. Most people could do this manually in less than a minute. So what happens if we increase the key to five bits?

$$2^5 = 32$$

Still pretty easy to crack but look at what happened. Increasing the key length by one bit increased the number of potential keys exponentially. This is why a small change in key size can have a dramatic effect on the number of possible keys. For example check out the difference between 40 and 56 bit. Those extra 16 bits make a big difference!

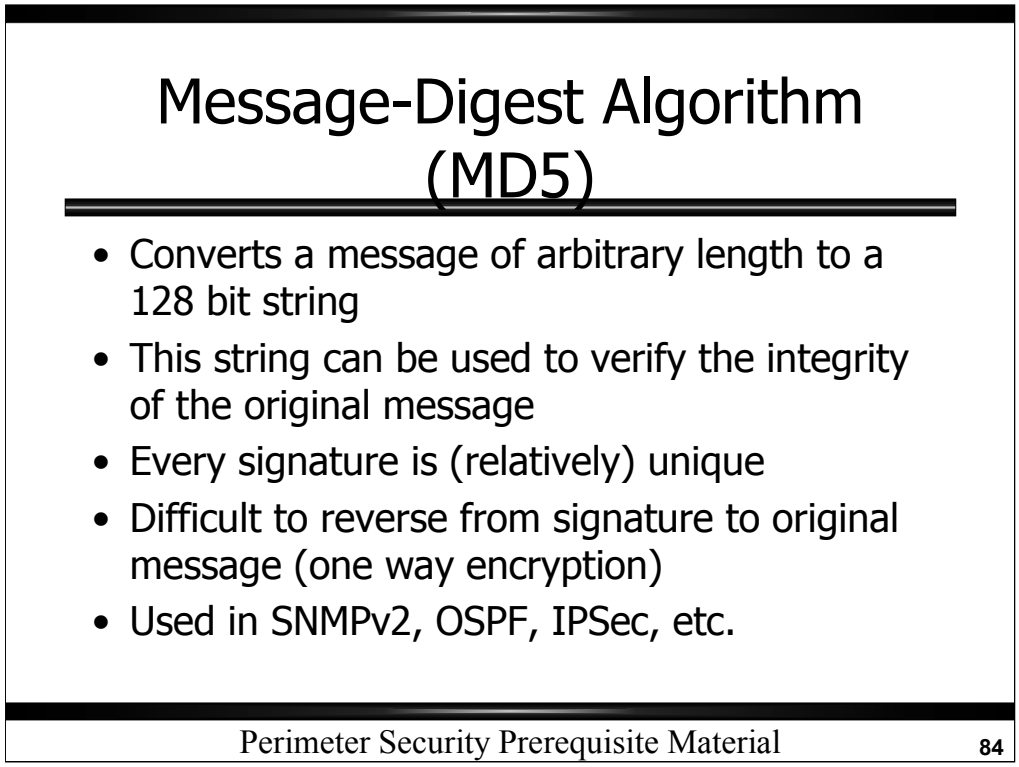
Note that key size is not everything however. The integrity of the encryption algorithm is even more important. For example an algorithm that uses a 256 bit key space would still be considered to be insecure if a deficiency in the algorithm allows a cracker to guess keys by only checking 5 bits of the full key space. This is why you should always use proven algorithms like DES and RSA.



The need for good authentication should be obvious. A service that passes logon information as clear text is far too easy to monitor. Easily snooped logons can be an even bigger problem in environments that do not require frequent password changes. This gives an attacker plenty of time to launch an attack using the compromised account. Also of concern is that most users try to maintain the same logon name and password for all accounts. This means that if an attacker can capture the authentication credentials from an insecure service (such as Post Office Protocol - POP3), the attacker may now have a valid logon name and passwords to other systems on the network, such as Windows NT and Novell NetWare servers.

Good authentication goes beyond validating the source attempting to access a service during initial logon. You should also validate that the source has not been replaced by an attacking host in the course of the communication session. This type of attack is commonly called session hijacking.

Good authentication is also a two way process. You want to validate the server as well as the client. For example if I'm doing on-line banking and transmitting account as well as logon and password information, I want to make sure that this information is in fact going to my bank's web site. I don't want to read in tomorrow's paper that the domain's DNS info was hijacked and I sent all my information to some black hat.



So how can we verify that a message actually came from the person we think it did? One way is by using a *hash* or half of the encryption process to create a digital signatures. In fact the most popular method of creating a digital signature is to use MD5. With MD5, you do not use a cipher key, the original message is simply passed though the MD5 computational process. While MD5 can be considered a form of encryption, remember that once the message is processed it's difficult (impossible) to take the final 128 bit string and return it to its original clear text.

For example:

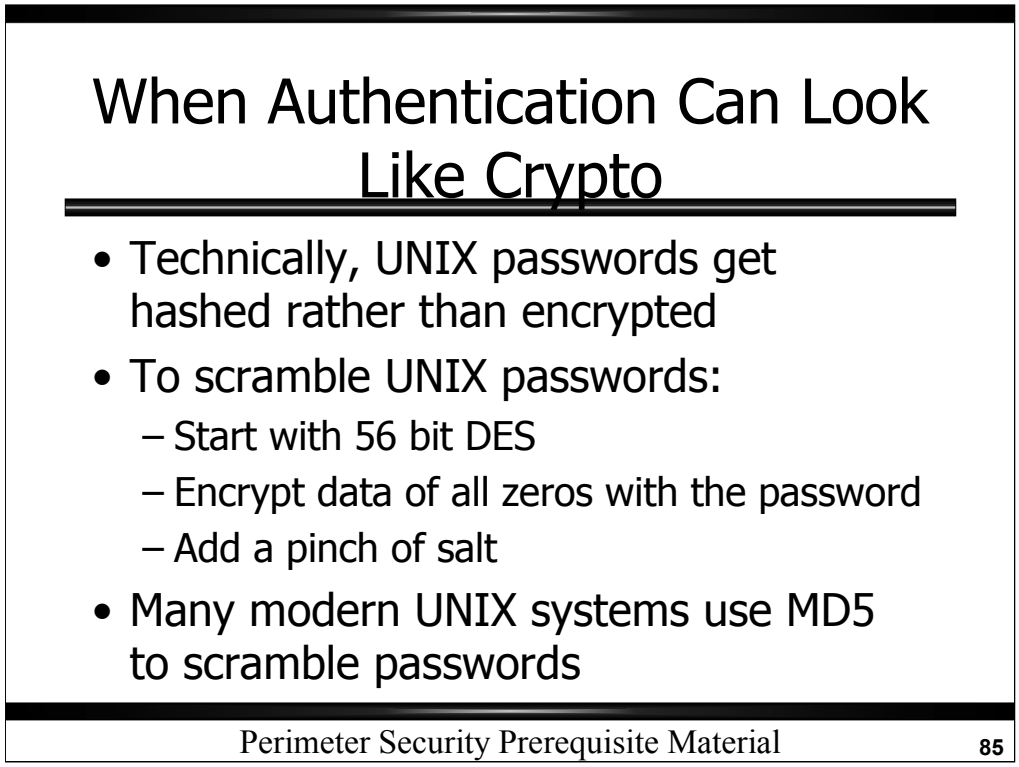
This is a secret message

becomes:

c89cba7b7df028e65cb01d86f4d27077

Since there are no keys involved, I can not take the message digest and easily produce the original text. My only option is a straight brute force attack where I continually try text combinations till the same message digest is produced.

So MD5 is mostly used to create unique signatures. For example Cisco scrambles the secret password within the config file of their routers using MD5. When you enter a password at the enable prompt, the device runs the password you entered through the same MD5 process and compares the results. If they match, the password is considered valid.



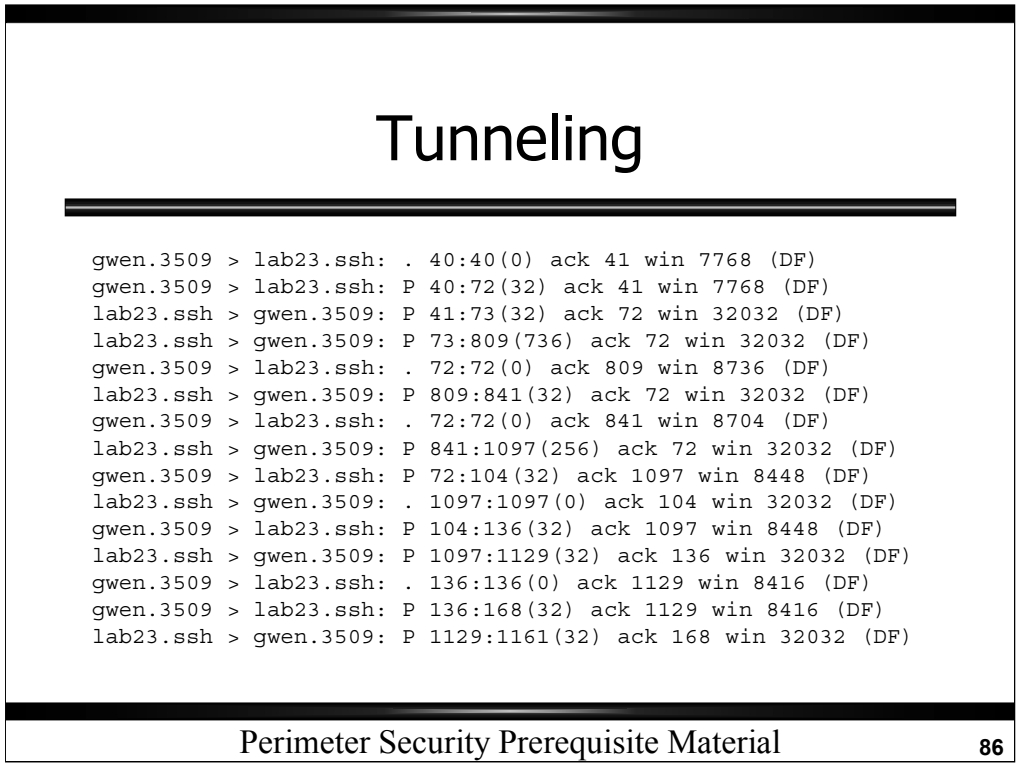
Sometimes what we think is encryption is actually authentication. A good example is the crypto function on UNIX systems. When a user's password is stored, what gets created is a hash rather than ciphertext. The difference is that the hash has no "key" which can be used to easily return to plaintext.

UNIX uses a twist on 56-bit DES, where the plaintext is all zeros and the encryption key is the user's password. The resulting ciphertext is then encrypted again, using the user's password as the key. This process is repeated a total of 25 times. The result is a hash which is not crackable using existing technology.

To make the final ciphertext even more difficult to crack, a second key is introduced known as a *grain of salt*. This salt is based on the time of day and is a value between 0 and 4095. This insures that if two users have identical passwords, the resulting ciphertexts will not be identical.

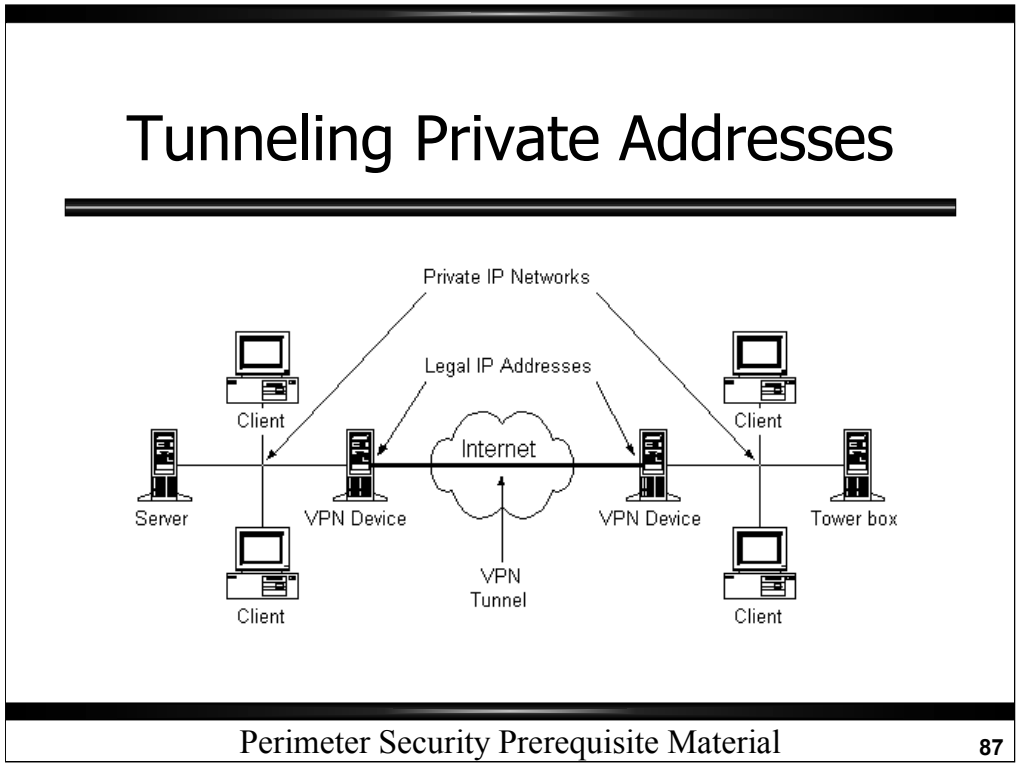
The salt value used to encrypt the password is the first two characters of the ciphertext stored within the passwd or shadow password file. When a user authenticates with the system, the salt is extracted from the ciphertext and used to encrypt the password entered by the user. If the two ciphertext values match, the user is validated and permitted access to the system.

Many UNIX systems have adopted the use of MD5 for creating password hashes. The idea is to ensure that the technology used to protect password strings stays far superior than the technology used to crack them.



Tunneling is the use of one communication protocol as a conduit in order to pass other communication sessions. The conduit sets up a connection between the two end points. You can then use this conduit as a method of connecting to services on either end.

For example, take a look at the attached slide. This shows an SSH session between two hosts, gwen and lab23. What is not obvious from this trace is that the user on gwen is actually retrieving e-mail from lab23 using POP-3. But wait, POP-3 uses TCP/110, right? That it does. What's happening however is that the POP-3 session is being tunneled through the SSH session. The POP-3 TCP headers are actually part of the SSH payload (encrypted of course ;). This improves the privacy of of our communication session as crackers sniffing the wire can not identify which services we are accessing.



A benefit of VPN technology is the ability to allow two networks using private addressing to communicate over the Internet. In the above drawing we have created a VPN tunnel across the Internet. All the traffic passing across the Internet will be using the external IP addresses of the two VPN devices. Since all exposed addresses are legal IP's, there are no routing issues to worry about.

Now what we do is program the VPN device such that all traffic leaving our local network which is headed to the remote network gets passed through the tunnel. If we traced the packet flow it would look something like this:

Local client transmits the packet

Local VPN device receives the packet and encapsulates it in a VPN packet

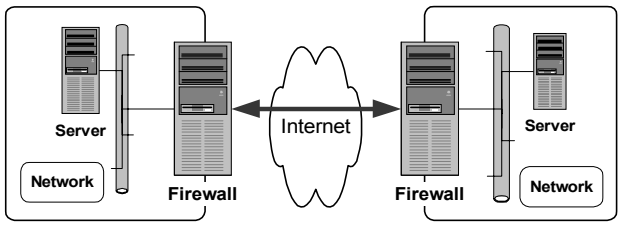
VPN packet is transmitted to the remote VPN device

Remote VPN device receives the packet and removes the VPN encapsulation

Remote VPN device transmits the client's packet to the remote server

So as you can see, our private addressing is never seen on the Internet. We could have used name address translation (NAT) if security were not a concern; NAT can "break" some services during the translation process. By tunneling the traffic, the packets are received on the remote end unchanged.

Network <-> Network VPN

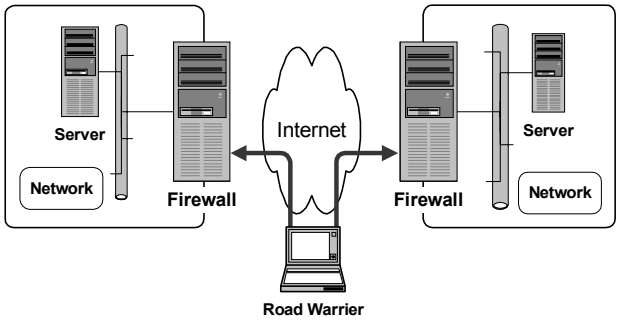


Perimeter Security Prerequisite Material 88

A VPN is implemented in one of three ways, network to network, host to network or host to host.

In a network-to-network architecture, there are two gateways that serve as ingress and egress between to networks. In our diagram, the gateways are firewalls, but this isn't always the case. The gateways may be dedicated devices that run in parallel to the firewall or possibly even behind it via punched through tunnels.

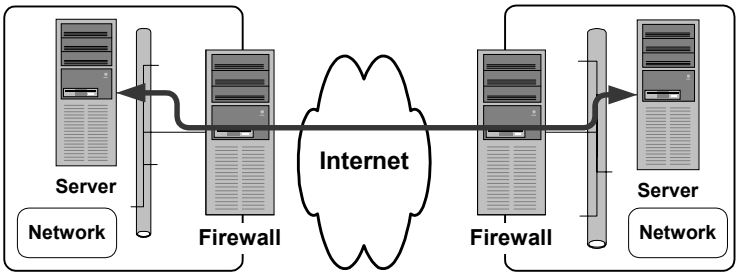
Host <-> Network VPN



Perimeter Security Prerequisite Material 89

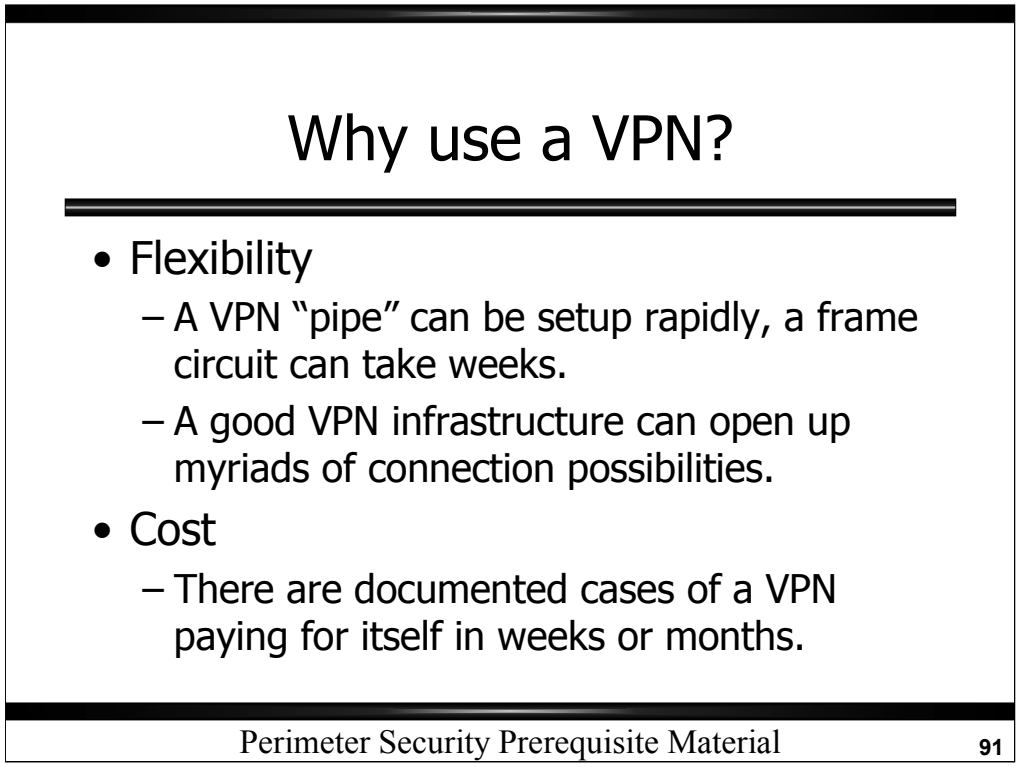
Here we see one of the uses for a host-to-network application. In this diagram we have a laptop using VPN tunnels to communicate to our gateways to gain access to the networks behind them. Both of these illustrations are obviously overly simplified. We're missing the whole dimension of how things are encrypted, authentication and practical implementation issues.

Host <-> Host VPN



Perimeter Security Prerequisite Material 90

This is again a highly simplified diagram of how a host-to-host communication *might* take place. Imagine the scenario where you have two highly secured and trustworthy hosts, but a suspect network between them. An exchange of information needs to take place, and to avoid all the nastiness in-between it's easier to communicate between the two interested parties at a peer level. This could also be used at a client/server level between our laptop user and his mail server for instance.



One of the biggest benefits of VPN technology is their flexibility. Do you need a secure channel between two hosts for only a day? Or just for an hour every business day? A VPN may fit the bill. Once you have the components, setting up a VPN is a software change. This makes the technology far more flexible than legacy frame and dedicated circuits which must be wired and possibly require additional hardware. This flexibility lends itself to creating new business solutions. For example it's not cost effective to wire a T1 for every employee who works from home. It's very practical however to load up software on their laptop and let them connect to the home office via a VPN.

Cost is another potential benefit. With a frame or dedicated circuit, you typically pay a flat monthly fee so even if the circuit goes unused it's costing you money. Also, crossing state and government boundaries with a dedicated circuit only increase their cost. With a VPN, you pay for a local connection to the Internet with no "distance" charges.

Given these benefits, it's not surprising that Taylor & Hecht report that VPN technology is expected to expand 300-1000% by 2003 (Taylor & Hecht).

Useful VPN Features

- Secure access to vulnerable Services like POP-3, Telnet and FTP
- Double or triple authentication (local on laptop, NT, radius)
- Timed access control (User is allowed to connect between the hours of....)
- Access to specific networks or servers

VPNs can be an effective method of locking down insecure services. For example instead of checking mail via POP-3 which passes logon information in the clear, you could tunnel the POP-3 traffic over a VPN connection so all logon information and data remains secure.

Practical Uses of VPN

- Safe and secure access sensitive data
- Remote access by employees, suppliers, contractors, temporary users, etc.
- Quick, flexible business to business capability

Review Questions

- Here are a few review questions to ensure you understand the material
- Please take a few moments to test your knowledge
- If you have questions or problems, please ask during day 2 of the track

True or False?

- Most LAN based topologies use Time Division to share available bandwidth
- Point to point communications are referred to as "multicast" communications
- Ethernet communication rules allow multiple systems to communicate at the same time
- An Ethernet "chunk" of data is referred to as a "frame" while an IP "chunk" of data is referred to as a "packet"

Network/Host Calculations

- Calculate the number of networks and hosts generated by each mask value
  - 255.255.255.224
  - 255.255.255.252
  - 255.255.240.0

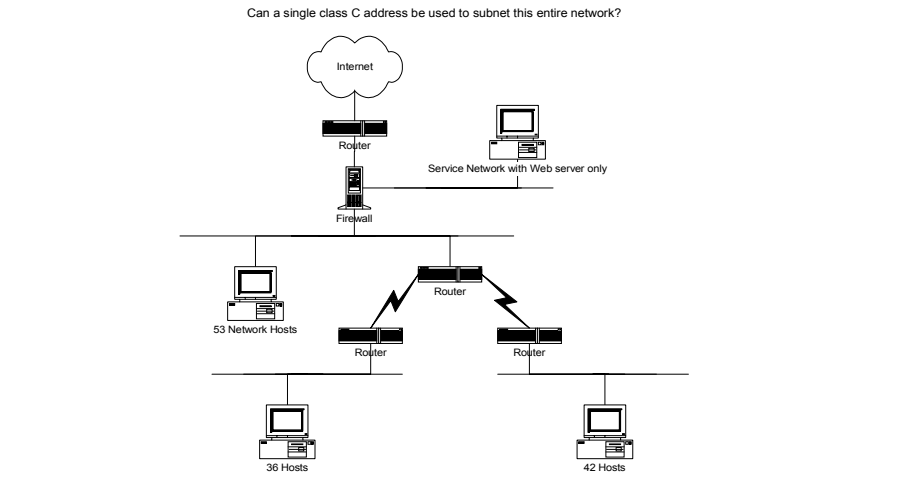
Let's see if you have the hang of it. Take five minutes to calculate mathematically how many networks and hosts are created with each of the above subnet masks. No fair cheating by flipping back through the slides!

Subnet Calculations

- Calculate the following for each address
  - How many network segments?
  - How many hosts per network?
  - What is the network address?
  - What is the broadcast address?
- 192.168.5.233/29
- 192.168.15.32/31
- 172.17.223.107/22

Now let's see if you can live without an IP calculator. In the space below, calculate the requested values for each of the listed IP addresses.

Try Your Hand at VLSM



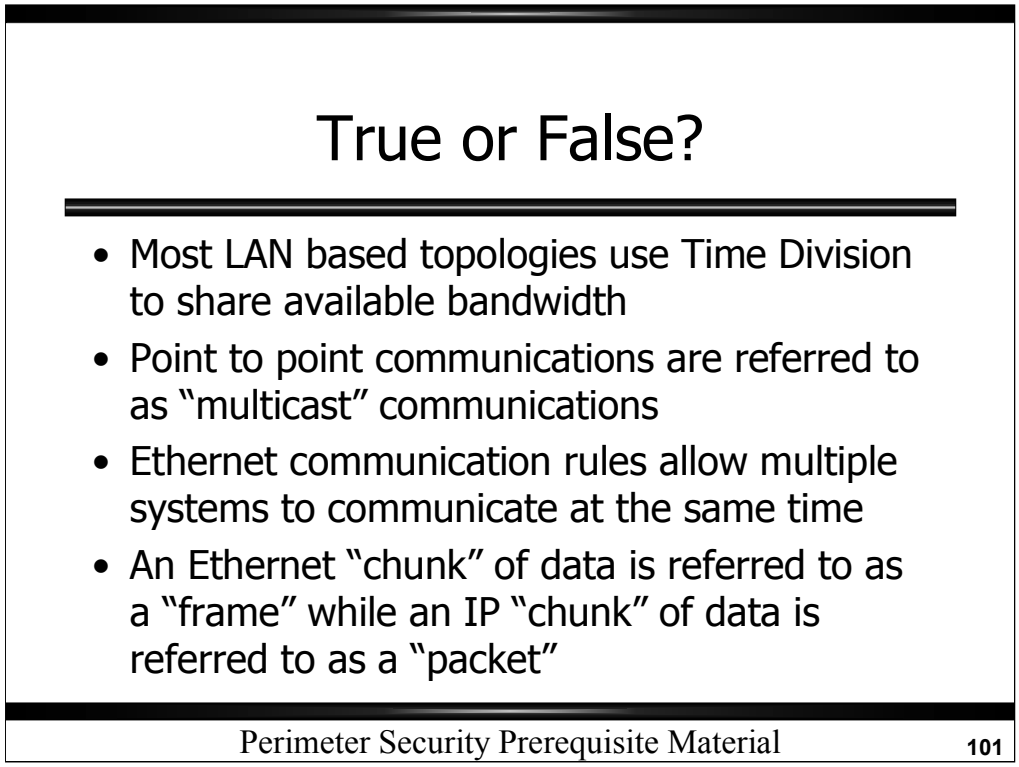
Perimeter Security Prerequisite Material 98

Now its your turn to give it a shot. Given the network space 192.168.1.0/24, use VLSM to address each of the networks shown in the slide.

Some VPN Questions

- All encryption algorithms are susceptible to brute force attacks
- When selecting an encryption algorithm, the primary concern is the key size used
- MD5 is considered to be a trusted encryption algorithm
- Tunneling allows two privately addressed networks to communicate over a publicly addressed network such as the Internet

The Answers



The first statement is false. Most LAN based topologies use a preamble with collision detection or avoidance to share available bandwidth. Time division is typically used in WAN topologies such as T1's and T3's.

The second statement is false. "Unicast" refers to one to one communications while "multicast" refers to one to many communications.

The third statement is false. Ethernet provides a set of communication rules which allow multiple systems to connect to the same medium but only one system is allowed to be transmitting at any given time. The exception is "full duplex mode" Ethernet which require that only two systems be connected to any specific collision domain but each system is allowed to be transmitting at the same time.

The fourth statement is true.

Network/Host Calculations

- Calculate the number of networks and hosts generated by each mask value
  - 255.255.255.224
  - 255.255.255.252
  - 255.255.240.0

Answers:

255.255.255.224

Converting the masked byte to binary, we get: 11100000

number of networks = 2^3 = 2x2x2 = 8

number of hosts = 2^5 - 2 = (2x2x2x2x2) - 2 = 30

255.255.255.252

Converting the masked byte to binary, we get: 11111100

number of networks = 2^6 = 2x2x2x2x2x2 = 64

number of hosts = 2^2 - 2 = (2x2) - 2 = 2

255.255.240.0

Converting the masked byte to binary, we get: 11110000.00000000

number of networks = 2^4 = 2x2x2x2 = 16

number of hosts = 2^{12} - 2 = (2x2x2x2x2x2x2x2x2x2x2x2) - 2 = 4094

Subnet Calculations

- Calculate the following for each address
  - How many network segments?
  - How many hosts per network?
  - What is the network address?
  - What is the broadcast address?
- 192.168.5.233/29
- 192.168.15.32/31
- 172.17.223.107/22

Answers:

192.168.5.233/29 (mask is 255.255.255.248)

Number of networks is 32. Number of hosts per network is 6.

network address =

11101001 = host

\+ 1<u>1111000</u> = mask (.248)

11101000 = 232 or a network address of 192.168.5.232

broadcast address is network address with host bits set high:

11101111 or 239. Full address is 192.168.5.239

192.168.15.32/31 = trick question. No valid hosts per network (2^1-2=0)

172.17.223.107/22 (mask is 255.255.252.0)

Number of networks is 64. Number of hosts per network is 1022.

network address = 11101000.00000000 or 232.0 (172.17.232.0)

Broadcast address = 11101111.11111111 or 235.255 (172.17.235.255)

Answers:

53 hosts - 192.168.1.0/26

42 hosts - 192.168.1.64/26

36 hosts - 192.168.1.128/26

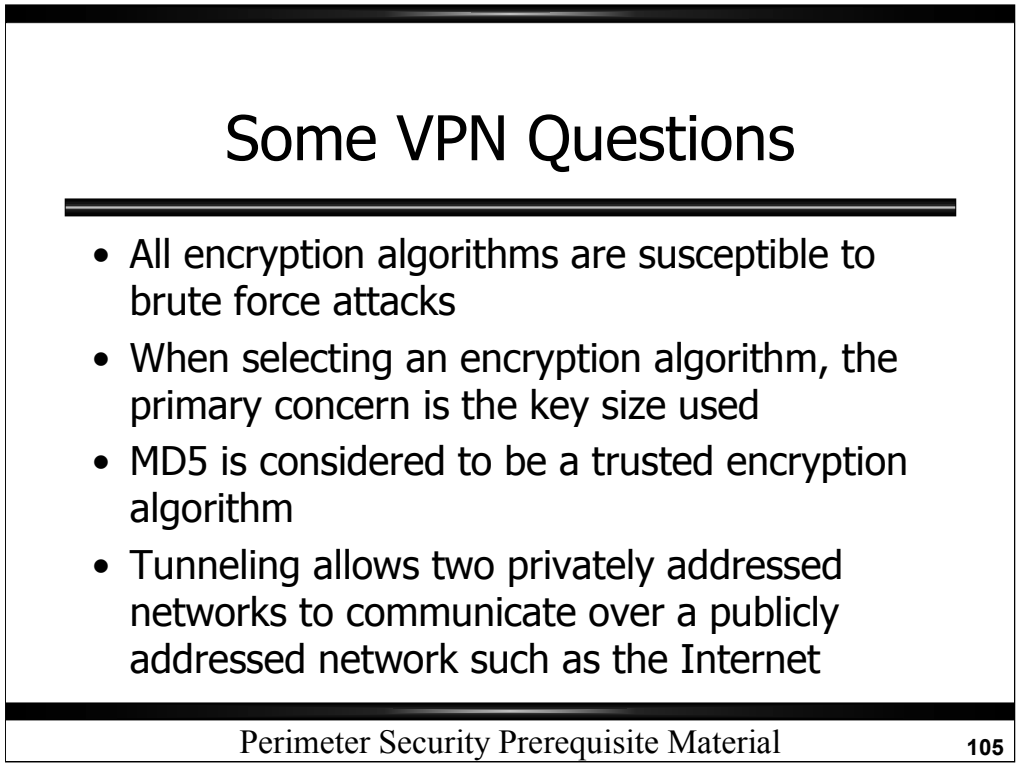
service network - 192.168.1.192/29

(a /30 would work on the service network but /29 leaves room for growth)

internal WAN 1 - 192.168.1.200/30

internal WAN 2 - 192.168.1.204/30

Firewall to router - 192.168.1.208/30



The first statement is true. It is theoretically possible to crack all encryption algorithms by exhaustively trying each possible key. This activity can however be considered to be impossible in a practical sense. Large key spaces require lots of CPU time to test all possible key combinations. If a large encryption key is used, the amount of time necessary to try all possible keys could be hundreds or thousands of years based on current technology.

The second statement is false. The primary concern when selecting an encryption algorithm is whether the algorithm has been thoroughly tested and open to public scrutiny. Proprietary algorithms could potentially contain flaws which allow them to be cracked relatively easily.

The third statement is false. MD5 is an authentication algorithm because the hash generated can not be easily reversed into its original plain text. True encryption algorithms must provide some method of easily returning the cipher text to plain text.

The fourth statement is correct. When tunneling is used, the private addresses are encapsulated within packets that contain legal addressing.