Congestion Avoidance in Computer Networks

John Cox

CONGESTION AVOIDANCE IN COMPUTER NETWORKS

CONGESTION AVOIDANCE IN COMPUTER NETWORKS

John Cox

BIBLIOTEX
Digital Library

Congestion Avoidance in Computer Networks
by John Cox

Ebook ISBN: 9781984664129

Published by:
Bibliotex
Canada
Website: www.bibliotex.com

Contents

1

Computer Network

The primary purpose of a computer network is to share resources:

- You can play a CD music from one computer while sitting on another computer
- You may have a computer that doesn't have a DVD or BluRay (BD) player. In this case, you can place a movie disc (DVD or BD) on the computer that has the player, and then view the movie on a computer that lacks the player
- You may have a computer with a CD/DVD/BD writer or a backup system but the other computer(s) doesn't (don't) have it. In this case, you can burn discs or make backups on a computer that has one of these but using data from a computer that doesn't have a disc writer or a backup system

- You can connect a printer (or a scanner, or a fax machine) to one computer and let other computers of the network print (or scan, or fax) to that printer (or scanner, or fax machine)
- You can place a disc with pictures on one computer and let other computers access those pictures
- You can create files and store them in one computer, then access those files from the other computer(s) connected to it.

PEER-TO-PEER NETWORKING

Based on their layout (not the physical but the imagined layout, also referred to as topology), there are various types of networks. A network is referred to as peer-to-peer if most computers are similar and run workstation operating systems. In a peer-to-peer network, each computer holds its files and resources. Other computers can access these resources but a computer that has a particular resource must be turned on for other computers to access the resource it has. For example, if a printer is connected to computer A and computer B wants to printer to that printer, computer A must be turned On.

A computer network is made of two distinct subsets of components

- Distributed applications are programmes running on interconnected computers; a web server, a remote login server, an e-mail exchanger are examples. This is the visible part of what people call “the Internet”. In this chapter we will study the simplest aspects of distributed applications. More sophisticated aspects are the object of lectures called “Distributed Systems” and “Information Systems”.

- The network infrastructure is the collection of systems which are required for the interconnection of computers running the distributed applications. It is the main focus of this lecture.

The network infrastructure problem has itself two aspects:

- Distance: interconnect remote systems that are too far apart for a direct cable connection
- Meshing: interconnect systems together; even in the case of systems close to each other, it is not possible in non-trivial cases to put cables from all systems to all systems (combinatorial explosion, cable salad management problems etc.).

The distance problem is solved by using a network, such as the telephone network with modems. The meshing problem was originally solved easily because the terminals were not able to communicate with each other, but always has to go through a main computer. The mesh in such cases is reduced to a star network. Today this is solved by a complex set of bridges and routers.

HISTORY OF NETWORK

Prior to the widespread inter-networking that led to the Internet, most communication networks were limited by their nature to only allow communications between the stations on the network. Some networks had gateways or bridges between them, but these bridges were often limited or built specifically for a single use. One prevalent computer networking method was based on the central mainframe method, simply allowing its terminals to be connected via long leased lines. This method was used in the 1950s by Project RAND to support researchers such as Herbert Simon, in Pittsburgh, Pennsylvania, when collaborating across the continent with researchers in Santa Monica, California, on automated theorem proving and artificial intelligence.

Earlier, computers were only used as stand-alone machines. Computer networks were created to establish a communication link between different users within an organization. The Advanced Research Projects Agency (ARPA) designed Advanced Research Projects Agency Network (ARPANET) for United States Department of Defence. It was the foremost computer network in the Universe.

Small computers were used to manage communication links. These small computers were connected to large mainframe computers. These large computers were connected to the ARPANET. The purpose of ARPANET was that each computer would be connected to a specialized computer, called as Interface Message Processes (IMP). The IMPs perform the store and forward of data and were connected to each other using modems.

In 1969, the first ARPANET link was established between the IMP nodes at the University of California at Los Angeles (UCLA). Then Douglas Engelbart's hypertext-project computer at Stanford Research Institute (SPi was connected to IMP node. By the end of the year, the computers at the University of California, Santa Barbara (UCSB) and the University of Utah were connected to the network. All the computers in the network were using different operating systems and were able communicate with each other across the network.

ARPANET was originally developed for providing long-distance (remote) computing. Remote computing was done by a utility programme known as Telnet which allows the user on one computer to connect to another computer on a network. Telnet service can be used for debate, discussion and news sharing by the news discussion groups. Still the remote computing is considered as an effective and useful service of the APRANET.

Explosive Internet growth in the late 1990s dramatically affected the evolution of computer networking. Some new network technologies and initiatives boomed but quickly faded into oblivion. Others have stood the test of time.

HOME BROADBAND

High-speed home networking struggled to get off the ground in 1997 and 1998. Cable modem was the first broadband option available to many, but only a few hundred thousand subscribed to Internet cable in that first year. In 1999, competition from DSL kicked in, but DSL availability remained quite limited at first.

The expected competition from satellite services did not emerge until later, and even today, satellite services remain a distant third in the home broadband market. It took until 2001 for home broadband to enter mainstream usage and begin growing at a faster rate than Internet dial-up services. Although the networking industry continues to promote broadband as the future pathway to new and exciting Internet applications, tens of millions of U.S. households remain on dial-up. The spirited battle between cable and DSL also continues.

Although many in the industry remain disappointed in the slow adoption rate of home broadband, initial concerns over:

- Reliability of DSL,
- Security of cable modem,
- Broadband accessibility in rural areas, and
- Viability of the broadband service providers, have all largely been addressed.

The future of home broadband appears quite promising.

Napster and Peer to Peer

A 19 year-old student named Shawn Fanning dropped out of college in 1999 to build a piece of software called Napster. Within a few months, Napster became one of the most popular software applications of all time. People all over the world regularly logged into Napster to freely swap music files in the MP3 digital format. Some proclaimed Napster "revolutionary." It certainly created a large stir in the industry press.

Users invested large amounts of time and energy in Napster. They also consumed big chunks of network bandwidth. Some universities and businesses reacted by banning or blocking Napster to keep their networks stable, generating even more controversy.

Napster was built using a network design technique called peer-to-peer (P2P). Though peer networking had existed for decades, Napster generated a new wave of interest in P2P. Many startup and some established companies jumped on the P2P bandwagon, activitely promoting new or rehashed, generally unproven business opportunities based on this technology.

Both Napster and corporate P2P have rapidly faded into obscurity. Napster faced the wrath of the music industry establishment, who claimed that open music file sharing violated copyright laws. The legal process moved slowly, but eventually the courts shut Napster down. Corporate interest in P2P suffered a similar fate. The allure of Napster proved to be its openness, not its network architecture, and initiatives to create comparable paid services have all struggled mightily to get off the ground.

Apache

Score one for the Open Source movement. Since 1996, and despite formidable competition from the likes of Microsoft, Apache has remained the world's most popular Web server by a wide margin. Web site owners frequently choose Apache for its reliability, performance and zero cost. Apache works well not only for "mom-and-pop" sites but also supports some of the busiest Web sites on the Internet.

Today, Apache has expanded well beyond its original roots as a mere HTTP server to support numerous new Internet technologies including Web Services. Apache should remain a key Internet technology for years to come.

NETWORK SERVER

The word "server" often appears in discussions concerning the Internet (Web server, mail server and chat server for example) and company networks (print server and document server for example).

But how does a server differ from the computer sitting beneath your desk? What are servers used for and how might you benefit from using one?

A server is a computer, often a mid-to-high end system, which offers some kind of service or resource to a number of users, most often over a network. A server differs from a desktop system in that a server usually provides a small number of specific services to its users. Desktop systems are designed for everyday use, allowing users to perform a large number of tasks, from checking e-mail or writing documents, to watching streaming video or archiving data.

Servers are used to support a number of users who may need to share a single service or resource. This resource is placed in a single location, on the server, allowing for easy configuration and management, while also allowing for access by a large number of users. Take, for example, a Web server. A Web server can hold a multitude of information regarding an organization, such as product information, contact information, investor information, and support documents, just to name a few. Because all of this information resides in a single place, you can easily change, update, and manage it, while allowing any number of users to access that information by simply visiting the Web site.

However, there are many other uses for servers outside of Web hosting. Servers can be used for data storage or archival, for running centralized applications, for sending and receiving mail, or even as simply an efficient way to organize and manage a network infrastructure.

HP and Compaq offer a wide range of server solutions to meet your organization's needs, from small Web hosting solutions to mid-sized data storage to scalable cutting-edge critical systems and beyond.

- *Compaq Proliant Servers*: comprehensive computing systems that include all the necessary components — processor, memory, and network connections — on a single plane called a blade. These systems are designed to reduce space requirements in high-density computing centers while offering a rack-mountable system that utilizes shared components, such as cabling, power-supplies and cooling fans.
- *HP Tower Servers*: offered in standalone and rack-mountable versions and with 1 to 6 CPUs per server, these systems are designed to accommodate a large amount of internal storage and I/O expansion. These systems are an excellent server solution for small-to-medium organizations.
- *Compaq Alpha Servers*: high performance systems for Tru64 Unix, OpenVMS and Linux, offering entry-level, mid-range, and high-performance Alpha architecture solutions, with 1 to 32 processors. These systems are available in both rack-mount and desktop configurations.
- *HP Rack-optimized Servers*: high availability servers with a unique space-saving design, offering 1 to 16

CPUs per server and available in a number of different architectures. These systems offer robust service and management features in a scalable package.

- *HP Super-scalable Servers*: ideal for large-scale applications and databases, these servers are extremely powerful and flexible, offering high-available solutions with 1 to 64 CPUs per server. These servers are designed to meet high-end demands.
- *Compaq NonStop Integrity Servers*: providing the highest levels of availability and scalability for critical applications, these servers include fault tolerance reliability, data integrity, and continuous availability.

VIRTUAL PRIVATE NETWORK (VPN)

A VPN or a Virtual Private Network is a dedicated communications channel which is streamlined using another network. A VPN is useful for companies which need their own space on the internet.

For example a business community that needs to secure its own network on the internet and wants to carry out various business activities within this environment can use a VPN.

The VPN operates on a different and more complicated topology than a point to point network. Also VPN dos not stress on security features and rather the network on which it is based may tend to have extreme security features.

Virtual Private Networks have gained a lot of popularity in the recent past because it not only reduce costs but also increases and expands the network capabilities. Because of the functionalities of VPN, employees can connect from anywhere in the world to the company's network and carry out important and sensitive task on time. It keeps the employees connected to the company's intranet.

WHAT IS A VIRTUAL PRIVATE NETWORK

A Virtual Private Network is a system that keeps the computers connected even if there are long distances or physical barriers. The VPN is categorized as a form of a wide area network. The VPN supports functions like remote access for clients; it can network one LAN to another LAN, and also keeps the functions limited to the intranet.

Companies find it easier to use VPN because it needs to give special access or create special access for their employees who have to be mobile. It is globally active intra network so employees from anywhere can connect to the company's intranet

VPN FUNCTIONALITY

VPN supports many protocols like PPTP, L2TP, IPSec and SOCKS. These protocols help the authentication process in the VPN. The VPN clients can establish a connection and identify the authorized people on the network. The VPN networks are also encrypted which means that they are usually invisible on the bigger networks. Being encrypted enhances the security feature for the VPN. The Virtual Private Networks also have paved the way for Wi-Fi and private wireless connections networks. More and more technologies are basing their developments in the VPN because of the mobility it provides.

VIRTUAL PRIVATE NETWORK FEATURES

Virtual Private Networks are cost effective solutions for large business organizations instead of dedicated network facilities It increases the mobility for organizations by instantly connecting home networks or mobile workers to the organization. Security features can be customised.

VPN Disadvantages

By providing access to employees globally the security is at risk. This also puts the sensitive information of the company accessible globally. For a VPN extra care has to be taken and the security should be clearly defined.

VN for remote access can be set up for employees who are constantly travelling and have to stay connected to the company's intranet. Even when the remote access is established the employee or the person have to connect top the local internet service provider first in order to be able to connect to the virtual private network of the organization. That is why the VPN is known to work on an underlying network and cannot work independently.

VPN Solutions

There are many solutions that a VPN can provide like long distance connectivity, file sharing, video conferencing and other services related to networks. The features that are already being offered on the internet through various services can be offered through VPN in perhaps a cost effective way. The VPN is so flexible that it can work on private networks and public networks.

2

Networking Topologies

DEFINITION

Network topology refers to the way that your computer network is arranged. The network can have a physical or a logical topology.

The physical topology describes the layout of computers and where the workstations are positioned. The logical network topology describes how the information flows through the network.

Choosing your physical topology is important because if it is not chosen correctly, this could cause your network to not operate properly. There are several terms that describe the type of physical topology that a network can have. The most common topologies are bus, ring, star, and mesh.

In a bus topology, all of the computers are attached to a single cable using terminators. The terminators work to absorb the energy from the signals in the network. The bus topology is easy to install, but it is not reliable because a single default can bring down your network.

In a ring topology, each computer is connected directly to two other computers on the network. As with a bus topology, a single fault can disrupt a ring network. This type of network does have advantages, however, and does not require a network server. In a star topology, each computer is connected using its own separate cable. This set up is more reliable than a bus topology because its design makes it fault tolerant and susceptible to errors. The information on the network is transmitted from one system to another and the data flows in one single direction. The topology is expensive to maintain and is not reliable because removing one computer can disrupt your entire network.

In a mesh topology, a path is present from one computer to another computer in the network. The mesh topology is usually used in internet structure. The mesh topology can be complicated to construct because it has multiple connection between locations. For every computer that you have you will need at least one and a half connections for each one. This contributes to the expensive structure.

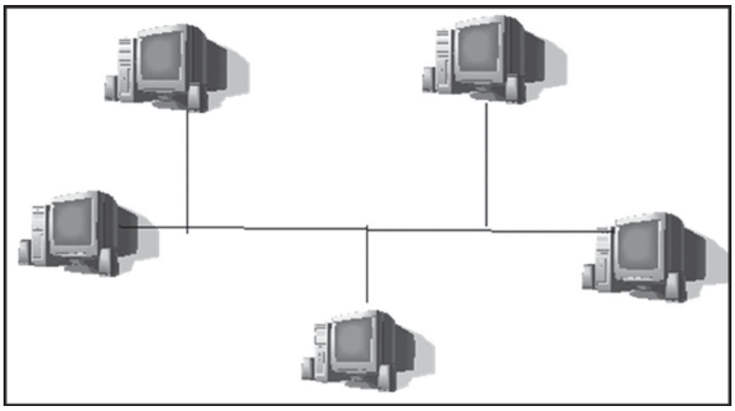
In choosing a network topology, understand that each one has its advantages and disadvantages. Research each one and see which one fits within your budget and which one fits your network layout. Also consider the amount of maintenance required in each topology. When setting up a network at home, consider a network topology that is simple and easy to maintain. You will also want a topology that is inexpensive to set up. If you are choosing a topology for a business, you will want to consider a topology that is reliable and resistant to errors. Your customers will need a network that is free of downtime; therefore, be sure that your topology is one that withstands disruption.

TYPES OF NETWORKING TOPOLOGIES

BUS TOPOLOGY

A bus topology is a method of transmission on networks that uses a common vehicle for transmissions and thus it is categorized as shared communication. Imagine a bus picking up various people from one stop and dropping of people as it travels and then picking a few more. That is what happens in a bus network exactly.

A bus network uses a multi-drop transmission medium, all node on the network share a common bus and thus share communication. This allows only one device to transmit at a time. A distributed access protocol determines which station is to transmit. Data frames contain source and destination addresses, where each station monitors the bus and copies frames addressed to itself.



A bus topology connects each computer (nodes) to a single segment trunk (a communication line, typically coax cable, that is referred to as the 'bus'. The signal travels from one end of the bus to the other. A terminator is required at each to absorb the signal so as it does not reflect back across the bus. A media access method called CSMA/MA is used to handle the collision that occur when two signals placed on the wire at the same time. The bus topology is passive. In other words, the computers on the bus simply 'listen' for a signal; they are not responsible for moving the signal along.

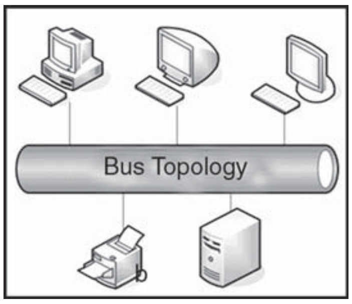
However in a Bus topology only one device is allowed to transmit at a given point of time. The DAP or the Distribute Access Protocol has the information about which station has to transmit the data. The data that is being transmitted have frames that will have the source name and the network address.

Function of Bus Topology

The bus topology connects each computer on the network into something called the segment trunk. A bus is usually referred to a cable that connects end to end and this is used to transmit the signals from one end to the other end. At every end a terminator is placed so that it understands in which direction the data is traveling and also the terminator is used to absorb the signals.

If the terminator does not absorb the signal then the same signal is reflected back to the bus confusing the whole data flow.

The bus is considered to be a passive network because the computers are largely dependant on the signal that is being transmitted.



Advantages

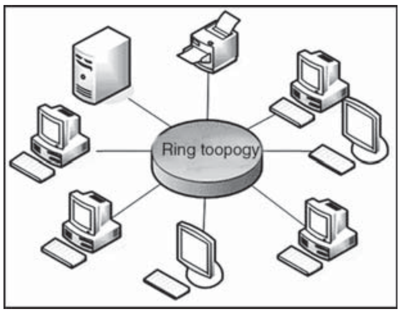
The advantage of the Bus network is that if one computer fails in the network the others are still not affected and they continue to work. Bus network is very simple and easy to set up.

If there is an urgent need to set up a network and perhaps be used on a short term basis then the Bus network is the best possibility. Bus networks use the least amount of cable to set up making it cost effective.

RING TOPOLOGY

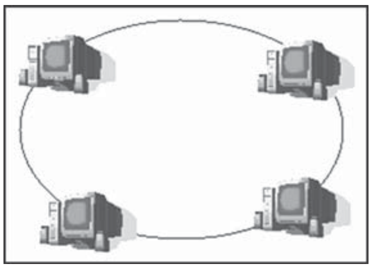
Definition

A ring topology is a network topology or circuit arrangement in which each network device is attached along the same signal path to two other devices, forming a path in the shape of a ring. Each device in the network that is also referred to as node handles every message that flows through the ring. Each node in the ring has a unique address. Since in a ring topology there is only one pathway between any two nodes, ring networks are generally disrupted by the failure of a single link.



The redundant topologies are used to eliminate network downtime caused by a single point of failure. All networks need redundancy for enhanced reliability. Network reliability is achieved through reliable equipment and network designs that are tolerant to failures and faults. The FDDI networks overcome the disruption in the network by sending data on a clockwise and a counterclockwise ring. In case there is a break in data flow,the data is wrapped back onto the complementary ring before it reaches the end of the cable thereby maintaining a path to every node within the complementary ring.

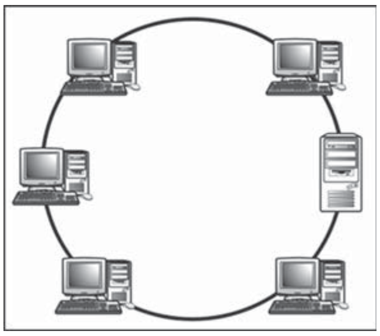
Ring network connects computers in a circle of point-to-point connections, with no central server, such as a series of desktop computers in an office. Each node handles its own applications and also shares resources over the entire network. If one node becomes inoperative, the other nodes are still able to maintain contact with one another. Such a network is best for decentralized systems in which no priorities are required



Under the network, a signal is transferred sequentially via a "token" fro one station to the next. When a station wants to transmit, it "grabs" the token, attaches data and an address to it, and then sends it around the ring.

The token travels along the ring until it reaches the destination address. The receiving computer acknowledges receipt with a return message to the sender. The sender then releases the token for the token for use by another computer.

Each station on the ring has equal access but only one station can talk at a time. To allow an orderly access to the ring, a single electronic token passes from one computer to the next around the ring as seen in (token ring). A computer can only transmit data when it capture the token.



In a true ring topology, if a single computer or cable fails, there is an interruption in the signal. The entire network becomes inaccessible. Network disruption can also occur when computers are added or removed from the network, making it an impractical network design in environments where there is constant change to the network.

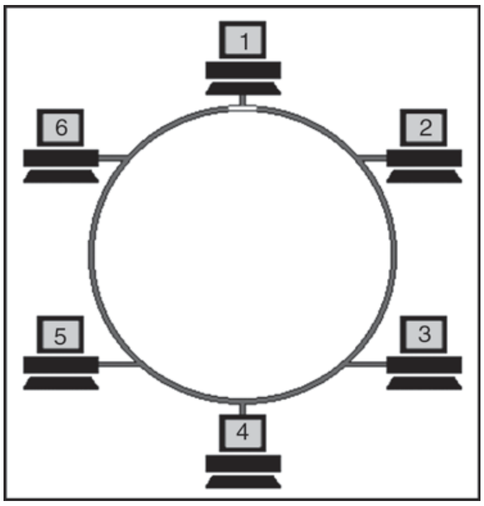
Ring networks are most commonly wired in a star configuration. In a Token Ring network, a multistation access unit (MSAU) is equivalent to a hub or switch on an Ethernet network.

The MSAU performs the token circulation internally. To create the complete ring, the ring in (RI) port on each MSAU is connected to the ring out (RO) port on another MSAU. The last MSAU in the ring is then connected to the first, to complete the ring.

TOKEN RING TOPOLOGY

Unlike Ethernet, Token Ring uses a ring topology whereby the data is sent from one machine to the next and so on around the ring until it ends up back where it started. It also uses a token passing protocol which means that a machine can only use the network when it has control of the Token, this ensures that there are no collisions because only one machine can use the network at any given time.

Figure below shows the basic of a Token Ring Topology. Although 16Mbps is the standard ring speed these days (and Fast Token Ring is being developed) we will consider a 4Mbps Token Ring in this tutorial to explain the basic concepts.



At the start, a free Token is circulating on the ring, this is a data frame which to all intents and purposes is an empty vessel for transporting data. To use the network, a machine first has to capture the free Token and replace the data with its own message.

In the example above, machine wants to send some data to machine, so it first has to capture the free Token. It then writes its data and the recipient's address onto the Token.

The packet of data is then sent to machine who reads the address, realises it is not its own, so passes it on to machine

Machine does the same and passes the Token on to machine. This time it is the correct address and so number reads the message. It cannot, however, release a free Token on to the ring, it must first send the message back to number with an acknowledgement to say that it has received the data (represented by the purple flashing screen).

The receipt is then sent to machine who checks the address, realises that it is not its own and so forwards it on to the next machine in the ring, number.Machine does the same and forwards the data to number, who sent the original message. Machin erecognizes the address, reads the acknowledgement from number (represented by the purple flashing screen) and then releases the free Token back on to the ring ready for the next machine to use.

That's the basics of Token Ring and it shows how data is sent, received and acknowledged, but Token Ring also has a built in management and recovery system which makes it very fault tolerant. Below is a brief outline of Token Ring's self maintenance system.

Token Ring Self Maintenance

When a Token Ring network starts up, the machines all take part in a negotiation to decide who will control the ring, or become the 'Active Monitor' to give it its proper title. This is won by the machine with the highest MAC address who is participating in the contention procedure, and all other machines become 'Standby Monitors'. The job of the Active

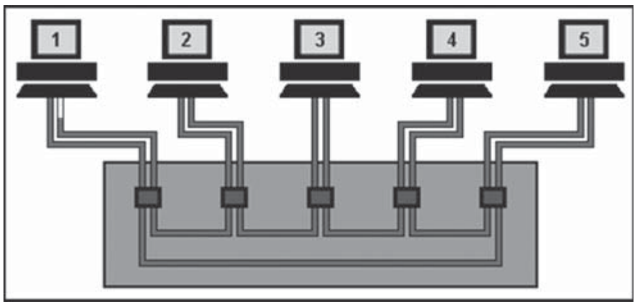
Monitor is to make sure that none of the machines are causing problems on the network, and to re-establish the ring after a break or an error has occurred. The Active Monitor performs Ring Polling every seven seconds and ring purges when there appears to be a problem. The ring polling allows all machines on the network to find out who is participating in the ring and to learn the address of their Nearest Active Upstream Neighbour (NAUN). Ring purges reset the ring after an interruption or loss of data is reported.

Each machine knows the address of its Nearest Active Upstream Neighbour. This is an important function in a Token Ring as it updates the information required to re-establish itself when machines enter or leave the ring. When a machine enters the ring it performs a lobe test to verify that its own connection is working properly, if it passes, it sends a voltage to the hub which operates a relay to insert it into the ring.

If a problem occurs anywhere on the ring, the machine that is immediately after the fault will cease to receive signals. If this situation continues for a short period of time it initiates a recovery procedure which assumes that its NAUN is at fault, the outcome of this procedure either removes its neighbour from the ring or it removes itself.

Token Ring Operation using a Hub

A Token Ring hub simply changes the topology from a physical ring to a star wired ring. The Token still circulates around the network and is still controlled in the same manner, however, using a hub or a switch greatly improves reliability because the hub can automatically bypass any ports that are disconnected or have a cabling fault.



A Token Ring hub simply changes the topology from a physical ring to a star wired ring.

The Token still circulates around the network and is still controlled in the same manner, however, using a hub or a switch greatly improves reliability because the hub can automatically bypass any ports that are disconnected or have a cabling fault

Further advancements have been made in recent years with regard to Token Ring technology, such as early Token release and Token Ring switching but as this site is primarily concerned with cabling issues we will not go into any more detail here.

STAR TOPOLOGY

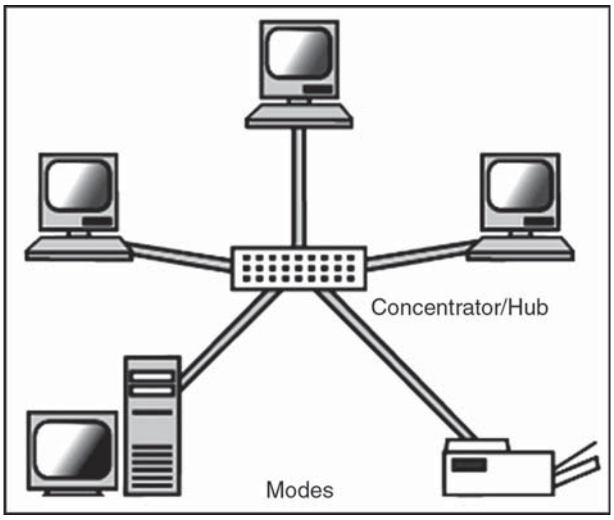
Star topology is the most commonly used physical technology in Ethernets LANS, when installed, the star topology resembles spokes I n a bicycle wheel. The star topology is made up of a central connection point that is a device such as a hub, switch, or router, where all the cabling segments meet. Each host in the network is connected to the central devices with its own cables although a physical star topology costs more to implement than the physical bus topology, the advantage of star topology make it worth the additional cost.

Because each host it connected to the central device with its own cable, when that cable has a problem, only that host is affected, the rest of the network remains operational this benefit is extremely important and is why virtually every newly designed Ethernet a central connection point might be desirable for security or restricted access, but this is also a main disadvantage of a star topology.if the central device fails, the whole network become disconnected.when a star network is expanded to include an additional networking device that is connected for the main networking device, it is called an extended star topology.

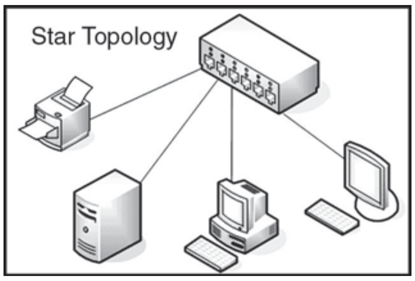
Description

A star topology is designed with each node (file server, workstations, and peripherals) connected directly to a central network hub or concentrator as shown in figure below:

Data on a star network passes through the hub or concentrator before continuing to its destination. The hub or concentrator manages and controls all functions of the network. It also acts as a repeater for the data flow. This configuration is common with twisted pair cable; however, it can also be used with coaxial cable or fibre optic cable.



Many businesses and home networks use the star topology. A star network features a central connection point called a "hub" that may be an actual hub or a switch as shown below:



Devices typically connect to the hub with Unshielded Twisted Pair (UTP) Ethernet cables also known as RJ45 cables. Compared to the bus topology, a star network generally requires more cable, but a failure in any star network cable will only take down one computer's network access and not the entire LAN. (If the hub fails, however, the entire network also fails.)

Star topology has the following advantages:

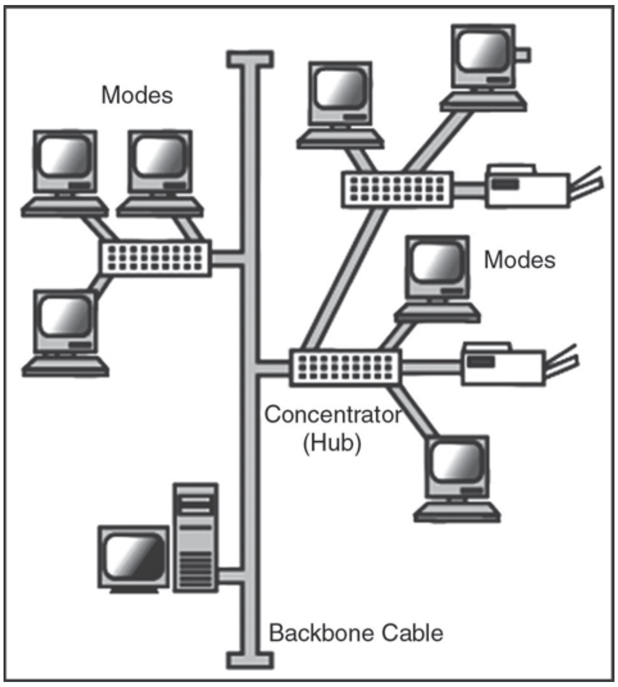
- Easy to manage
- Easy to locate problems (cable/workstations)
- Easier to expand than a bus or ring topology
- Easy to Install and Wire
- No disruptions to the network then connecting or removing devices.
- Easy to detect faults and to remove parts.

Star topology has following disadvantages:

- It is susceptible to a single point of failure be it the server or hub/switch
- With Increased devices and traffic can make the network slow
- Requires more Cable Length.
- If the hub or concentrator fails, nodes attached are disabled.
- More expensive than linear bus topologies because of the cost of the concentrators.

TREE TOPOLOGY

Among all the Network Topologies we can derive that the Tree Topology is a combination of the bus and the Star Topology. The tree like structure allows you to have many servers on the network and you can branch out the network in many ways. This is particularly helpful for colleges, universities and schools so that each of the branches can identify the relevant systems in their own network and yet connect to the big network in some way.



A Tree Structure suits best when the network is widely spread and vastly divided into many branches. Like any other topologies, the Tree Topology has its advantages and disadvantages.

A Tree Network may not suit small networks and it may be a waste of cable to use it for small networks. Tree Topology has some limitations and the configuration should suit those limitations.

Benefits

A Tree Topology is supported by many network vendors ad even hardware vendors. A point to point connection is possible with Tree Networks. All the computers have access to the larger and their immediate networks. Best topology for branched out networks.

Limitations

In a Network Topology the length of the network depends on the type of cable that is being used. The Tree Topology network is entirely dependant on the trunk which is the main backbone of the network. If that has to fail then the entire network would fail. Since the Tree Topology network is big it is difficult to configure and can get complicated after a certain point.

The Tree Topology follows a hierarchical pattern where each level is connected to the next higher level in a symmetrical pattern. Each level in the hierarchy follows a certain pattern in connecting the nodes. Like the top most level might have only one node or two nodes and the following level in the hierarchy might have few more nodes which work on the point to point connectivity and the third level also has asymmetrical node to node pattern and each of these levels are connected to the root level in the hierarchy. Think of a tree that branches out in various directions and all these branches need the roots and the tree trunk to survive. A Tree Structured network is very similar to this and that is why it is called the Tree Topology.

Features

There will be at least three levels of hierarchy in the Tree Network Topology and they all work based on the root node. The Tree Topology has two kinds of topology integral in it, the star and the linear way of connecting to nodes. The Tree

Topology functions by taking into account the total number of nodes present in the network. It does not matter how many nodes are there on each level.

Nodes can be added to any level of the hierarchy and there are no limitations a far as the total number of nodes do not exceed. The higher levels in the hierarchy are expected to perform more functions than the lower levels in the network.

Advantages of a Tree Topology

- Point-to-point wiring for individual segments.
- Supported by several hardware and software venders.

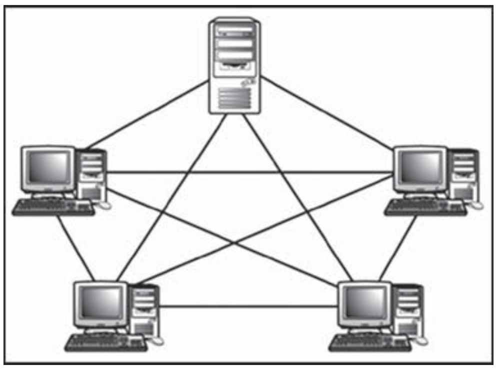
Disadvantages of a Tree Topology

- Overall length of each segment is limited by the type of cabling used.
- If the backbone line breaks, the entire segment goes down.
- More difficult to configure and wire than other topologies.

MESH TOPOLOGY

A type of network setup where each of the computers and network devices are interconnected with one another, allowing for most transmissions to be distributed, even if one of the connections go down. This type of topology is not commonly used for most computer networks as it is difficult and expensive to have redundant connection to every computer. However, this type of topology is commonly used for wireless networks.

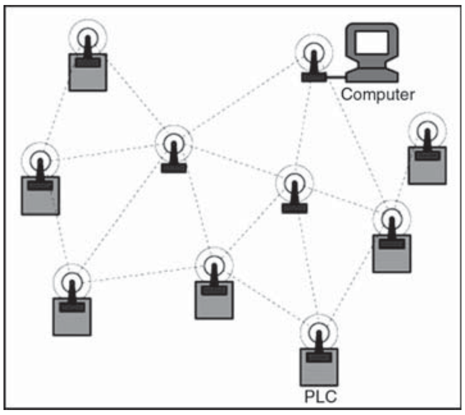
The mesh topology incorporates a unique network design in which each computer on the network connects to every other, creating a point-to-point connection between every device on the network. The purpose of the mesh design is to provide a high level of redundancy. If one network cable fails, the data always has an alternative path to get to its destination.



As seen in the above figure, the wiring for a mesh network can be very complicated. Further, the cabling costs associated with the mesh topology can be high, and troubleshooting a failed cable can be tricky. Because of this, the mesh topology is rarely used. A variation on a true mesh topology is the hybrid mesh. It creates a redundant point-to-point network connection between only specific network devices.

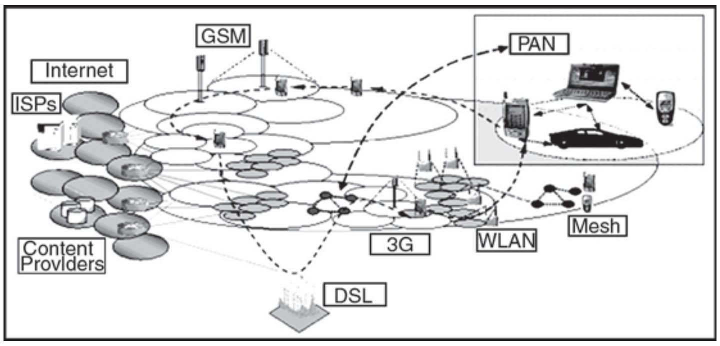
Wireless Mesh Networks

Wireless Mesh Networks work based on the radio frequencies and was originally developed by the army to be able to communicate. The reliability factor is high in any kind of Mesh Network. There are three types of wireless Mesh Topologies.



- Fixed Wireless Connections
- Peer to Peer or Adhoc Networks
- Node to Node

Fixed Mesh Networks



The fixed Mesh Networks will work only in the specified location and they are not mobile networks. They are meant to be used in a limited surrounding with boundaries. The location of nodes in affixed Mesh Network is all pre determined and they are not interchangeable.

The fixed Mesh Network does not work on line of sight like the other types of Mesh Networks. The total number of hops in a fixed Mesh Network is usually fixed and also short. There may not be many nodes as this kind of Mesh Networks exist within an office or building. More often than not the data travels ion a specific direction.

Peer to Peer Mobile Networks

In a peer to peer mobile network the individual devices connect to each other using the Mesh Network. The peer does not require connecting to the main node and they can still communicate from one device to another device by taking the shortest possible data transfer route. However many experts believe that in the peer to peer Mesh Networks the problems with scalability in terms of time taken for data transfer is questionable. The device has to know to transmit the data in the most optimal path and the entire data transfer or depends on this single factor. If the device is incapable then the whole purpose of using it in a peer to peer connection is lost.

E-MAIL

DEFINITION

E-mail is simply the shortened form of electronic mail, a protocol for receiving, sending and storing electronic messages. E-mail has gained popularity with the spread of the Internet. In many cases, e-mail has become the preferred method of communication. Though there is some degree of uncertainty as to when e-mail was invented, the father of the modern version is generally regarded to be American Ray Tomlinson. Before Tomlinson, messages could be sent between users but only those connected to the same computer. They could not be targeted to a particular individual. Tomlinson devised a way to address e-mail to certain users and thus was credited for one of the most important communication inventions in the 20th century.

Tomlinson's idea was to identify the name of the user and the computer at which they were located. As a result, the basic formula for addressing an e-mail was username@usercomputer. This standard has not changed over the years, other than the user computer now commonly being replaced by the name of an e-mail provider. In many cases, this is the same as the user's Internet service provider.

The influence of e-mail cannot be overstated. The United States Postal Service, for example, notes that it handles 212 billion pieces of mail per year. Many sources have nearly that many e-mails being sent back and forth every single day. In other words, e-mail handles more than 300 times the amount of mail of the largest postal system in the world.

In many cases, users receive, send and store e-mail based on one of two standards, SMTP or POP. SMTP stands for simple message transfer protocol. POP stands for Post Office Protocol. Though there is some amount of confusion over what these two protocols mean, the explanation is rather simple. POP is a protocol for storage of e-mail. SMTP is a protocol for sending and receiving.

WORKING WITH E MAIL

In reality, sending your message off into the network cloud is a bit like sending Little Red Riding Hood into the deep dark woods. You never know what might happen. In this diagram, the sender is a human being using their company account to send an e-mail to someone at a different company.

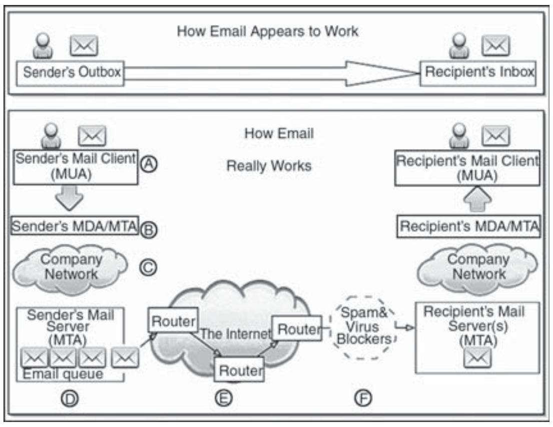
Step A: Sender creates and sends an e-mail

The originating sender creates an e-mail in their Mail User Agent (MUA) and clicks 'Send'. The MUA is the application the originating sender uses to compose and read e-mail, such as Eudora, Outlook, etc.

Step B: Sender's MDA/MTA routes the e-mail

The sender's MUA transfers the e-mail to a Mail Delivery Agent (MDA). Frequently, the sender's MTA also handles the responsibilities of an MDA. Several of the most common MTAs do this, including sendmail and qmail (which Kavi uses).

The MDA/MTA accepts the e-mail, then routes it to local mailboxes or forwards it if it isn't locally addressed. In our diagram, an MDA forwards the e-mail to an MTA and it enters the first of a series of "network clouds," labeled as a "Company Network" cloud.



Step C: Network Cloud

An e-mail can encounter a network cloud within a large company or ISP, or the largest network cloud in existence: the Internet. The network cloud may encompass a multitude of mail servers, DNS servers, routers, lions, tigers, bears (wolves!) and other devices and services too numerous to mention.

These are prone to be slow when processing an unusually heavy load, temporarily unable to receive an e-mail when taken down for maintenance, and sometimes may not have identified themselves properly to the Internet through the Domain Name System (DNS) so that other MTAs in the network cloud are unable to deliver mail as addressed. These devices may be protected by firewalls, spam filters and malware detection software that may bounce or even delete an e-mail.

When an e-mail is deleted by this kind of software, it tends to fail silently, so the sender is given no information about where or when the delivery failure occurred.

E-mail service providers and other companies that process a large volume of e-mail often have their own, private network clouds. These organizations commonly have multiple mail servers, and route all e-mail through a central gateway server (*i.e.*, mail hub) that redistributes mail to whichever MTA is available.

E-mail on these secondary MTAs must usually wait for the primary MTA (*i.e.*, the designated host for that domain) to become available, at which time the secondary mail server will transfer its messages to the primary MTA.

Step D: E-mail Queue

The e-mail in the diagram is addressed to someone at another company, so it enters an e-mail queue with other outgoing e-mail messages. If there is a high volume of mail in the queue-either because there are many messages or the messages are unusually large, or both-the message will be delayed in the queue until the MTA processes the messages ahead of it.

Step E: MTA to MTA Transfer

When transferring an e-mail, the sending MTA handles all aspects of mail delivery until the message has been either accepted or rejected by the receiving MTA.

As the e-mail clears the queue, it enters the Internet network cloud, where it is routed along a host-to-host chain of servers. Each MTA in the Internet network cloud needs to "stop and ask directions" from the Domain Name System (DNS) in order to identify the next MTA in the delivery chain. The exact route depends partly on server availability and mostly on which MTA can be found to accept e-mail for the domain specified in the address.

Most e-mail takes a path that is dependent on server availability, so a pair of messages originating from the same host and addressed to the same receiving host could take different paths. These days, it's mostly spammers that specify any part of the path, deliberately routing their message through a series of relay servers in an attempt to obscure the true origin of the message.

To find the recipient's IP address and mailbox, the MTA must drill down through the Domain Name System (DNS), which consists of a set of servers distributed across the Internet.

Beginning with the root nameservers at the top-level domain (.tld), then domain nameservers that handle requests for domains within that.tld, and eventually to nameservers that know about the local domain.

DNS resolution and transfer process

- There are 13 root servers serving the top-level domains (*e.g.*.org.com.edu.gov.net, etc.). These root servers refer requests for a given domain to the root name servers that handle requests for that tld. In practice, this step is seldom necessary.

- The MTA can bypass this step because it has already knows which domain name servers handle requests for these.tlds. It asks the appropriate DNS server which Mail Exchange (MX) servers have knowledge of the subdomain or local host in the e-mail address. The DNS server responds with an MX record: a prioritized list of MX servers for this domain.

An MX server is really an MTA wearing a different hat, just like a person who holds two jobs with different job titles (or three, if the MTA also handles the responsibilities of an MDA). To the DNS server, the server that accepts messages is an MX server. When is transferring messages, it is called an MTA.

- The MTA contacts the MX servers on the MX record in order of priority until it finds the designated host for that address domain.
- The sending MTA asks if the host accepts messages for the recipient's username

Step F: Firewalls, Spam and Virus Filters

The transfer process described in the last step is somewhat simplified.

An e-mail may be transferred to more than one MTA within a network cloud and is likely to be passed to at least one firewall before it reaches it's destination.

An e-mail encountering a firewall may be tested by spam and virus filters before it is allowed to pass inside the firewall.

These filters test to see if the message qualifies as spam or malware. If the message contains malware, the file is usually quarantined and the sender is notified. If the message is identified as spam, it will probably be deleted without notifying the sender.

Spam is difficult to detect because it can assume so many different forms, so spam filters test on a broad set of criteria and tend to misclassify a significant number of messages as spam, particularly messages from mailing lists. When an email from a list or other automated source seems to have vanished somewhere in the network cloud, the culprit is usually a spam filter at the receiver's ISP or company. This explained in greater detail in Virus Scanning and Spam Blocking.

Delivery

In the diagram, the e-mail makes it past the hazards of the spam trap...er...filter, and is accepted for delivery by the receiver's MTA. The MTA calls a local MDA to deliver the mail to the correct mailbox, where it will sit until it is retrieved by the recipient's MUA.

RFCs

Documents that define e-mail standards are called "Request For Comments (RFCs)", and are available on the Internet through the Internet Engineering Task Force (IETF) website. There are many RFCs and they form a somewhat complex, interlocking set of standards, but they are a font of information for anyone interested in gaining a deeper understanding of e-mail.

3

Taxonomy of Wireless Micro-Sensor Network Models

Advances in hardware and wireless network technologies have placed us at the doorstep of a new era where small wireless devices will provide access to information anytime, anywhere as well as actively participate in creating smart environments. One of the applications of smart spaces is sensor networks, networks that are formed when a set of small untethered sensor devices that are deployed in an *ad hoc* fashion cooperate on sensing a physical phenomenon. Sensor networks hold the promise of revolutionizing sensing in a wide range of application domains because of their reliability, accuracy, flexibility, cost-effectiveness, and ease of deployment. To motivate the challenges in designing sensor networks, consider the following scenarios: sensors are rapidly deployed in a remote in hospitable area for a surveillance application; sensors are used to analyse the motion of a tornado; sensors are deployed in a forest for fire detection; sensors are attached to taxi cabs in a large metro politan area to study the traffic conditions and plan routes effectively; and smart Kindergarten where sensor networks are deployed to create a developmental problem-solving environment for early childhood education.

Clearly, there is a wide range of applications for sensor networks with differing requirements. We believe that a better understanding of micro-sensor network requirements as well as the underlying differences between micro-sensor applications is needed to assist designers. To this end, in this paper we attempt to classify wireless micro-sensor networks. In particular, we classify the aspects of wireless micro-sensor networks that we believe are most relevant to communication. We examine the characteristics and goals of typical micro-sensor networks as well as the different types of communication that are required to achieve these goals. We compare different data delivery models and network dynamics to create a taxonomy of wireless micro-sensor network communication. We believe that this taxonomy will aid network designers in making better decisions regarding the organization of the network, the network protocol and information dissemination models. Furthermore, it will aid in developing realistic sensor network models and benchmarks for use in future sensor network research.

SENSOR NETWORK ARCHITECTURE

A sensor network is a tool for measuring and relaying information about the phenomenon to the observer within the desired performance bound and deployment cost. As such, the organization of the network may be viewed as follows:

Infrastructure: The infrastructure consists of the sensors and their current deployment status. More specifically, the infrastructure is influenced by the characteristics of the sensors (*e.g.*, sensing accuracy, memory size, battery life, transmission range) and deployment strategy (*e.g.*, sensor density, sensor location, sensor mobility).

Network Protocol: The network protocol is responsible for creating paths and accomplishing communication between the sensors and the observer(s).

Application/Observer: The observer(s) interests in the phenomenon are queries from the observer(s) about the phenomenon as approximated by the distributed data that the sensors are capable of sensing. These queries could be static (the sensors are preprogrammed to report data according to a specific pattern) or dynamic.

The network may participate in synthesizing the query; we consider such intelligence to be part of the translation process between observer interests and low-level implementation.

In this work, we focus on classifying issues that influence the second level: the network protocol. The other two levels only with regard to issues that influence communication. Thus, we do not address the difficult problem of translation between the observer query and the specific low-level interests. This translation could be done by the application software at the observer and/or the sensor nodes, or directly by a human observer. Similarly, we do not discuss the engineering of the infrastructure.

We also note that there is a significant opportunity for optimizations that cut across the three organizational levels. For example, Bhatnagar *et al.* discuss supporting QoS for sensor networks. More specifically, they discuss discriminating among the type of data that the sensors are reporting and preferentially treating high priority data (for example, by giving it priority in forwarding and using redundancy to increase the chance of its reception). This is an example of an optimization where application-level knowledge provides hints to the network protocol. As another example, consider the case where the deployment of the sensors is chosen to mirror the expected motion pattern of the phenomenon or the interests of the observer. Such a deployment strategy incorporates application knowledge in the infrastructure design.

The network protocol in a sensor network is responsible for supporting all communication, both among sensor nodes as well as between the sensor nodes and the observer(s). The performance of the protocol will be highly influenced by the network dynamics, as well as by the specific data delivery model employed. In order to determine how the network protocol behaves for different scenarios, it is important to classify these features. We classify the different types of communication required in a sensor network and then look at the possible data delivery models and network dynamics.

COMMUNICATION MODELS

There are multiple ways for a sensor network to achieve its accuracy and delay requirements; a well designed network meets these requirements while optimizing the sensor energy usage and providing fault tolerance. By studying the communication patterns systematically, the network designer will be able to choose the infrastructure and communication protocol that provide the best combination of performance, robustness, efficiency and deployment cost. Conceptually, communication within a sensor network can be classified into two categories: application and infrastructure. The network protocol must support both these types of communication. Application communication relates to the transfer of sensed data (or information obtained from it) with the goal of informing the observer about the phenomena. Within application communication, there are two models: cooperative and non-cooperative. Under the cooperative sensor model, sensors communicate with other sensors to realise the observer interest. This communication is beyond the relay function needed for routing. For example, in a clustering protocol a cluster-head and the sensor nodes communicate with each other for information dissemination related to the actual phenomenon. In-network data processing is an example of co-operative sensors. Non-cooperative sensors do not cooperate for information dissemination.

Infrastructure communication refers to the communication needed to configure, maintain and optimize operation. More specifically, because of the *ad hoc* nature of sensor networks, sensors must be able to discover paths to other sensors of interest to them and to the observer regardless of sensor mobility or failure. Thus, infrastructure communication is needed to keep the network functional, ensure robust operation in dynamic environments, as well as optimize overall performance. We note that such infrastructure communication is highly influenced by the application interests since the network must reconfigure itself to best satisfy these interests. As infrastructure communication represents the overhead of the protocol, it is important to minimize this communication while ensuring that the network can support efficient application communication. In sensor networks, an initial phase of infrastructure communication is needed to set up the network. Furthermore, if the sensors are energy-constrained, there will be additional communication for reconfiguration.

Similarly, if the sensors are mobile or the observer interests dynamic, additional communication is needed for path discovery/reconfiguration. For example, in a clustering protocol, infrastructure communication is required for the formation of clusters and cluster-head selection; under mobility or sensor failure, this communication must be repeated (periodically or upon detecting failure). Finally, infrastructure communication is used for network optimization. Consider the Frisbee model, where the set of active sensors follows a moving phenomenon to optimize energy efficiency. In this case, the sensors wake up other sensors in the network using infrastructure communication.

Sensor networks require both application and infrastructure communication. The amount of required communication is highly influenced by the networking protocol used. Application communication is optimized by reporting measurements at the minimal rate that will satisfy the accuracy and delay requirements given sensor abilities and the quality of the paths between the sensors and the observer. The infrastructure communication is generated by the networking protocol in response to application requests or events in the network. Investing in infrastructure communication can reduce application traffic and optimize overall network operation.

Data Delivery Models

Ideally, the observer interest is specified in terms of the phenomenon, allowing the observer to be oblivious to the underlying sensor network infrastructure and protocol. The query is implemented as one or more specific low-level interests (*e.g.*, requesting a specific sensor to report a specific measurement at some specific interval). Sensor networks can be classified in terms of the data delivery required by the application (observer) interest as: continuous, eventdriven, observer-initiated and hybrid. These models govern the generation of the application traffic. In the continuous model, the sensors communicate their data continuously at a prespecified rate. The authors in showed that clustering is most efficient for static networks where data is continuously transmitted. For dynamic sensor networks, depending upon the degree of mobility, clustering may be applicable as well. In the event-driven data model the sensors report information only if an event of interest occurs. In this case, the observer is interested only in the occurrence of a specific phenomenon or set of phenomena. In the observer-initiated (or request-reply) model, the sensors only report their results in response to an explicit request from the observer (either directly, or indirectly through other sensors). Finally, the three approaches can coexist in the same network; we refer to this model as the hybrid model.

Thus far, we have only discussed data delivery from the application perspective, and not the actual flow of data packets between the sensors and the observer; this is a routing problem subject to the network protocol. We can classify the routing approach as: flooding (broadcastbased), unicast, or multicast/other. Using a flooding approach, sensors broadcast their information to their neighbours, who rebroadcast this data until it reaches the observer. This approach incurs high overhead but is immune to dynamic changes in the topology of the network. Research has been conducted on techniques such as data aggregation that can be used to reduce the overhead of the broadcast. Alternatively, the sensors can either communicate to the observer directly (possibly using a multi-hop routing protocol) or communicate with a cluster-head using one-to-one unicast. Finally, in a multicast approach, sensors form application-directed groups and use multicast to communicate among group members. The observer could communicate with any member of the group to obtain the desired data. A major advantage of flooding or broadcast is the lack of a complex network layer protocol for routing, address and location management; existing sensor network efforts have mostly relied on this approach. However, the overhead of a broadcasting approach may be prohibitive. It is likely that the interaction between the data delivery model from the application and the routing model employed by the network protocol will significantly impact the performance of the network. Consider a scenario where a sensor network is deployed for intrusion detection. In this case, the data delivery model is event driven – the event being an intruder entering the area. If the network level routing model is flooding based, in such a case physically co-located sensors will in general sense the intruder at the same time and try to send data to the observer simultaneously. These concurrent communications in the neighbourhood will contend with each other for the use of the communication medium, raising: (1) the probability of loss of critical information; and (2) the latency in event reporting. A similar problem is studied by Woo and Culler.

NETWORK DYNAMICS MODELS

A sensor network forms a path between the phenomenon and the observer. The goal of the sensor network protocol is to create and maintain this path (or multiple paths) under dynamic conditions while meeting the application requirements of low energy, low latency, high accuracy, and fault tolerance. Without loss of generality, this discussion assumes a single observer. Multiple observers can be supported as multiple instances of a single observer. More sophisticated protocols could also capitalize on the presence of multiple observers to merge related interests and/or optimize communication.

The problem of setting up paths for information dissemination is similar to the problem of routing in *ad hoc* networks. However, there are a few critical differences, including: (i) the sensors are not generally addressed individually; rather, the interest is in the set of sensors that are in a position to contribute to the active observer interests. The sensors could be addressed by attributes of the sensor (*e.g.*, their capabilities) and/or the phenomenon (*e.g.*, the sensors close to a lion in a habitat monitoring scenario). The mapping between the observer interest and a set of sensors is influenced by the network dynamics and the application; and (ii) nodes along the path can take an active role in the information dissemination and processing. In this respect, sensor networks are similar to Active Networks whereas ad hoc networks are traditional "passive" networks. There are several approaches to construct and maintain a path between observer and phenomenon. These will differ depending on the network dynamics, which we classify as: static sensor networks and mobile sensor networks. We focus on mobility because it is the most common source of dynamic conditions; other sources include sensor failure and changes in observer interests.

Static Sensor Networks

In static sensor networks, there is no motion among communicating sensors, the observer and the phenomenon. An example is a group of sensors spread for temperature sensing. For these types of sensor networks, previous studies have shown that localized algorithms can be used in an effective way. The sensors in localized algorithms communicate with nodes in their locality. An elected node relays a summary of the local observations to the observer, perhaps through one on more levels of hierarchy. Such algorithms extend the lifetime of the sensor network because they trade-off local computation for communication. In this type of network, sensor nodes require an initial set-up infrastructure communication to create the path between the observer and the sensors with the remaining traffic exclusively application communication.

Dynamic Sensor Networks

In dynamic sensor networks, either the sensors themselves, the observer, or the phenomenon are mobile. Whenever any of the sensors associated with the current path from the observer to the phenomenon moves, the path may fail. In this case, either the observer or the concerned sensor must take the initiative to rebuild a new path. During initial set-up, the observer can build multiple paths between itself and the phenomenon and cache them, choosing the one that is the most beneficial at that time as the current path. If the path fails, another of the cached paths can be used. If all the cached paths are invalid then the observer must rebuild new paths. This observer-initiated approach is a reactive approach, where path recovery action is only taken after observing a broken path.

Another model for rebuilding new paths from the observer to the phenomenon is a sensor-initiated approach. In a sensor-initiated path recovery procedure, path recovery is initiated by a sensor that is currently part of the logical path between the observer and the phenomenon and is planning to move out of range. The sensor might perform some local patching procedure to build a new path by broadcasting a participation request for a given logical flow to all its neighbouring sensors. Any one of the neighbouring sensors can send a participation reply message to the given initiator sensor indicating willingness to participate and become a part of the requested path. If none of the neighbouring sensors respond, the sensor can default to sending a path invalidation request to the observer so that the observer can start building the path. This is similar to soft hand-off in traditional Mobile IP based networks. This sensor-initiated approach is a proactive approach where path recovery operations are begun in anticipation of a future broken path. Dynamic sensor networks can be further classified by considering the motion of the components. This motion is important from the communications perspective since the degree and type of communication is dependent on network dynamics. We believe that each of the following require different infrastructures, data delivery models, and protocols:

Mobile observer: In this case the observer is mobile with respect to the sensors and phenomena.

An example of this paradigm is sensors deployed in an inhospitable area for environment monitoring. For example, a plane might fly over a field periodically to collect information from a sensor network. Thus the observer, in the plane, is moving relative to the sensors and phenomena on the ground.

Mobile sensors: In this case, the sensors are moving with respect to each other and the observer. For example, consider traffic monitoring implemented by attaching sensors to taxis.

As the taxis move, the attached sensors continuously communicate with each other about their own observations of the traffic conditions. If the sensors are co-operative, the communication paradigm imposes additional constraints such as detecting the link layer addresses of the neighbours and constructing localization and information dissemination structures. From previous work, we know that the overhead of maintaining a globally unique sensor ID in a hierarchical fashion like an IP address is expensive and not needed.

Instead, these sensors should communicate only with their neighbours with the link layer MAC address. In such networks, the proactive algorithm with local patching for repairing a path can be used so that the information about the phenomenon is always available to the observer regardless of the mobility of the individual sensors.

Mobile phenomena: In this case, the phenomenon itself is moving. A typical example of this paradigm is sensors deployed for animal detection. In this case the infrastructure level communication should be event-driven. Depending on the density of the phenomena, it will be ineffficient if all the sensor nodes are active all the time. Only the sensors in the vicinity of the mobile phenomenon need to be active. The number of active sensors in the vicinity of the phenomenon can be determined by application specific goals such as accuracy, latency, and energy efficiency. A model that is well-suited to this case is the Frisbee model.

It is important to note that the effect of mobility in sensor networks is fundamentally different than that in traditional wireless networks. Mobility in *ad hoc* networks has been addressed from the point of view of mobility of one or more of the communicating nodes during communication. However, since the sensors themselves are of no interest to the observer, their mobility is not necessarily of interest; rather, the sensor network must adapt its operation to continue to reflect the observer interests in the presence of mobility. Thus, the mobility of the sensing nodes themselves should be handled in a different way than for *ad hoc* networks; for example, a node that is moving away from a phenomenon may choose to hand-off the responsibility of monitoring to a closer node as it drifts away.

CONSIDERATIONS SURROUNDING THE STUDY OF PROTECTION

General Observations: As computers become better understood and more economical, every day brings new applications. Many of these new applications involve both storing information and simultaneous use by several individuals. The key concern in this paper is multiple use. For those applications in which all users should not have identical authority, some scheme is needed to ensure that the computer system implements the desired authority structure.

For example, in an airline seat reservation system, a reservation agent might have authority to make reservations and to cancel reservations for people whose names he can supply. A flight boarding agent might have the additional authority to print out the list of all passengers who hold reservations on the flights for which he is responsible. The airline might wish to withhold from the reservation agent the authority to print out a list of reservations, so as to be sure that a request for a passenger list from a law enforcement agency is reviewed by the correct level of management.

The airline example is one of protection of corporate information for corporate self-protection (or public interest, depending on one's view). A different kind of example is an online warehouse inventory management system that generates reports about the current status of the inventory. These reports not only represent corporate information that must be protected from release outside the company, but also may indicate the quality of the job being done by the warehouse manager. In order to preserve his personal privacy, it may be appropriate to restrict the access to such reports, even within the company, to those who have a legitimate reason to be judging the quality of the warehouse manager's work.

Many other examples of systems requiring protection of information are encountered every day: credit bureau data banks; law enforcement information systems; time-sharing service bureaus; on-line medical information systems; and government social service data processing systems. These examples span a wide range of needs for organizational and personal privacy. All have in common controlled sharing of information among multiple users. All, therefore, require some plan to ensure that the computer system helps implement the correct authority structure.

Of course, in some applications no special provisions in the computer system are necessary. It may be, for instance, that an externally administered code of ethics or a lack of knowledge about computers adequately protects the stored information. Although there are situations in which the computer need provide no aids to ensure protection of information, often it is appropriate to have the computer enforce a desired authority structure.

The words “privacy,” “security,” and “protection” are frequently used in connection with information-storing systems. Not all authors use these terms in the same way. This chapter uses definitions commonly encountered in computer science literature.

The term “privacy” denotes a socially defined ability of an individual (or organization) to determine whether, when, and to whom personal (or organizational) information is to be released.

This chapter will not be explicitly concerned with privacy, but instead with the mechanisms used to help achieve it.

The term "security" describes techniques that control who may use or modify the computer or the information contained in it.

Security specialists have found it useful to place potential security violations in three categories.

- Unauthorized information release: an unauthorized person is able to read and take advantage of information stored in the computer. This category of concern sometimes extends to "traffic analysis," in which the intruder observes only the patterns of information use and from those patterns can infer some information content. It also includes unauthorized use of a proprietary programme.
- Unauthorized information modification: an unauthorized person is able to make changes in stored information—a form of sabotage. Note that this kind of violation does not require that the intruder see the information he has changed.
- Unauthorized denial of use: an intruder can prevent an authorized user from referring to or modifying information, even though the intruder may not be able to refer to or modify the information. Causing a system "crash," disrupting a scheduling algorithm, or firing a bullet into a computer are examples of denial of use. This is another form of sabotage.

The term "unauthorized" in the three categories listed above means that release, modification, or denial of use occurs contrary to the desire of the person who controls the information, possibly even contrary to the constraints supposedly enforced by the system. The biggest complication in a general-purpose remote-accessed computer system is that the "intruder" in these definitions may be an otherwise legitimate user of the computer system.

Examples of security techniques sometimes applied to computer systems are the following:

1. Labelling files with lists of authorized users,
2. Verifying the identity of a prospective user by demanding a password,
3. Shielding the computer to prevent interception and subsequent interpretation of electromagnetic radiation,
4. Enciphering information sent over telephone lines,
5. Locking the room containing the computer,
6. Controlling who is allowed to make changes to the computer system (both its hardware and software),
7. Using redundant circuits or programmed cross-checks that maintain security in the face of hardware or software failures,
8. Certifying that the hardware and software are actually implemented as intended.

It is apparent that a wide range of considerations are pertinent to the engineering of security of information. Historically, the literature of computer systems has more narrowly defined the term *protection* to be just those security techniques that control the access of executing programmes to stored information. An example of a protection technique is labelling of computer-stored files with lists of authorized users. Similarly, the term *authentication* is used for those security techniques that verify the identity of a person (or other external agent) making a request of a computer system. An example of an authentication technique is demanding a password. This chapter concentrates on protection and authentication mechanisms, with only occasional reference to the other equally necessary security mechanisms. One should recognize that concentration on protection and authentication mechanisms provides a narrow view of information security, and that a narrow view is dangerous. The objective of a secure system is to prevent all unauthorized use of information, a negative kind of requirement. It is hard to prove that this negative requirement has been achieved, for one must demonstrate that every possible threat has been anticipated. Thus an expansive view of the problem is most appropriate to help ensure that no gaps appear in the strategy. In contrast, a narrow concentration on protection mechanisms, especially those logically impossible to defeat, may lead to false confidence in the system as a whole.

Functional Levels of Information Protection

Many different designs have been proposed and mechanisms implemented for protecting information in computer systems. One reason for differences among protection schemes is their different functional properties—the kinds of access control that can be expressed naturally and enforced. It is convenient to divide protection schemes according to their functional properties. A rough categorization is the following.

Unprotected Systems

Some systems have no provision for preventing a determined user from having access to every piece of information stored in the system. Although these systems are not directly of interest here, they are worth mentioning since, as of 1975, many of the most widely used, commercially available batch data processing systems fall into this category—for example, the Disk Operating System for the IBM System 370. Our definition of protection, which excludes features usable only for mistake prevention, is important here since it is common for unprotected systems to contain a variety of mistake-prevention features. These may provide just enough control that any breach of control is likely to be the result of a deliberate act rather than an accident. Nevertheless, it would be a mistake to claim that such systems provide any security.

All-or-nothing Systems

These are systems that provide isolation of users, sometimes moderated by total sharing of some pieces of information. If only isolation is provided, the user of such a system might just as well be using his own private computer, as far as protection and sharing of information are concerned. More commonly, such systems also have public libraries to which every user may have access. In some cases the public library mechanism may be extended to accept user contributions, but still on the basis that all users have equal access.

Most of the first generation of commercial timesharing systems provide a protection scheme with this level of function. Examples include the Dartmouth Time-Sharing System (DTSS) and IBM's VM/370 system. There are innumerable others.

Controlled Sharing

Significantly more complex machinery is required to control explicitly who may access each data item stored in the system. For example, such a system might provide each file with a list of authorized users and allow an owner to distinguish several common patterns of use, such as reading, writing, or executing the contents of the file as a programme. Although conceptually straightforward, actual implementation is surprisingly intricate, and only a few complete examples exist. These include M.l.T.'s Compatible Time-Sharing System (CTSS), Digital Equipment Corporation's DECsystem/10, System Development Corporation's Advanced Development Prototype (ADEPT) System, and Bolt, Beranek, and Newman's TENEX

User-programmed Sharing Controls

A user may want to restrict access to a file in a way not provided in the standard facilities for controlling sharing. For example, he may wish to permit access only on weekdays between 9:00 A.M. and 4:00 P.M. Possibly, he may wish to permit access to only the average value of the data in a file. Maybe he wishes to require that a file be modified only if two users agree. For such cases, and a myriad of others, a general escape is to provide for user-defined *protected objects* and *subsystems*. A *protected subsystem* is a collection of programmes and data with the property that only the programmes of the subsystem have direct access to the data (that is, the protected objects). Access to those programmes is limited to calling specified entry points. Thus the programmes of the subsystem completely control the operations performed on the data. By constructing a protected subsystem, a user can develop any programmable form of access control to the objects he creates. Only a few of the most advanced system designs have tried to permit user-specified protected subsystems. These include Honeywell's Multics, the University of California's CAL system, Bell Laboratories' UNIX system, the Berkeley Computer Corporation BCC-500, and two systems currently under construction: the CAP system of Cambridge University, and the HYDRA system of Carnegie-Mellon University. Exploring alternative mechanisms for implementing protected subsystems is a current research topic. A specialized use of protected subsystems is the implementation of protection controls based on data content. For example, in a file of salaries, one may wish to permit access to all salaries under $15 000. Another example is permitting access to certain statistical aggregations of data but not to any individual data item. This area of protection raises questions about the possibility of discerning information by statistical tests and by examining indexes, without ever having direct access to the data itself. Protection based on content is the subject of a variety of recent or current research projects and will not be explored in this tutorial.

Putting Strings on Information

The foregoing three levels have been concerned with establishing conditions for the release of information to an executing programme. The fourth level of capability is to maintain some control over the user of the information even *after* it has been released. Such control is desired, for example, in releasing income information to a tax advisor; constraints should prevent him from passing the information on to a firm which prepares mailing lists. The printed labels on classified military information declaring a document to be “Top Secret” are another example of a constraint on information after its release to a person authorized to receive it. One may not (without risking severe penalties) release such information to others, and the label serves as a notice of the restriction. Computer systems that implement such strings on information are rare and the mechanisms are incomplete. For example, the ADEPT system keeps track of the classification level of all input data used to create a file; all output data are automatically labelled with the highest classification encountered during execution.

There is a consideration that cuts across all levels of functional capability: the *dynamics of use*. This term refers to how one establishes and changes the specification of who may access what. At any of the levels it is relatively easy to envision (and design) systems that statically express a particular protection intent. But the need to change access authorization dynamically and the need for such changes to be requested by executing programmes introduces much complexity into protection systems. For a given functional level, most existing protection systems differ primarily in the way they handle protection dynamics. To gain some insight into the complexity introduced by programme-directed changes to access authorization, consider the question "Is there any way that O'Hara could access file X?" One should check to see not only if O'Hara has access to file X, but also whether or not O'Hara may change the specification of file X's accessibility. The next step is to see if O'Hara can change the specification of who may change the specification of file X's accessibility, etc. Another problem of dynamics arises when the owner revokes a user's access to a file while that file is being used. Letting the previously authorized user continue until he is “finished” with the information may not be acceptable, if the owner has suddenly realised that the file contains sensitive data. On the other hand, immediate withdrawal of authorization may severely disrupt the user. It should be apparent that provisions for the dynamics of use are at least as important as those for static specification of protection intent.

In many cases, it is not necessary to meet the protection needs of the person responsible for the information stored in the computer entirely through computer-aided enforcement. External mechanisms such as contracts, ignorance, or barbed wire fences may provide some of the required functional capability. However, is focused on the internal mechanisms.

Design Principles

Whatever the level of functionality provided, the usefulness of a set of protection mechanisms depends upon the ability of a system to prevent security violations. In practice, producing a system at any level of functionality (except level one) that actually does prevent all such unauthorized acts has proved to be extremely difficult. Sophisticated users of most systems are aware of at least one way to crash the system, denying other users authorized access to stored information. Penetration exercises involving a large number of different general-purpose systems all have shown that users can construct programmes that can obtain unauthorized access to information stored within. Even in systems designed and implemented with security as an important objective, design and implementation flaws provide paths that circumvent the intended access constraints. Design and construction techniques that systematically exclude flaws are the topic of much research activity, but no complete method applicable to the construction of large general-purpose systems exists yet. This difficulty is related to the negative quality of the requirement to prevent *all* unauthorized actions.

In the absence of such methodical techniques, experience has provided some useful principles that can guide the design and contribute to an implementation without security flaws. Here are eight examples of design principles that apply particularly to protection mechanisms.

Economy of Mechanism

Keep the design as simple and small as possible. This well-known principle applies to any aspect of a system, but it deserves emphasis for protection mechanisms for this reason: design and implementation errors that result in unwanted access paths will not be noticed during normal use (since normal use usually does not include attempts to exercise improper access paths). As a result, techniques such as line-by-line inspection of software and physical examination of hardware that implements protection mechanisms are necessary. For such techniques to be successful, a small and simple design is essential.

Fail-safe Defaults

Base access decisions on permission rather than exclusion. This principle, suggested by E. Glaser in 1965, means that the default situation is lack of access, and the protection scheme identifies conditions under which access is permitted. The alternative, in which mechanisms attempt to identify conditions under which access should be refused, presents the wrong psychological base for secure system design. A conservative design must be based on arguments why objects should be accessible, rather than why they should not. In a large system some objects will be inadequately considered, so a default of lack of permission is safer. A design or implementation mistake in a mechanism that gives explicit permission tends to fail by refusing permission, a safe situation, since it will be quickly detected. On the other hand, a design or implementation mistake in a mechanism that explicitly excludes access tends to fail by allowing access, a failure which may go unnoticed in normal use. This principle applies both to the outward appearance of the protection mechanism and to its underlying implementation.

Complete Mediation

Every access to every object must be checked for authority. This principle, when systematically applied, is the primary underpinning of the protection system. It forces a system-wide view of access control, which in addition to normal operation includes initialization, recovery, shutdown, and maintenance.

It implies that a foolproof method of identifying the source of every request must be devised. It also requires that proposals to gain performance by remembering the result of an authority check be examined skeptically. If a change in authority occurs, such remembered results must be systematically updated.

Open Design

The design should not be secret. The mechanisms should not depend on the ignorance of potential attackers, but rather on the possession of specific, more easily protected, keys or passwords. This decoupling of protection mechanisms from protection keys permits the mechanisms to be examined by many reviewers without concern that the review may itself compromise the safeguards. In addition, any skeptical user may be allowed to convince himself that the system he is about to use is adequate for his purpose. Finally, it is simply not realistic to attempt to maintain secrecy for any system which receives wide distribution.

Separation of Privilege

Where feasible, a protection mechanism that requires two keys to unlock it is more robust and flexible than one that allows access to the presenter of only a single key. The relevance of this observation to computer systems was pointed out by R. Needham in 1973. The reason is that, once the mechanism is locked, the two keys can be physically separated and distinct programmes, organizations, or individuals made responsible for them. From then on, no single accident, deception, or breach of trust is sufficient to compromise the protected information. This principle is often used in bank safe-deposit boxes. It is also at work in the defence system that fires a nuclear weapon only if two different people both give the correct command. In a computer system, separated keys apply to any situation in which two or more conditions must be met before access should be permitted. For example, systems providing user-extendible protected data types usually depend on separation of privilege for their implementation.

Least Privilege

Every programme and every user of the system should operate using the least set of privileges necessary to complete the job. Primarily, this principle limits the damage that can result from an accident or error. It also reduces the number of potential interactions among privileged programmes to the minimum for correct operation, so that unintentional, unwanted, or improper uses of privilege are less likely to occur. Thus, if a question arises related to misuse of a privilege, the number of programmes that must be audited is minimized. Put another way, if a mechanism can provide "firewalls," the principle of least privilege provides a rationale for where to install the firewalls. The military security rule of "need-to-know" is an example of this principle.

Least Common Mechanism

Minimize the amount of mechanism common to more than one user and depended on by all users. Every shared mechanism (especially one involving shared variables) represents a potential information path between users and must be designed with great care to be sure it does not unintentionally compromise security. Further, any mechanism serving all users must be certified to the satisfaction of every user, a job presumably harder than satisfying only one or a few users.

For example, given the choice of implementing a new function as a supervisor procedure shared by all users or as a library procedure that can be handled as though it were the user's own, choose the latter course. Then, if one or a few users are not satisfied with the level of certification of the function, they can provide a substitute or not use it at all. Either way, they can avoid being harmed by a mistake in it.

Psychological Acceptability

It is essential that the human interface be designed for ease of use, so that users routinely and automatically apply the protection mechanisms correctly. Also, to the extent that the user's mental image of his protection goals matches the mechanisms he must use, mistakes will be minimized. If he must translate his image of his protection needs into a radically different specification language, he will make errors.

4

Collection of Network Elements

A CDN is a collection of network elements arranged for more effective delivery of content to end-users. Collaboration among distributed CDN components can occur over nodes in both homogeneous and heterogeneous environments. CDNs can take various forms and structures.

They can be centralized, hierarchical infrastructure under certain administrative control, or completely decentralized systems. There can also be various forms of internetworking and control sharing among different CDN entities. General considerations on designing a CDN can be found.

The typical functionality of a CDN includes:

- Request redirection and content delivery services to direct a request to the closest suitable surrogate server using mechanisms to bypass congestion, thus overcoming flash crowds or SlashDot effects.

- Content outsourcing and distribution services to replicate and/or cache content to distributed surrogate servers on behalf of the origin server.
- Content negotiation services to meet specific needs of each individual user (or group of users).
- Management services to manage the network components, to handle accounting, and to monitor and report on content usage.

A CDN provides better performance through caching or replicating content over some mirrored Web servers (*i.e.* surrogate servers) strategically placed at various locations in order to deal with the sudden spike in Web content requests, which is often termed as flash crowd or SlashDot effect. The users are redirected to the surrogate server nearest to them.

This approach helps to reduce network impact on the response time of user requests. In the context of CDNs, content refers to any digital data resources and it consists of two main parts: the encoded media and metadata. The encoded media includes static, dynamic and continuous media data (*e.g.* audio, video, documents, images and Web pages). Metadata is the content description that allows identification, discovery, and management of multimedia data, and also facilitates the interpretation of multimedia data. Content can be pre-recorded or retrieved from live sources; it can be persistent or transient data within the system.

CDNs can be seen as a new virtual overlay to the Open Systems Interconnection (OSI) basic reference model. This layer provides overlay network services relying on application layer protocols such as HTTP or RTSP for transport. The three key components of a CDN architecture are – content provider, CDN provider and end-users. A content provider or customer is one who delegates the URI name space of the Web objects to be distributed. The origin server of the content provider holds those objects. A CDN provider is a proprietary organization or company that provides infrastructure facilities to content providers in order to deliver content in a timely and reliable manner. End-users or clients are the entities who access content from the content provider's website.

CDN providers use caching and/or replica servers located in different geographical locations to replicate content. CDN cache servers are also called edge servers or surrogates. In this chapter, we will use these terms interchangeably. The surrogates of a CDN are called Web cluster as a whole. CDNs distribute content to the surrogates in such a way that all cache servers share the same content and URL. Client requests are redirected to the nearby surrogate, and a selected surrogate server delivers requested content to the end-users. Thus, transparency for users is achieved.

Additionally, surrogates send accounting information for the delivered content to the accounting system of the CDN provider.

The Evolution of CDNs

Over the last decades, users have witnessed the growth and maturity of the Internet. As a consequence, there has been an enormous growth in network traffic, driven by rapid acceptance of broadband access, along with increases in system complexity and content richness. The over-evolving nature of the Internet brings new challenges in managing and delivering content to users. As an example, popular Web services often suffer congestion and bottleneck due to the large demands made on their services. A sudden spike in Web content requests may cause heavy workload on particular Web server(s), and as a result a hotspot can be generated. Coping with such unexpected demand causes significant strain on a Web server. Eventually the Web servers are totally overwhelmed with the sudden increase in traffic, and the Web site holding the content becomes temporarily unavailable.

Content providers view the Web as a vehicle to bring rich content to their users. A decrease in service quality, along with high access delays mainly caused by long download times, leaves the users in frustration. Companies earn significant financial incentives from Web-based e-business.

Hence, they are concerned to improve the service quality experienced by the users while accessing their Web sites. As such, the past few years have seen an evolution of technologies that aim to improve content delivery and service provisioning over the Web. When used together, the infrastructures supporting these technologies form a new type of network, which is often referred to as content network.

Several content networks attempt to address the performance problem through using different mechanisms to improve the Quality of Service (QoS). One approach is to modify the traditional Web architecture by improving the Web server hardware adding a high-speed processor, more memory and disk space, or maybe even a multi-processor system. This approach is not flexible. Moreover, small enhancements are not possible and at some point, the complete server system might have to be replaced. Caching proxy deployment by an ISP can be beneficial for the narrow bandwidth users accessing the Internet.

In order to improve performance and reduce bandwidth utilization, caching proxies are deployed close to the users. Caching proxies may also be equipped with technologies to detect a server failure and maximize efficient use of caching proxy resources. Users often configure their browsers to send their Web request through these caches rather than sending directly to origin servers. When this configuration is properly done, the user’s entire browsing session goes through a specific caching proxy. Thus, the caches contain most popular content viewed by all the users of the caching proxies.

A provider may also deploy different levels of local, regional, international caches at geographically distributed locations. Such arrangement is referred to as hierarchical caching. This may provide additional performance improvements and bandwidth savings.

A more scalable solution is the establishment of server farms. It is a type of content network that has been in widespread use for several years. A server farm is comprised of multiple Web servers, each of them sharing the burden of answering requests for the same Web site. It also makes use of a Layer 4-7 switch, Web switch or content switch that examines content request and dispatches them among the group of servers. A server farm can also be constructed with surrogates instead of a switch.

This approach is more flexible and shows better scalability. Moreover, it provides the inherent benefit of fault tolerance. Deployment and growth of server farms progresses with the upgrade of network links that connects the Web sites to the Internet. Although server farms and hierarchical caching through caching proxies are useful techniques to address the Internet Web performance problem, they have limitations. In the first case, since servers are deployed near the origin server, they do little to improve the network performance due to network congestion. Caching proxies may be beneficial in this case. But they cache objects based on client demands. This may force the content providers with a popular content source to invest in large server farms, load balancing, and high bandwidth connections to keep up with the demand. To address these limitations, another type of content network has been deployed in late 1990s. This is termed as Content Distribution Network or Content Delivery Network, which is a system of computers networked together across the Internet to cooperate transparently for delivering content to end-users.

With the introduction of CDN, content providers started putting their Web sites on a CDN. Soon they realised its usefulness through receiving increased reliability and scalability without the need to maintain expensive infrastructure. Hence, several initiatives kicked off for developing infrastructure for CDNs. As a consequence, Akamai Technologies evolved out of an MIT research effort aimed at solving the flash crowd problem. Within a couple of years, several companies became specialists in providing fast and reliable delivery of content, and CDNs became a huge market for generating large revenues. The flash crowd events like the 9/11 incident in USA, resulted in serious caching problems for some site. This influenced the CDN providers to invest more in CDN infrastructure development, since CDNs provide desired level of protection to Web sites against flash crowds.

First generation CDNs mostly focused on static or Dynamic Web documents. On the other hand, for second generation of CDNs the focus has shifted to Video-on-Demand (VoD), audio and video streaming. But they are still in research phase and have not reached to the market yet. With the booming of the CDN business, several standardization activities also emerged since vendors started organizing themselves. The Internet Engineering Task Force (IETF) as a official body took several initiatives through releasing RFCs (Request For Comments). Other than IETF, several other organizations such as Broadband Services Forum (BSF), ICAP forum, Internet Streaming Media Alliance took initiatives to develop standards for delivering broadband content, streaming rich media content – video, audio, and associated data – over the Internet.

In the same breath, by 2002, large-scale ISPs started building their own CDN functionality, providing customized services. In 2004, more than 3000 companies were found to use CDNs, spending more than $20 million monthly. A market analysis shows that CDN providers have doubled their earnings from streaming media delivery in 2004 compared to 2003. In 2005, CDN revenue for both streaming video and Internet radio was estimated to grow at 40%.

A recent marketing research shows that combined commercial market value for streaming audio, video, streaming audio and video advertising, download media and entertainment was estimated at between $385 million to $452 million in 2005. Considering this trend, the market was forecasted to reach $2 billion in four-year (2002-2006) total revenue in 2006, with music, sports, and entertainment subscription and download revenue for the leading content categories. However, the latest report from AccuStream iMedia Research reveals that since 2002, the CDN market has invested $1.65 billion to deliver streaming media (excluding storage, hosting, applications layering), and the commercial market value in 2006 would make up 36% of the $1.65 billion four-year total in media and entertainment, including content, streaming advertising, movie and music downloads and User Generated Video (UGV) distribution.

Insight into CDNs

A typical content delivery environment where the replicated Web server clusters are located at the edge of the network to which the end-users are connected. A content provider (*i.e.* customer) can sign up with a CDN provider for service and have its content placed on the content servers.

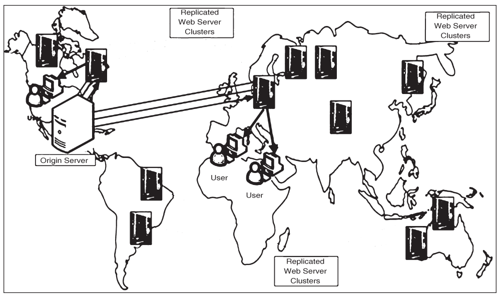

Replicated Web Server Clusters
Replicated Web Server Clusters
Origin Server
User
User
Replicated Web Server Clusters

Fig. Abstract Architecture of a Content Delivery Network (CDN)

The content is replicated either on-demand when users request for it, or it can be replicated beforehand, by pushing the content to the surrogate servers. A user is served with the content from the nearby replicated Web server. Thus, the user ends up unknowingly communicating with a replicated CDN server close to it and retrieves files from that server.

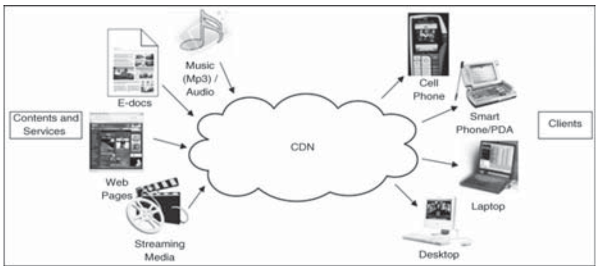


Fig. Content/services Provided by a CDN

CDN providers ensure the fast delivery of any digital content. They host third-party content including static content (*e.g.* static HTML pages, images, documents, software patches), streaming media (*e.g.* audio, real time video), User Generated Videos (UGV), and varying content services (*e.g.* directory service, e-commerce service, file transfer service). The sources of content include large enterprises, Web service providers, media companies and news broadcasters. The end-users can interact with the CDN by specifying the content/service request through cell phone, smart phone/PDA, laptop and desktop. The different content/services served by a CDN provider to end-users.

CDN providers charge their customers according to the content delivered (*i.e.* traffic) to the end-users by their surrogate servers. CDNs support an accounting mechanism that collects and tracks client usage information related to request-routing, distribution and delivery. This mechanism gathers information in real time and collects it for each CDN component. This information can be used in CDNs for accounting, billing and maintenance purposes. The average cost of charging of CDN services is quite high, often out of reach for many small to medium enterprises (SME) or not-for-profit organizations.

The most influencing factors affecting the price of CDN services include:

- Bandwidth cost
- Variation of traffic distribution
- Size of content replicated over surrogate servers
- Number of surrogate servers
- Reliability and stability of the whole system and security issues of outsourcing content delivery A CDN is essentially aimed at content providers or customers who want to ensure QoS to the end-users while accessing their Web content.

The analysis of present day CDNs reveals that, at the minimum, a CDN focuses on the following business goals: scalability, security, reliability, responsiveness and performance.

Scalability: The main business goal of a CDN is to achieve scalability. Scalability refers to the ability of the system to expand in order to handle new and large amounts of data, users and transactions without any significant decline in performance. To expand in a global scale, CDNs need to invest time and costs in provisioning additional network connections and infrastructures. It includes provisioning resources dynamically to address flash crowds and varying traffic. A CDN should act as a shock absorber for traffic by automatically providing capacity-on-demand to meet the requirements of flash crowds. This capability allows a CDN to avoid costly over-provisioning of resources and to provide high performance to every user.

Security: One of the major concerns of a CDN is to provide potential security solutions for confidential and high-value content. Security is the protection of content against unauthorized access and modification. Without proper security control, a CDN platform is subject to cyber fraud, distributed denial-of-service (DDoS) attacks, viruses, and other unwanted intrusions that can cripple business. A CDN aims at meeting the stringent requirements of physical, network, software, data and procedural security. Once the security requirements are met, a CDN can eliminate the need for costly hardware and dedicated component to protect content and transactions. In accordance to the security issues, a CDN combat against any other potential risk concerns including denial-of-service attacks or other malicious activity that may interrupt business.

Reliability, Responsiveness and Performance: Reliability refers to when a service is available and what are the bounds on service outages that may be expected. A CDN provider can improve client access to specialized content through delivering it from multiple locations. For this a fault-tolerant network with appropriate load balancing mechanism is to be implemented. Responsiveness implies, while in the face of possible outages, how soon a service would start performing the normal course of operation. Performance of a CDN is typically characterized by the response time (*i.e.* latency) perceived by the end-users. Slow response time is the single greatest contributor to customers' abandoning Web sites and processes. The reliability and performance of a CDN is affected by the distributed content location and routing mechanism, as well as by data replication and caching strategies.

Hence, a CDN employs caching and streaming to enhance performance especially for delivery of media content. A CDN hosting a Web site also focuses on providing fast and reliable service since it reinforces the message that the company is reliable and customer-focused.

TAXONOMY OF WIRELESS MICRO-SENSOR NETWORK MODELS

Advances in hardware and wireless network technologies have placed us at the doorstep of a new era where small wireless devices will provide access to information anytime, anywhere as well as actively participate in creating smart environments. One of the applications of smart spaces is sensor networks, networks that are formed when a set of small untethered sensor devices that are deployed in an *ad hoc* fashion cooperate on sensing a physical phenomenon. Sensor networks hold the promise of revolutionizing sensing in a wide range of application domains because of their reliability, accuracy, flexibility, cost-effectiveness, and ease of deployment. To motivate the challenges in designing sensor networks, consider the following scenarios: sensors are rapidly deployed in a remote in hospitable area for a surveillance application; sensors are used to analyse the motion of a tornado; sensors are deployed in a forest for fire detection; sensors are attached to taxi cabs in a large metro politan area to study the traffic conditions and plan routes effectively; and smart Kindergarten where sensor networks are deployed to create a developmental problem-solving environment for early childhood education.

Clearly, there is a wide range of applications for sensor networks with differing requirements. We believe that a better understanding of micro-sensor network requirements as well as the underlying differences between micro-sensor applications is needed to assist designers. To this end, in this paper we attempt to classify wireless micro-sensor networks. In particular, we classify the aspects of wireless micro-sensor networks that we believe are most relevant to communication. We examine the characteristics and goals of typical micro-sensor networks as well as the different types of communication that are required to achieve these goals. We compare different data delivery models and network dynamics to create a taxonomy of wireless micro-sensor network communication. We believe that this taxonomy will aid network designers in making better decisions regarding the organization of the network, the network protocol and information dissemination models. Furthermore, it will aid in developing realistic sensor network models and benchmarks for use in future sensor network research.

SENSOR NETWORK ARCHITECTURE

A sensor network is a tool for measuring and relaying information about the phenomenon to the observer within the desired performance bound and deployment cost. As such, the organization of the network may be viewed as follows:

Infrastructure: The infrastructure consists of the sensors and their current deployment status. More specifically, the infrastructure is influenced by the characteristics of the sensors (*e.g.*, sensing accuracy, memory size, battery life, transmission range) and deployment strategy (*e.g.*, sensor density, sensor location, sensor mobility).

Network Protocol: The network protocol is responsible for creating paths and accomplishing communication between the sensors and the observer(s).

Application/Observer: The observer(s) interests in the phenomenon are queries from the observer(s) about the phenomenon as approximated by the distributed data that the sensors are capable of sensing. These queries could be static (the sensors are preprogrammed to report data according to a specific pattern) or dynamic.

The network may participate in synthesizing the query; we consider such intelligence to be part of the translation process between observer interests and low-level implementation.

In this work, we focus on classifying issues that influence the second level: the network protocol. The other two levels only with regard to issues that influence communication. Thus, we do not address the difficult problem of translation between the observer query and the specific low-level interests. This translation could be done by the application software at the observer and/or the sensor nodes, or directly by a human observer. Similarly, we do not discuss the engineering of the infrastructure.

We also note that there is a significant opportunity for optimizations that cut across the three organizational levels. For example, Bhatnagar *et al.* discuss supporting QoS for sensor networks. More specifically, they discuss discriminating among the type of data that the sensors are reporting and preferentially treating high priority data (for example, by giving it priority in forwarding and using redundancy to increase the chance of its reception). This is an example of an optimization where application-level knowledge provides hints to the network protocol. As another example, consider the case where the deployment of the sensors is chosen to mirror the expected motion pattern of the phenomenon or the interests of the observer. Such a deployment strategy incorporates application knowledge in the infrastructure design.

The network protocol in a sensor network is responsible for supporting all communication, both among sensor nodes as well as between the sensor nodes and the observer(s). The performance of the protocol will be highly influenced by the network dynamics, as well as by the specific data delivery model employed. In order to determine how the network protocol behaves for different scenarios, it is important to classify these features. We classify the different types of communication required in a sensor network and then look at the possible data delivery models and network dynamics.

COMMUNICATION MODELS

There are multiple ways for a sensor network to achieve its accuracy and delay requirements; a well designed network meets these requirements while optimizing the sensor energy usage and providing fault tolerance. By studying the communication patterns systematically, the network designer will be able to choose the infrastructure and communication protocol that provide the best combination of performance, robustness, efficiency and deployment cost. Conceptually, communication within a sensor network can be classified into two categories: application and infrastructure. The network protocol must support both these types of communication. Application communication relates to the transfer of sensed data (or information obtained from it) with the goal of informing the observer about the phenomena. Within application communication, there are two models: cooperative and non-cooperative. Under the cooperative sensor model, sensors communicate with other sensors to realise the observer interest. This communication is beyond the relay function needed for routing. For example, in a clustering protocol a cluster-head and the sensor nodes communicate with each other for information dissemination related to the actual phenomenon. In-network data processing is an example of co-operative sensors. Non-cooperative sensors do not cooperate for information dissemination.

Infrastructure communication refers to the communication needed to configure, maintain and optimize operation. More specifically, because of the *ad hoc* nature of sensor networks, sensors must be able to discover paths to other sensors of interest to them and to the observer regardless of sensor mobility or failure. Thus, infrastructure communication is needed to keep the network functional, ensure robust operation in dynamic environments, as well as optimize overall performance. We note that such infrastructure communication is highly influenced by the application interests since the network must reconfigure itself to best satisfy these interests. As infrastructure communication represents the overhead of the protocol, it is important to minimize this communication while ensuring that the network can support efficient application communication. In sensor networks, an initial phase of infrastructure communication is needed to set up the network. Furthermore, if the sensors are energy-constrained, there will be additional communication for reconfiguration. Similarly, if the sensors are mobile or the observer interests dynamic, additional communication is needed for path discovery/reconfiguration. For example, in a clustering protocol, infrastructure communication is required for the formation of clusters and cluster-head selection; under mobility or sensor failure, this communication must be repeated (periodically or upon detecting failure). Finally, infrastructure communication is used for network optimization. Consider the Frisbee model, where the set of active sensors follows a moving phenomenon to optimize energy efficiency. In this case, the sensors wake up other sensors in the network using infrastructure communication.

Sensor networks require both application and infrastructure communication. The amount of required communication is highly influenced by the networking protocol used. Application communication is optimized by reporting measurements at the minimal rate that will satisfy the accuracy and delay requirements given sensor abilities and the quality of the paths between the sensors and the observer. The infrastructure communication is generated by the networking protocol in response to application requests or events in the network. Investing in infrastructure communication can reduce application traffic and optimize overall network operation.

Data Delivery Models

Ideally, the observer interest is specified in terms of the phenomenon, allowing the observer to be oblivious to the underlying sensor network infrastructure and protocol. The query is implemented as one or more specific low-level interests (*e.g.*, requesting a specific sensor to report a specific measurement at some specific interval). Sensor networks can be classified in terms of the data delivery required by the application (observer) interest as: continuous, eventdriven, observer-initiated and hybrid. These models govern the generation of the application traffic. In the continuous model, the sensors communicate their data continuously at a prespecified rate. The authors in showed that clustering is most efficient for static networks where data is continuously transmitted. For dynamic sensor networks, depending upon the degree of mobility, clustering may be applicable as well. In the event-driven data model the sensors report information only if an event of interest occurs. In this case, the observer is interested only in the occurrence of a specific phenomenon or set of phenomena. In the observer-initiated (or request-reply) model, the sensors only report their results in response to an explicit request from the observer (either directly, or indirectly through other sensors). Finally, the three approaches can coexist in the same network; we refer to this model as the hybrid model.

Thus far, we have only discussed data delivery from the application perspective, and not the actual flow of data packets between the sensors and the observer; this is a routing problem subject to the network protocol. We can classify the routing approach as: flooding (broadcastbased), unicast, or multicast/other. Using a flooding approach, sensors broadcast their information to their neighbours, who rebroadcast this data until it reaches the observer. This approach incurs high overhead but is immune to dynamic changes in the topology of the network. Research has been conducted on techniques such as data aggregation that can be used to reduce the overhead of the broadcast. Alternatively, the sensors can either communicate to the observer directly (possibly using a multi-hop routing protocol) or communicate with a cluster-head using one-to-one unicast. Finally, in a multicast approach, sensors form application-directed groups and use multicast to communicate among group members. The observer could communicate with any member of the group to obtain the desired data. A major advantage of flooding or broadcast is the lack of a complex network layer protocol for routing, address and location management; existing sensor network efforts have mostly relied on this approach. However, the overhead of a broadcasting approach may be prohibitive. It is likely that the interaction between the data delivery model from the application and the routing model employed by the network protocol will significantly impact the performance of the network. Consider a scenario where a sensor network is deployed for intrusion detection. In this case, the data delivery model is event driven – the event being an intruder entering the area. If the network level routing model is flooding based, in such a case physically co-located sensors will in general sense the intruder at the same time and try to send data to the observer simultaneously. These concurrent communications in the neighbourhood will contend with each other for the use of the communication medium, raising: (1) the probability of loss of critical information; and (2) the latency in event reporting. A similar problem is studied by Woo and Culler.

NETWORK DYNAMICS MODELS

A sensor network forms a path between the phenomenon and the observer. The goal of the sensor network protocol is to create and maintain this path (or multiple paths) under dynamic conditions while meeting the application requirements of low energy, low latency, high accuracy, and fault tolerance. Without loss of generality, this discussion assumes a single observer. Multiple observers can be supported as multiple instances of a single observer. More sophisticated protocols could also capitalize on the presence of multiple observers to merge related interests and/or optimize communication.

The problem of setting up paths for information dissemination is similar to the problem of routing in *ad hoc* networks. However, there are a few critical differences, including: (i) the sensors are not generally addressed individually; rather, the interest is in the set of sensors that are in a position to contribute to the active observer interests. The sensors could be addressed by attributes of the sensor (*e.g.*, their capabilities) and/or the phenomenon (*e.g.*, the sensors close to a lion in a habitat monitoring scenario). The mapping between the observer interest and a set of sensors is influenced by the network dynamics and the application; and (ii) nodes along the path can take an active role in the information dissemination and processing. In this respect, sensor networks are similar to Active Networks whereas ad hoc networks are traditional "passive" networks. There are several approaches to construct and maintain a path between observer and phenomenon. These will differ depending on the network dynamics, which we classify as: static sensor networks and mobile sensor networks. We focus on mobility because it is the most common source of dynamic conditions; other sources include sensor failure and changes in observer interests.

Static Sensor Networks

In static sensor networks, there is no motion among communicating sensors, the observer and the phenomenon. An example is a group of sensors spread for temperature sensing. For these types of sensor networks, previous studies have shown that localized algorithms can be used in an effective way. The sensors in localized algorithms communicate with nodes in their locality. An elected node relays a summary of the local observations to the observer, perhaps through one on more levels of hierarchy. Such algorithms extend the lifetime of the sensor network because they trade-off local computation for communication. In this type of network, sensor nodes require an initial set-up infrastructure communication to create the path between the observer and the sensors with the remaining traffic exclusively application communication.

Dynamic Sensor Networks

In dynamic sensor networks, either the sensors themselves, the observer, or the phenomenon are mobile. Whenever any of the sensors associated with the current path from the observer to the phenomenon moves, the path may fail. In this case, either the observer or the concerned sensor must take the initiative to rebuild a new path. During initial set-up, the observer can build multiple paths between itself and the phenomenon and cache them, choosing the one that is the most beneficial at that time as the current path. If the path fails, another of the cached paths can be used. If all the cached paths are invalid then the observer must rebuild new paths. This observer-initiated approach is a reactive approach, where path recovery action is only taken after observing a broken path.

Another model for rebuilding new paths from the observer to the phenomenon is a sensor-initiated approach. In a sensor-initiated path recovery procedure, path recovery is initiated by a sensor that is currently part of the logical path between the observer and the phenomenon and is planning to move out of range. The sensor might perform some local patching procedure to build a new path by broadcasting a participation request for a given logical flow to all its neighbouring sensors. Any one of the neighbouring sensors can send a participation reply message to the given initiator sensor indicating willingness to participate and become a part of the requested path. If none of the neighbouring sensors respond, the sensor can default to sending a path invalidation request to the observer so that the observer can start building the path. This is similar to soft hand-off in traditional Mobile IP based networks. This sensor-initiated approach is a proactive approach where path recovery operations are begun in anticipation of a future broken path. Dynamic sensor networks can be further classified by considering the motion of the components. This motion is important from the communications perspective since the degree and type of communication is dependent on network dynamics. We believe that each of the following require different infrastructures, data delivery models, and protocols:

Mobile observer: In this case the observer is mobile with respect to the sensors and phenomena.

An example of this paradigm is sensors deployed in an inhospitable area for environment monitoring. For example, a plane might fly over a field periodically to collect information from a sensor network. Thus the observer, in the plane, is moving relative to the sensors and phenomena on the ground.

Mobile sensors: In this case, the sensors are moving with respect to each other and the observer. For example, consider traffic monitoring implemented by attaching sensors to taxis. As the taxis move, the attached sensors continuously communicate with each other about their own observations of the traffic conditions. If the sensors are co-operative, the communication paradigm imposes additional constraints such as detecting the link layer addresses of the neighbours and constructing localization and information dissemination structures. From previous work, we know that the overhead of maintaining a globally unique sensor ID in a hierarchical fashion like an IP address is expensive and not needed.

Instead, these sensors should communicate only with their neighbours with the link layer MAC address. In such networks, the proactive algorithm with local patching for repairing a path can be used so that the information about the phenomenon is always available to the observer regardless of the mobility of the individual sensors.

Mobile phenomena: In this case, the phenomenon itself is moving. A typical example of this paradigm is sensors deployed for animal detection. In this case the infrastructure level communication should be event-driven. Depending on the density of the phenomena, it will be ineffficient if all the sensor nodes are active all the time. Only the sensors in the vicinity of the mobile phenomenon need to be active. The number of active sensors in the vicinity of the phenomenon can be determined by application specific goals such as accuracy, latency, and energy efficiency. A model that is well-suited to this case is the Frisbee model.

It is important to note that the effect of mobility in sensor networks is fundamentally different than that in traditional wireless networks. Mobility in *ad hoc* networks has been addressed from the point of view of mobility of one or more of the communicating nodes during communication. However, since the sensors themselves are of no interest to the observer, their mobility is not necessarily of interest; rather, the sensor network must adapt its operation to continue to reflect the observer interests in the presence of mobility. Thus, the mobility of the sensing nodes themselves should be handled in a different way than for *ad hoc* networks; for example, a node that is moving away from a phenomenon may choose to hand-off the responsibility of monitoring to a closer node as it drifts away.

CONSIDERATIONS SURROUNDING THE STUDY OF PROTECTION

General Observations: As computers become better understood and more economical, every day brings new applications. Many of these new applications involve both storing information and simultaneous use by several individuals. The key concern in this paper is multiple use. For those applications in which all users should not have identical authority, some scheme is needed to ensure that the computer system implements the desired authority structure.

For example, in an airline seat reservation system, a reservation agent might have authority to make reservations and to cancel reservations for people whose names he can supply. A flight boarding agent might have the additional authority to print out the list of all passengers who hold reservations on the flights for which he is responsible. The airline might wish to withhold from the reservation agent the authority to print out a list of reservations, so as to be sure that a request for a passenger list from a law enforcement agency is reviewed by the correct level of management.

The airline example is one of protection of corporate information for corporate self-protection (or public interest, depending on one's view). A different kind of example is an online warehouse inventory management system that generates reports about the current status of the inventory. These reports not only represent corporate information that must be protected from release outside the company, but also may indicate the quality of the job being done by the warehouse manager. In order to preserve his personal privacy, it may be appropriate to restrict the access to such reports, even within the company, to those who have a legitimate reason to be judging the quality of the warehouse manager's work.

Many other examples of systems requiring protection of information are encountered every day: credit bureau data banks; law enforcement information systems; time-sharing service bureaus; on-line medical information systems; and government social service data processing systems. These examples span a wide range of needs for organizational and personal privacy. All have in common controlled sharing of information among multiple users. All, therefore, require some plan to ensure that the computer system helps implement the correct authority structure.

Of course, in some applications no special provisions in the computer system are necessary. It may be, for instance, that an externally administered code of ethics or a lack of knowledge about computers adequately protects the stored information. Although there are situations in which the computer need provide no aids to ensure protection of information, often it is appropriate to have the computer enforce a desired authority structure.

The words "privacy," "security," and "protection" are frequently used in connection with information-storing systems. Not all authors use these terms in the same way. This chapter uses definitions commonly encountered in computer science literature.

The term "privacy" denotes a socially defined ability of an individual (or organization) to determine whether, when, and to whom personal (or organizational) information is to be released.

This chapter will not be explicitly concerned with privacy, but instead with the mechanisms used to help achieve it.

The term "security" describes techniques that control who may use or modify the computer or the information contained in it.

Security specialists have found it useful to place potential security violations in three categories.

- Unauthorized information release: an unauthorized person is able to read and take advantage of information stored in the computer. This category of concern sometimes extends to "traffic analysis," in which the intruder observes only the patterns of information use and from those patterns can infer some information content. It also includes unauthorized use of a proprietary programme.
- Unauthorized information modification: an unauthorized person is able to make changes in stored information—a form of sabotage. Note that this kind of violation does not require that the intruder see the information he has changed.
- Unauthorized denial of use: an intruder can prevent an authorized user from referring to or modifying

information, even though the intruder may not be able to refer to or modify the information. Causing a system "crash," disrupting a scheduling algorithm, or firing a bullet into a computer are examples of denial of use. This is another form of sabotage.

The term "unauthorized" in the three categories listed above means that release, modification, or denial of use occurs contrary to the desire of the person who controls the information, possibly even contrary to the constraints supposedly enforced by the system.

The biggest complication in a general-purpose remote-accessed computer system is that the "intruder" in these definitions may be an otherwise legitimate user of the computer system.

Examples of security techniques sometimes applied to computer systems are the following:

1. Labelling files with lists of authorized users,
2. Verifying the identity of a prospective user by demanding a password,
3. Shielding the computer to prevent interception and subsequent interpretation of electromagnetic radiation,
4. Enciphering information sent over telephone lines,
5. Locking the room containing the computer,
6. Controlling who is allowed to make changes to the computer system (both its hardware and software),

7. Using redundant circuits or programmed cross-checks that maintain security in the face of hardware or software failures,
8. Certifying that the hardware and software are actually implemented as intended.

It is apparent that a wide range of considerations are pertinent to the engineering of security of information. Historically, the literature of computer systems has more narrowly defined the term *protection* to be just those security techniques that control the access of executing programmes to stored information. An example of a protection technique is labelling of computer-stored files with lists of authorized users. Similarly, the term *authentication* is used for those security techniques that verify the identity of a person (or other external agent) making a request of a computer system. An example of an authentication technique is demanding a password. This chapter concentrates on protection and authentication mechanisms, with only occasional reference to the other equally necessary security mechanisms. One should recognize that concentration on protection and authentication mechanisms provides a narrow view of information security, and that a narrow view is dangerous. The objective of a secure system is to prevent all unauthorized use of information, a negative kind of requirement. It is hard to prove that this negative requirement has been achieved, for one must demonstrate that every possible threat has been anticipated. Thus an expansive view of the problem is most appropriate to help ensure that no gaps appear in the strategy. In contrast, a narrow concentration on protection mechanisms, especially those logically impossible to defeat, may lead to false confidence in the system as a whole.

Functional Levels of Information Protection

Many different designs have been proposed and mechanisms implemented for protecting information in computer systems. One reason for differences among protection schemes is their different functional properties—the kinds of access control that can be expressed naturally and enforced. It is convenient to divide protection schemes according to their functional properties. A rough categorization is the following.

Unprotected Systems

Some systems have no provision for preventing a determined user from having access to every piece of information stored in the system. Although these systems are not directly of interest here, they are worth mentioning since, as of 1975, many of the most widely used, commercially available batch data processing systems fall into this category—for example, the Disk Operating System for the IBM System 370. Our definition of protection, which excludes features usable only for mistake prevention, is important here since it is common for unprotected systems to contain a variety of mistake-prevention features. These may provide just enough control that any breach of control is likely to be the result of a deliberate act rather than an accident. Nevertheless, it would be a mistake to claim that such systems provide any security.

All-or-nothing Systems

These are systems that provide isolation of users, sometimes moderated by total sharing of some pieces of information. If only isolation is provided, the user of such a system might just as well be using his own private computer, as far as protection and sharing of information are concerned. More commonly, such systems also have public libraries to which every user may have access. In some cases the public library mechanism may be extended to accept user contributions, but still on the basis that all users have equal access. Most of the first generation of commercial timesharing systems provide a protection scheme with this level of function. Examples include the Dartmouth Time-Sharing System (DTSS) and IBM's VM/370 system. There are innumerable others.

Controlled Sharing

Significantly more complex machinery is required to control explicitly who may access each data item stored in the system. For example, such a system might provide each file with a list of authorized users and allow an owner to distinguish several common patterns of use, such as reading, writing, or executing the contents of the file as a programme. Although conceptually straightforward, actual implementation is surprisingly intricate, and only a few complete examples exist. These include M.I.T.'s Compatible Time-Sharing System (CTSS), Digital Equipment Corporation's DECsystem/10, System Development Corporation's Advanced Development Prototype (ADEPT) System, and Bolt, Beranek, and Newman's TENEX

User-programmed Sharing Controls

A user may want to restrict access to a file in a way not provided in the standard facilities for controlling sharing. For example, he may wish to permit access only on weekdays between 9:00 A.M. and 4:00 P.M. Possibly, he may wish to permit access to only the average value of the data in a file. Maybe he wishes to require that a file be modified only if two users agree. For such cases, and a myriad of others, a general escape is to provide for user-defined *protected objects* and *subsystems*. A *protected subsystem* is a collection of programmes and data with the property that only the programmes of the subsystem have direct access to the data (that is, the protected objects). Access to those programmes is limited to calling specified entry points. Thus the programmes of the subsystem completely control the operations performed on the data. By constructing a protected subsystem, a user can develop any programmable form of access control to the objects he creates. Only a few of the most advanced system designs have tried to permit user-specified protected subsystems. These include Honeywell's Multics, the University of California's CAL system, Bell Laboratories' UNIX system, the Berkeley Computer Corporation BCC-500, and two systems currently under construction: the CAP system of Cambridge University, and the HYDRA system of Carnegie-Mellon University. Exploring alternative mechanisms for implementing protected subsystems is a current research topic. A specialized use of protected subsystems is the implementation of protection controls based on data content. For example, in a file of salaries, one may wish to permit access to all salaries under $15 000. Another example is permitting access to certain statistical aggregations of data but not to any individual data item. This area of protection raises questions about the possibility of discerning information by statistical tests and by examining indexes, without ever having direct access to the data itself. Protection based on content is the subject of a variety of recent or current research projects and will not be explored in this tutorial.

Putting Strings on Information

The foregoing three levels have been concerned with establishing conditions for the release of information to an executing programme. The fourth level of capability is to maintain some control over the user of the information even *after* it has been released. Such control is desired, for example, in releasing income information to a tax advisor; constraints should prevent him from passing the information on to a firm which prepares mailing lists. The printed labels on classified military information declaring a document to be "Top Secret" are another example of a constraint on information after its release to a person authorized to receive it. One may not (without risking severe penalties) release such information to others, and the label serves as a notice of the restriction. Computer systems that implement such strings on information are rare and the mechanisms are incomplete. For example, the ADEPT system keeps track of the classification level of all input data used to create a file; all output data are automatically labelled with the highest classification encountered during execution.

There is a consideration that cuts across all levels of functional capability: the *dynamics of use*. This term refers to how one establishes and changes the specification of who may access what. At any of the levels it is relatively easy to envision (and design) systems that statically express a particular protection intent. But the need to change access authorization dynamically and the need for such changes to be requested by executing programmes introduces much complexity into protection systems. For a given functional level, most existing protection systems differ primarily in the way they handle protection dynamics. To gain some insight into the complexity introduced by programme-directed changes to access authorization, consider the question "Is there any way that O'Hara could access file X?" One should check to see not only if O'Hara has access to file X, but also whether or not O'Hara may change the specification of file X's accessibility. The next step is to see if O'Hara can change the specification of who may change the specification of file X's accessibility, etc. Another problem of dynamics arises when the owner revokes a user's access to a file while that file is being used. Letting the previously authorized user continue until he is "finished" with the information may not be acceptable, if the owner has suddenly realised that the file contains sensitive data. On the other hand, immediate withdrawal of authorization may severely disrupt the user. It should be apparent that provisions for the dynamics of use are at least as important as those for static specification of protection intent.

In many cases, it is not necessary to meet the protection needs of the person responsible for the information stored in the computer entirely through computer-aided enforcement. External mechanisms such as contracts, ignorance, or barbed wire fences may provide some of the required functional capability. However, is focused on the internal mechanisms.

Design Principles

Whatever the level of functionality provided, the usefulness of a set of protection mechanisms depends upon the ability of a system to prevent security violations. In practice, producing a system at any level of functionality (except level one) that actually does prevent all such unauthorized acts has proved to be extremely difficult. Sophisticated users of most systems are aware of at least one way to crash the system, denying other users authorized access to stored information. Penetration exercises involving a large number of different general-purpose systems all have shown that users can construct programmes that can obtain unauthorized access to information stored within. Even in systems designed and implemented with security as an important objective, design and implementation flaws provide paths that circumvent the intended access constraints. Design and construction techniques that systematically exclude flaws are the topic of much research activity, but no complete method applicable to the construction of large general-purpose systems exists yet. This difficulty is related to the negative quality of the requirement to prevent *all* unauthorized actions.

In the absence of such methodical techniques, experience has provided some useful principles that can guide the design and contribute to an implementation without security flaws. Here are eight examples of design principles that apply particularly to protection mechanisms.

Economy of Mechanism

Keep the design as simple and small as possible. This well-known principle applies to any aspect of a system, but it deserves emphasis for protection mechanisms for this reason: design and implementation errors that result in unwanted access paths will not be noticed during normal use (since normal use usually does not include attempts to exercise improper access paths). As a result, techniques such as line-by-line inspection of software and physical examination of hardware that implements protection mechanisms are necessary. For such techniques to be successful, a small and simple design is essential.

Fail-safe Defaults

Base access decisions on permission rather than exclusion. This principle, suggested by E. Glaser in 1965, means that the default situation is lack of access, and the protection scheme identifies conditions under which access is permitted. The alternative, in which mechanisms attempt to identify conditions under which access should be refused, presents the wrong psychological base for secure system design. A conservative design must be based on arguments why objects should be accessible, rather than why they should not. In a large system some objects will be inadequately considered, so a default of lack of permission is safer. A design or implementation mistake in a mechanism that gives explicit permission tends to fail by refusing permission, a safe situation, since it will be quickly detected. On the other hand, a design or implementation mistake in a mechanism that explicitly excludes access tends to fail by allowing access, a failure which may go unnoticed in normal use. This principle applies both to the outward appearance of the protection mechanism and to its underlying implementation.

Complete Mediation

Every access to every object must be checked for authority. This principle, when systematically applied, is the primary underpinning of the protection system. It forces a system-wide view of access control, which in addition to normal operation includes initialization, recovery, shutdown, and maintenance.

It implies that a foolproof method of identifying the source of every request must be devised. It also requires that proposals to gain performance by remembering the result of an authority check be examined skeptically. If a change in authority occurs, such remembered results must be systematically updated.

Open Design

The design should not be secret. The mechanisms should not depend on the ignorance of potential attackers, but rather on the possession of specific, more easily protected, keys or passwords.

This decoupling of protection mechanisms from protection keys permits the mechanisms to be examined by many reviewers without concern that the review may itself compromise the safeguards. In addition, any skeptical user may be allowed to convince himself that the system he is about to use is adequate for his purpose. Finally, it is simply not realistic to attempt to maintain secrecy for any system which receives wide distribution.

Separation of Privilege

Where feasible, a protection mechanism that requires two keys to unlock it is more robust and flexible than one that allows access to the presenter of only a single key. The relevance of this observation to computer systems was pointed out by R. Needham in 1973. The reason is that, once the mechanism is locked, the two keys can be physically separated and distinct programmes, organizations, or individuals made responsible for them. From then on, no single accident, deception, or breach of trust is sufficient to compromise the protected information. This principle is often used in bank safe-deposit boxes. It is also at work in the defence system that fires a nuclear weapon only if two different people both give the correct command. In a computer system, separated keys apply to any situation in which two or more conditions must be met before access should be permitted. For example, systems providing user-extendible protected data types usually depend on separation of privilege for their implementation.

Least Privilege

Every programme and every user of the system should operate using the least set of privileges necessary to complete the job. Primarily, this principle limits the damage that can result from an accident or error. It also reduces the number of potential interactions among privileged programmes to the minimum for correct operation, so that unintentional, unwanted, or improper uses of privilege are less likely to occur. Thus, if a question arises related to misuse of a privilege, the number of programmes that must be audited is minimized. Put another way, if a mechanism can provide "firewalls," the principle of least privilege provides a rationale for where to install the firewalls. The military security rule of "need-to-know" is an example of this principle.

Least Common Mechanism

Minimize the amount of mechanism common to more than one user and depended on by all users. Every shared mechanism (especially one involving shared variables) represents a potential information path between users and must be designed with great care to be sure it does not unintentionally compromise security. Further, any mechanism serving all users must be certified to the satisfaction of every user, a job presumably harder than satisfying only one or a few users.

For example, given the choice of implementing a new function as a supervisor procedure shared by all users or as a library procedure that can be handled as though it were the user's own, choose the latter course. Then, if one or a few users are not satisfied with the level of certification of the function, they can provide a substitute or not use it at all. Either way, they can avoid being harmed by a mistake in it.

Psychological Acceptability

It is essential that the human interface be designed for ease of use, so that users routinely and automatically apply the protection mechanisms correctly. Also, to the extent that the user's mental image of his protection goals matches the mechanisms he must use, mistakes will be minimized. If he must translate his image of his protection needs into a radically different specification language, he will make errors.

5

Valiant Load Balanced Networks

The motivation for Valiant Load-balancing is that it is much easier to estimate the aggregate traffic entering and leaving a node than to estimate the traffic matrix. The approach assumes a full-mesh topology among the nodes, and load-balances traffic over two-hop paths.

GENERAL APPROACH

Consider a backbone network consisting of multiple PoPs interconnected by long-haul links. The whole network is arranged as a hierarchy, and each PoP connects an access network to the backbone.

Although traffic matrices are hard to obtain, it is straightforward to measure, or estimate, the total amount of traffic entering (leaving) a PoP from (to) its access network. When a new customer joins the network, we add its aggregate traffic rate to the node. When new locations are planned, the aggregate traffic demand for a new node can be estimated from the population that the node serves.

This is much easier than trying to estimate the traffic rates from one node to every other node in the backbone. Imagine that we establish a full mesh of logical links from each node to every other node.

They are not necessarily physical links; they could be implemented using tunnelling or an overlay mechanism. Traffic entering the backbone is spread equally across all the nodes. Each packet traverses the backbone twice: once from the ingress node to an arbitrary intermediate node, and once again to reach the egress node.

A flow is load-balanced across every two-hop path from its ingress to egress node. This means the capacity requirement on each link is lower than the node capacity, and we'll see that a network designed this way can serve any valid traffic matrix. What's more, it is possible to calculate the exact amount of capacity needed on every logical link in the load-balanced network, making network design more systematic. Since each flow is load-balanced over multiple paths through the network, only a small amount of excess capacity is required in order to restore from a small number of failures. More importantly, we can systematically determine how much extra capacity is needed to recover from a given number of failures.

Forwarding packets twice through the backbone network leads to increased average delay. However, we believe it is a reasonable tradeoff for improved predictability, and lower delay variance. And while many network customers claim they are sensitive to fixed delays, there are few, if any, applications that would really suffer from a two-hop traversal of the network. We believe the fixation on delay is only because it is easier to measure than bandwidth. The VLB architecture can easily reroute around link and router failures. When a failure happens, the network doesn't need to rely on a complicated routing protocol to discover and converge to new routes. Failure recovery and restoration in a VLB network can be as fast as failure detection at a single router.

Details of the Approach

Consider a homogeneous network of *N* identical nodes, each with capacity *r*, where capacity is the aggregate capacity of the access network that the backbone node serves.

This means a node can initiate traffic to, or receive traffic from, other backbone nodes at an aggregate rate up to *r*. The analysis can be extended to heterogeneous networks but space limitations preclude us from including it here.

ARCHITECTURE OF NETWORKING

In communications, data is transferred wither on parallel or serial mode. Parallel communication is faster but requires more wires. Serial communications is much slower but requires fewer wires. Serial is more practical for communication with remote sites, and the existing telephone networks use serial communications for linking computers.

LOCAL AREA NETWORK

LAN is normally transmit data in a digital form with typical transmission speeds of up to several megabytes per second. These speeds can be achieved using parallel transmission where a cable with multiple core is used, or serial transmission, making use of high frequency carriers, using coaxial cables, fibre optics or even a simple pair of wires because distances are short. Modems are not normally required although some mechanism for converting from parallel to serial transmission and back again may be needed. The terminals and workstations are able to connect to either one of the two host computers at will. The network also has a file server and a print server. The former is a special computer which provides a form of auxiliary storage which can be used by any other computer on the network and the latter is a special computer which can receive data from other computers on the network and print it. There is also an external communication server on the LAN, which enables communication between equipment on the network and system elsewhere.

Local Area Network link computers in the same area for the purpose of sharing information and hardware. Usually the computers are within 300 metres of each other, because they must be connected by a cable hookup, which can be expensive. People at the work stations in a LAN gain more capabilities in work processing, data processing, information retrieval and communication without duplication of equipment, databases and activities. LAN has become popular. Many businesses are installing LAN in order to improve the efficiency of office functions and to facilitate office automation. The configuration of the LAN can be a star or a ring topology or simply devises attached along a length of cable.

Description

LAN or Local Area Network is the most common kind of network set up. There are two ways to connect a LAN network. The simplest and easiest way is the peer-to-peer connection network. This is when two or more computers are directly connected to each other. For example if there were four computers in the network, computer 1 would be connected to computer 2, computer 2 would be connected to computer 3 and computer 3 would be connected to computer 4. This means each computer is dependent on the other. And if there were a network problem with any one computer, all of them would be affected. The other type if the client server connection.

A local area network, or LAN, is a network of connected computers in a room, building, or set of buildings. Local area networks have been around since the beginning of computer use. A LAN is defined as a user network whereby data is sent at high rates between people located relatively close to each other. LANs do not usually make use of leased communication lines, but only means of communication that are provided by the installer of the network.

The Internet is a wide area network, or WAN, which is distinct from a LAN. In contrast to the term *Internet*, local area networks are often called *intranets*, though sometimes this term refers to a cluster of LANs associated with a particular company or organization but not connected to the larger Internet. A local area network uses a hub or router to connect computers together. The means of communication is the omnipresent Ethernet cable or wireless wi-fi technology. These technologies offer data transfer rates running between 10 to 10000 Mbit/s.

Perhaps the most frequently employed use of a LAN is to connect users to the Internet with only one connected router. In modern times, we use broadband cable or DSL modems to connect to the Internet, and it would be clumsy to have a modem associated with every computer, so we simply plug the modem into a router and link the router to computers with Ethernet cables. Configuring a LAN can be intimidating at first, but contemporary operating systems have programmes that do most of the necessary configurations automatically, so setting up a local area network is pretty easy.

A local area network (LAN) supplies networking capability to a group of computers in close proximity to each other such as in an office building, a school, or a home. A LAN is useful for sharing resources like files, printers, games or other applications. A LAN in turn often connects to other LANs, and to the Internet or other WAN.

ARCHITECTURE AND STRUCTURE

The importance of having standardised networking protocols have led to the evolution of several networking reference models. The Open System Interconnection (OSI) and the TCP/IP models will be looked at. These models allow us to more easily study the structure of networks.

OSI Reference Model

The OSI model is based on a proposal by the International Standards Organisation. The OSI reference model has a layered architecture, whereby each layer encapsulates a function of the overall complex task. The proposed model has 7 layers. The key idea is that although data is transmitted vertically, each layer is programmed horizontally. Each layer ignores the complications of the lower layers and assumes that the information it passes out is immediately seen by destination host. At each layer, appropriate headers are appended onto the data when sending, and are removed as the data is decoded. Since the headers from the lower levels are removed before the next higher level looks at the data, the concept of horizontal coding is preserved.

Layer 1: The physical layer, and is concerned with transmitting raw bits over a communication channel.

Layer 2: The data link layer, handles error correction and flow regulation.

Layer 3: The network layer. It is concerned with the operation of the subnet, that is, it determines how packets are routed from source to destination. Routes can be based on hardwired static tables, or can be highly dynamic, where each packet is assigned a different route through the network, depending on the network load. In broadcast networks, routing is simple, and hence the network layer is often thin or non-existent.

Layer 4: The transport layer. Data accepted from the session layer is split into smaller units if need be and passed to the network layer. It is also in charge of ensuring that the information arrives correctly on the other end. Connections across the network, are established and deleted from this layer. There are mechanisms to regulate flow of information between fast and slow host machines. This flow control is distinct from the flow control talked about in layer 2.

Layer 5: The session layer implements a set of rules for establishing and terminating data streams between hosts. Services provided include dialogue control(full of half duplex), token management and other end to end data control.

Layer 6: The presentation layer, and it performs certain functions requested sufficiently often enough to warrant finding a general solution for them, rather than letting each user solve them. It manages between different abstract data types like ASCII, Unicode, integer representation, one's and two's complements etc. This allows computers with different data representations to communicate.

Layer 7: The application layer and is charged with translating between different applications. For example, for a screen editor to work over a network, it has to know the different escape codes of different terminals. Similarly with FTP. Different systems have different file naming conventions, different ways of representing text. The applications layer takes care of these complications.

TCP/IP Reference

- *Layer 4 is the application layer*: The TCP/IP model does not have a session or presentation layer. No need for them were perceived. Hence the application layer here corresponds to layers 5-7 in the OSI model. It contains higher protocols such as FTP, TELNET, HTTP etc.

- *Layer 3, the transport layer corresponds to the layer 4 in the OSI model*: The transport layer was designed to allow peer entities on the source and destination hosts to establish a conversation. Two end to end protocols are defined here. Transmission Control Protocol (TCP) is a reliable connection oriented protocol that allows a byte stream originating from one machine to de delivered without error to any other machine in the Internet. Incoming messages are fragmented into discrete messages before being transmitted. At the destination, TCP reassembles the received messages into the output stream. The second protocol, User Datagram Protocol (UDP) is an unreliable connectionless protocol. It is used for applications that do not want TCP's sequencing or flow control and wish to implement their own. It is widely used for one shot, client server type request-reply queries and applications in which prompt delivery is more important than accurate delivery. *E.g.* speech or video streams.
- *Layer 2 is the Internet layer*: The requirement that the network should survive a loss of subnet hardware,

led to the choice of a packet switching network based on a connectionless layer. The job of the Internet layer is to permit hosts to inject packets into the network and have them travel independently to the destination. The Internet layer defines an official format and protocol called Internet Protocol (IP). The primary job here is one of packet routing and avoidance of congestion. It is very similar to the OSI network layer.

- *The final layer is the host to network layer*: The TCP/IP model does not say much about what occurs here, except to point out that the host has to connect to the network using some protocol so it can send IP packets to the network.

Hardware Components

Here we will examine the different hardware that makes up a network. The most basic hardware that makes up a network is the transmission medium. The oldest and most common transmission medium is twisted pair wires. It consists of two insulated copper wires, typically about 1mm thick. The purpose of twisting is to reduce electrical interference.

The bandwidth depends on the thickness of the wires and distance travelled, but several megabytes/sec can be achieved for a few kilometres. Twisted pair cabling comes in several varieties, one of which is the Unshielded Twisted Pair (UTP) category 5 cabling. Category 5 UTP wires have more twists per centimetre than other twisted pair cabling, and has Teflon insulation. This results in less cross talk and a better quality signal over longer distances.

Coaxial cables are another popular form of transmission medium. It has better shielding than UTP so it can span for longer distances and at higher speeds. A coax cable is made up of a copper core, covered by a layer of insulating material, a braided outer conductor and finally a protective plastic covering. For 1 kilometre cables, bandwidths of 1 to 2 Gbps are feasible.

Fibre optic cables have a glass core centre, from which light propagates. It is next enclosed by a cladding of glass with a lower index of refraction than the core. This keeps the light within the core. A plastic jacket is the last layer and is to protect the cladding. Fibres are typically grouped into bundles, protected by another outer sheath.

The achievable bandwidth (about 1 Gbps) of fibre is current restricted by the ability to convert between electrical to optical signals. Speeds of 100Gbps are feasible and terabit performance is just a few years down the road. Fibre has many advantages over copper. It can handle higher bandwidths, due to lower attenuation, it only requires repeaters every 30 km as opposed to 5 km for copper. Fibre is also not affected by power surges, electromagnetic interference or power failures. Fibre does not leak light and thus is difficult to tap into. Fibre is light weight and its lower installation cost makes it a good choice over copper.

A repeater operates at the physical layer. It regenerates a signal received on one cable segment and retransmits it on another cable segment. A repeater can be used to extend the coverage of a network, and also to connect networks. Although a network can be extended with a repeater, a network is still constrained by the maximum permissible length of that LAN. Since a repeater functions at the physical layer, it is transparent to data flow and hence is limited to connecting two similar networks. Repeaters also cause problems with traffic.

If two networks are connected via a repeater, all messages from one network is passed to the other network, regardless of the intended recipient. If implemented without knowledge concerning the traffic flow on each LAN, performance problems will arise.

Bridges are more intelligent than repeaters. A bridge connects at the data link layer, where it can examine each frame that passes through it.

The bridge looks at the destination address of the incoming frame; if the destination is for the network across the bridge, it translates the frame, and retransmits it on the other network. If the frame is for a destination from the source network, the bridge repeats back the frame. There are typically two types of bridges. Transparent bridge and translating bridge. A transparent bridge connects two networks that employ the same protocol at the data link layer, whereas a translating bridge provides a connection capability between two networks that have differing protocols at the data link layer.

Routers operate one level higher, at the network layer. By operating at the network layer, routers are able to make decisions about how packets are routed between networks. Although multiported bridges can be said to have routing capabilities, this is usually for one point to point link within a network. A router has the ability to fragment and dynamically re-route the message across networks, making use of the most efficient paths. Routers are known to workstations that use their service, hence packets can be sent directly to routers. This means that routers do not have to examine in detail every packet it receives, and this makes them more efficient. However, the higher functionality of routers over bridges means that on average, routers perform a half to two thirds more processing over bridges.

Gateways function through all layers. Essentially, gateways perform protocol translation between networks. Gateways are generally designed and used for LAN-WAN connections and not for inter LAN communications.

The idea of intelligent hubs came about because running cables all over the building made it hard to configure and repair networks. Instead, collapse the LAN topology and put it in a box; terminate all devices in the network at the box using separate wires for each device. This centralised location makes it more convenient for network configuration, management and monitoring. Intelligent hubs are a level above normal hubs and wiring cabinets.

Intelligent hubs are becoming key network management points as functionality such as bridges, routers and servers are built into them. Some intelligent hubs provide remote management capabilities that makes it easier to diagnose problems at distant locations and isolate faults from the rest of the network.

6

Process of Troubleshooting

Outbound e-mail uses a process called SMTP (simple mail transport protocol). SMTP is the standard for e-mail transmissions across the Internet. SMTP is generally used to send messages from a mail client (Thunderbird) to a mail server. The e-mail is stored on the server until retrieved by another e-mail client using a POP3 or IMAP retrieval protocol. Outbound e-mail problems can be frustrating to deal with because you rarely see an indicator as to what might be causing the problem. However, there are fewer factors involved when setting up the SMTP section in Thunderbird, so isolating the problem can be much simpler. Unlike setting up an e-mail retrieval for inbound e-mail, with SMTP, a single SMTP setup can be used to cover all of your e-mail addresses, regardless of how many different ISPs are involved.

Most SMTP mail servers are set up to allow e-mail accounts from other ISPs to be run through them. Thunderbird identifies the first SMTP account that you set up as being the default account. That is because in most cases it can be shared by all of your e-mail accounts.

Most of the configuration issues with sending e-mail can be resolved in the Server Settings section in Thunderbird.

- Select Account Settings from the Tools menu.
- Scroll down to the bottom of the list if e-mail accounts.
- Select Outgoing Server (SMTP).

When you are sure that this information is correct, click the OK button, shut down Thunderbird and then start it up again to make sure that any new settings are loaded. If you have multiple e-mail accounts and you know that inbound e-mail is working, try sending a test message to another of your e-mail addresses. If it still does not work or you see an error, make sure that you Internet access is active. If you can access the Internet using your browser, you should be able to send e-mail messages once the SMTP information is correct.

If you are still having trouble connecting to the SMTP server, Thunderbird will usually display a message, but depending on the type of error, sometimes it does not. Most of the time the User Name is just the account name, but some configurations may require a complete e-mail address. If one way does not work, try the other.

If your SMTP server requires a secure connection, you may have to check the SSL box. I sometimes use the SMTP server at my AT&T account. AT&T connections usually require that you check the SSL box. If you are having connections problems, post your issues in the Comments below. Try to give us as much information as you can and we will see if we can help you to get connected.

INTERNET MAIL PROTOCOL

Most e-mail addresses today use the POP3 (Post Office Protocol 3) Internet protocol for receiving e-mail. With POP3 your e-mail is held on a mail server. You can download your mail to an e-mail client. Typically, the e-mail is deleted from the server when it is downloaded, but it can be set up to retain the e-mail messages.

An alternative Internet mail protocol is IMAP (Internet Message Access Protocol). IMAP provides some enhanced capabilities and is useful for dealing with large volumes of mail that may need to be organized. Most hosing companies today use POP3. If you are not sure about your situation, ask the ISP or hosting company. The e-mail documentation they provide will usually tell you which protocol you are working with.

Most of the configuration issues with receiving e-mail can be resolved in the Server Settings section for each e-mail address in Thunderbird.

Each e-mail address is configured separately:

- Select Account Settings from the Tools menu.
- Select Server Settings under the e-mail address you wish to troubleshoot.

The first thing to check is the Server Name. This identifies your mail server. You will need to get this information from your hosting company or ISP. Most mail servers are identified by using a subdomain preceding the domain name, such as mail.tech-evangelist.com. Some use other subdomain names, such as ipostoffice.tech-evangelist.com. Once again, the only way to know is to find out how your mail server is configured.

Next, check the User Name. Sometimes this is the entire e-mail address and sometimes is is just the account name, which is the portion of the e-mail address to the left of the @ sign. The third item to test is the Security Settings. Most mail servers do not use a secure connection, so Never would be checked. I've seen a few that use TLS (Transport Layer Security), which is a type of secure connection. If Never didn't work, try TLS, if available.

There are also a few e-mail servers that use SSL (Secure Sockets Layer) for secure connections. Secure connections encypt the messages as they are downloaded, which prevents others from intercepting your messages. I have an AT&T e-mail account that requires SSL. When you click the SSL radio button, the port number will change. You can also try checking the Use Secure Authentication box, but thus far I have never had to do that when setting up an e-mail account.

There is a very small possibility that your ISP or hosting company is using a non-standard port. Thunderbird does allow you to change the port, but you should never do that unless instructed to do so by your ISP or hosting company. If the port is wrong, you will not make the proper server connection. If none of this works, make sure that you do indeed have an active Internet connection. If your browser can access the Internet, you should be able to retrieve your e-mail.

By now your problems receiving e-mail should have been resolved. Most of the time when I've run into issues setting up new e-mail accounts in Thunderbird, the issue was an incorrect User Name or Security Setting. Remember to save the settings each time you make a change by clicking the OK button. It is also best if you shut down and restart Thunderbird to make sure that your changes are reloaded. After that, test it again.

THUNDERBIRD INCOMING E-MAIL PROBLEMS

Troubleshooting connection problems with an e-mail client, such as the freebie Mozilla Thunderbird e-mail programme can be frustrating and tricky. If you are having problems retrieving your e-mail or getting it configured correctly, don't give up. Most solutions are pretty simple. In this article we cover some of the more common connection issues you might find when using Thunderbird.

Most e-mail addresses today use the POP3 (Post Office Protocol 3) Internet protocol for receiving e-mail. With POP3 your e-mail is held on a mail server. You can download your mail to an e-mail client. Typically, the e-mail is deleted from the server when it is downloaded, but it can be set up to retain the e-mail messages.

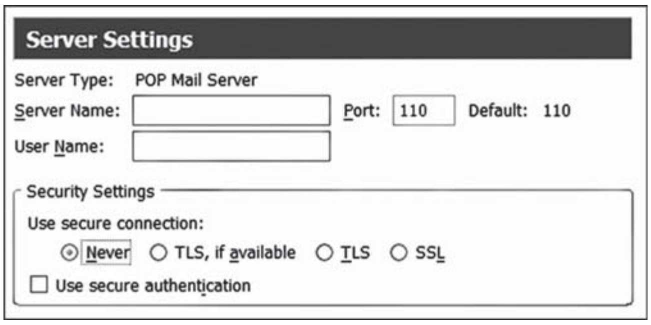
An alternative Internet mail protocol is IMAP (Internet Message Access Protocol). IMAP provides some enhanced capabilities and is useful for dealing with large volumes of mail that may need to be organized. Most hosing companies today use POP3. If you are not sure about your situation, ask the ISP or hosting company. The e-mail documentation they provide will usually tell you which protocol you are working with. Most of the configuration issues with receiving e-mail can be resolved in the Server Settings section for each e-mail address in Thunderbird.

Each e-mail address is configured separately:

1. Select Account Settings from the Tools menu.
2. Select Server Settings under the e-mail address you wish to troubleshoot.

If anything in this section is incorrect, you will not be able to download your e-mail. After making each change, click the OK button on this screen, close Thunderbird, re-open

Thunderbird, and try to download your e-mail messages once again. If you do not have detailed instructions from your ISP or hosting company, this is a trial-and-error process.



Server Settings

Server Type: POP Mail Server

Server Name: [ ] Port: 110 Default: 110

User Name: [ ]

Security Settings

Use secure connection:

[✓ Never] TLS, if available TLS SSL

☐ Use secure authentication

The first thing to check is the Server Name. This identifies your mail server. You will need to get this information from your hosting company or ISP. Most mail servers are identified by using a subdomain preceding the domain name, such as mail.tech-evangelist.com. Some use other subdomain names, such as ipostoffice.tech-evangelist.com. Once again, the only way to know is to find out how your mail server is configured.

Next, check the User Name. Sometimes this is the entire e-mail address and sometimes is is just the account name, which is the portion of the e-mail address to the left of the @ sign. The third item to test is the Security Settings. Most mail servers do not use a secure connection, so Never would be checked. I've seen a few that use TLS (Transport Layer Security), which is a type of secure connection. If Never didn't work, try TLS, if available.

There are also a few e-mail servers that use SSL (Secure Sockets Layer) for secure connections. Secure connections encypt the messages as they are downloaded, which prevents others from intercepting your messages. I have an AT&T e-mail account that requires SSL. When you click the SSL radio button, the port number will change. You can also try checking the Use Secure Authentication box, but thus far I have never had to do that when setting up an e-mail account.

There is a very small possibility that your ISP or hosting company is using a non-standard port. Thunderbird does allow you to change the port, but you should never do that unless instructed to do so by your ISP or hosting company. If the port is wrong, you will not make the proper server connection. If none of this works, make sure that you do indeed have an active Internet connection. If your browser can access the Internet, you should be able to retrieve your e-mail. By now your problems receiving e-mail should have been resolved. Most of the time when I've run into issues setting up new e-mail accounts in Thunderbird, the issue was an incorrect User Name or Security Setting. Remember to save the settings each time you make a change by clicking the OK button. It is also best if you shut down and restart Thunderbird to make sure that your changes are reloaded. After that, test it again.

SIGNATURE IN AN E-MAIL

A signature in an e-mail is kind of like a footer on a web page. It is generally used to convey contact information, legal notices and other repetitive information. This is something that you may not want to re-type every time you send an e-mail message. Thunderbird allows you to create as many signatures as you wish. You can assign a different signature to each individual e-mail address.

The first thing you need to know is that you must create the signature file in a pure text editor, such as Microsoft's Notepad. Do not use Word or any type of text editor that may embed codes in the text. You can save these files anywhere on your PC, but it makes sense to store them somewhere where they will not get separated from the rest of your Thunderbird files. In a previous tutorial, I changed the file location for the Thunderbird files in order to make it easier to back up these files.

I simply created a new folder called "signatures" in the folder with the rest of the Thunderbird files. Like I said, you can store them anywhere, but when you do it this easy, the signatures stay with your Thunderbird files if you need to move them to another PC or if you have to restore a PC from your backup files.

Next, you will need to assign a signature to each e-mail address. From the Tools menu, select Account Settings... Highlight an e-mail address on the left-hand column. On the right, check the box that says "Attach this signature:" and then click the Choose button to navigate to your signatures folder. Select the file that you want to attach. Click the OK button to save your settings. You will need to repeat this process for each e-mail address where you wish to add a signature. If you add a URL to the signature file, most e-mail clients will turn it into a hyperlink. But if you need to spruce it up of if you wish to use HTML in a signature instead of plain text, here is what you need to do. Create a text file in the same manner as described above. When you save the file, save it with a.html extension, rather than a.txt extension. Thunderbird will then recognize it as an HTML file. You can then use coloured fonts, hyperlinks or just about any type of CSS, as long as you use inline CSS styles.

When you are using CSS in e-mail messages, make sure that you fully specify the URLs to images and store the images on your server. Don't try to embed images or send them with the message.

It won't work well with many browsers.One more thing that has to be mentioned about HTML in signatures is that the message must be sent in HTML mode for the HTML and CSS to work. When you send any message that contains HTML, Thunderbird will ask you if you want to send it in in Plain Text. You do have to click on the Send in HTML Only or Send in Plain Text and HTML for the HTML and CSS to render properly.

SPAM AND JUNK MAIL

No one want to receive spam, but today it is almost inevitable that most e-mail addresses will eventually end up on spam lists. I have always protected my e-mail addresses, but recently ended up receiving more than 100 spam messages per day on my most important e-mail address used only by my business clients.

It turns out that a client thought she was doing me a favour when she posted it on a forum, along with a recommendation. As part of my battle to eliminate spam, I found that Thunderbird contains some amazingly powerful spam and junk mail controls that are very effective at filtering unwanted e-mail. This tutorial shows you how to turn these filters on and configure them to meet your needs.

There are actually several methods for filtering out spam using Thunderbird's tools. I used all of them in conjunction and found that Thunderbird effectively filtered out almost every undesirable message. It is also amazingly accurate and only very rarely grabs a legitimate message and identifies it as spam.

PRE-FILTERING SPAM MESSAGES

The first level for setting up spam filtering is to do it at the server level or on your personal computer.

If you have a web site, your hosting company might already offer spam filtering software, such as Spam Assassin or Spam Pal. This step is not absolutely necessary, but it does make the process of identifying and filtering spam more effective.

SpamAssassin is commonly found on Linux or Unix Apache servers. SpamAssassin checks messages coming through a hosting service's e-mail system. It does not actually filter the spam out, but rather uses heuristic algorithms that analyse a message. It then assigns a spam score. Thunderbird can use this score to determine if a message is likely to be spam. If your web site runs on Unix or Linux and uses a control panel such as Pleske or cPanel, check to see if SpamAssassin is included. You will find it under Mail, SpamAssassin. If the message near the top indicates that SpamAssassin is disabled, click the box labeled Enable SpamAssassin. Do not click the box labeled Enable Spam Box.

SpamPal is a different type of filtering system that installs on your PC. SpamPal uses spam block lists (lists of IP addresses used by spammers) to identify spam messages. I have not tried it, but if your hosting company does not use SpamAssassin, or if you do not have a web site (and therefore do not use a hosting company), you might want to try installing it. You will find it atspampal.org.

CONFIGURING JUNK MAIL

The first step is to configure Junk Settings:

- Select Account Settings from the Tools menu. Thunderbird allows you to configure each e-mail address separately.
- Select Junk Settings under one of your e-mail addresses.

- Check all of the boxes in the Junk Settings display. If you don't use SpamAssassin or SpamPal, skip that checkbox. If you do use one of these tools, make sure the box is checked and the proper filter selected.
- Checking "Move new junk messages to Junk folder" will create a Junk folder under the selected e-mail address. You can set the nuber of days before spam is deleted from the junk folder. It is a good idea to move the messages to this folder so that you can periodically review it for messages that may not be spam.

CONFIGURING PRIVACY SETTINGS

- Select Tools, Options.
- Select the Privacy button.
- Select the Junk tab.
- Choose how you want spam to be handled when you manually select it when reviewing your e-mails. When you are viewing messages and you identify a spam message, you can click on the column to the left of the date to send the message to the Junk folder. This also helps the system identify spam messages.

CONFIGURING MESSAGE FILTERS

When all else fails to identify spam, this is the way to do it. Message filters will search new messages for any keywords you designate. This section is pretty easy to figure out. Once again, you have to set up individual message filters for each e-mail address.

- Highlight an e-mail address in the main screen.
- Select Tools, Message Filters from the top menu.
- Click on the New button.
- Name your message filter at the top.
- Select the e-mail element that you wish to filter. Most spam keywords are going to be found in the Subject or Body.
- Select Contains.
- Enter the keyword or phrase you wish to use as a filter.
- Near the bottom of the displayed dialogue box, select the location to move any identified spam messages. The Junk folder for the current e-mail address is the most logical place for these messages so that you can review them.
- Click OK.

That's about all there is to configuring Thunderbird to filter out spam messages. Although the configuration sections are a bit scattered throughout Thunderbird, once you know where to find them they are easy to use.

PROGRAMMER

When a non-technically inclined web site owner speaks with an experienced programmer, technical jargon frequently enters the conversation. Some programmers do that intentionally, just to show off their expertise. For others, it is just the way they talk because they are immersed in a technical working environment. If you get confused talking with IT people, you may find this jargon guide to be handy.

The technical language of programmers can sometimes be very difficult to understand. It is loaded with shortcuts, euphemisms and analogies.

Frequently, the full name or technical term for the many special symbols or characters used in programming is just too cumbersome or boring to use in everyday technical conversations. Unique names or monikers for programming symbols has evolved. Most are intended to be humorous and help to liven up the communications between programmers.

The following are a few of the most common jargon and techno-terms used by professional programmers in everyday conversations amongst themselves. Step outside of your box for a few moments and compare the special characters to the jargon terms and you can easily envision the evolution of the terms.

- . *(dot)*: A dot is just a period in a file name or domain name, as in google-dot-com.
- * *(star, splat or spider)*: An asterisk is a wild card symbol used with servers and computer programmes.

- # (hash, pound, crunch, sharp)–Most commonly referred to as a hash, this symbol is commonly used to designate comments in many programming languages.
- *! (bang, pling, shriek, not)*: The exclamation point serves different purposes in different programming languages, but it is most commonly used to reverse a Boolean evaluation. In other words, if a statement is true, it reverses it to false.
- *#! (hash-bang or sh-bang)*: When used as the first two characters on the first line of a programming script, a hash-bang causes Unix and Linux operating systems to execute that script using the interpreter specified by the rest of that line.
- *' (tick):* This is a single quote commonly used to surround a string of characters.
- *" (snakebite, rabbit ears)*: A standard double quote. This jargon goes way back to the 1980s, so if you hear this, you are talking to an old-timer.
- *$ (buck, bling, cash)*: Used to designate a string variable in many programming languages.
- *() (left paren, right paren, open, close)*: Commonly used to group portions of algorithms or comparison statements. Algorithms and evaluations of statements always start with the inner-most set of parens in all programming languages.

- */ (wack, slash, stroke)*: A common slash character, most commonly used in directory paths on servers or to define a mathematical division.
- *\ (backslash, backwack, hack)*: A common blackslash used in some networking paths and also as a common designation for a escape character, which means that the next character in line should be taken literally and not executed.
- *{ } (left curly, right curly, hitchcocks*: Think Alfred Hitchcock silhouette, left banana, right banana)– Curly braces commonly used to group multiple programming statements.
- *~ (squiggle)*: A tilde character.
- *| (bar, pipe)*: A vertical bar character.
- *^ (hat)*: A caret character.
- *% (mod, grapes)*: Commonly used as a mathematical modulo character in programming languages.
- *& (amp, amper)*: An ampersand symbol.

VIEW SOURCE

If you are an avid Internet user and use Internet Explorer as your browser, you have probably noticed that there are times when the View Source function in Internet Explorer occasionally stops working.

For a Web developer or designer who likes to view the HTML code for a Web site, this can be frustrating. Fortunately, both the explanation to the problem and the solution are usually simple.

The root of the problem is that the browser cache has filled because of all the Web page code and images stored by the browser. The solution is to clear these temporary files and start out with a fresh cache. To do this,

- Select Tools on the top menu in Internet Explorer
- Select Internet Options...
- Click Delete Files in the Temporary Internet Files section

It may take several minutes to delete all the files in the cache, so don't panic if it appears like your PC has locked up. As soon as the hour glass once again turns into a mouse pointer, you can shut down Internet Explorer and start it up again. The View Source feature should now work once again.

If this does not fix the problem, you may have a more serious issue, such as a corrupted or deleted Notepad.exe programme file or problems related to the temporary files directory in Windows. Those problems are beyond the scope of this article. Other issues that have been reported that sometimes disable View Source include having a shortcut to Notepad on your desktop or having a shortcut named Notepad on your desktop. Renaming the shortcut should resolve those issues.

Most of the time, the problem tends to be related to the cache, so if you spend a lot of time on the Internet, it might be a good idea to clear the cache once a week.

MOZILLA THUNDERBIRD TIPS AND FAQS

Mozilla Thunderbird is, in my opinion, the absolute best free e-mail programme available. It doesn't have the vulnerabilities found with Outlook, primarily due to the fact that it is not an "attack target" like Microsoft products. Even when vulnerabilities are found, they are quickly patched by the community of developers that support Thunderbird. We've put together a list of tips to make your use of the Thunderbird e-mail application more enjoyable.

If you have a specific question regarding how to do something with Thunderbird, post a comment below and if we have a solution we will include the answer on either this tips page or in a separate tutorial.

Where do I Find the Download for Thunderbird

You can find the most current version at the Mozilla Thunderbird download page. Installation is pretty simple. If you are using Outlook, Eudora or another e-mail client, you can easily import your address book and messages during the installation.

Including Previous Message Text

We found this to be one of the annoying default features with Thunderbird. Unless you change the settings, when you forward a message with Thunderbird it will include the forwarded message as an attachment. I personally prefer that the forwarded text be included in the body of the message that I send.

This is easy to change:

- Click on the Tools menu
- Select Options
- Select Composition
- Select the General tab
- *At the top change Forward messages*: "As Attachment" to "Inline"
- Click OK

Your forwarded messages should now include the text of the forwarded message in the body of the e-mail just below any message text that you add. If you selected the default settings during the installation, Thunderbird will automatically download updates and let you know when they are ready for installation. You will find the settings for this under Tools, Options, Advanced and then select the Update tab. Check out our tutorial about setting up Thunderbird mailing lists. This is really simple to do, but like many things with open source software, the solution is not always intuitive. This is an area where Thunderbird excels. We did a separate post on that called Thunderbird spam and junk mail filters. We tried several different methods to eliminate spam and this was the only system that eliminated it almost completely and at zero cost. We were very impressed.

String E-mail messages

They are hidden, but you can find them using the tips we offer in Thunderbird file location. This is very useful information if you wish to move the location to a Windows folder that can be easily backed up.

Changing Passwords

Ah-h-h, grasshopper. Passwords used to access your e-mail accounts are easy to change if you follow the instructions we offer in Changing Thunderbird Passwords

Making Mozilla Thunderbird

When you set up Thunderbird as the default e-mail client, it will automatically open whenever you click on an e-mail address on a web page that uses the HTML mailto: protocol. To configure Thunderbird as the default e-mail client, simply Open the Tools menu, select Options and click on the General icon. Under system Defaults, check the box that says, "Always check to see if Thunderbird is the default e-mail client on startup." Click OK to save the setting. From now on, whenever the PC is booted it will check to see if Thunderbird is the default e-mail client. If one of the Windows programmes tries to change the setting to Outlook or if another programme changes it to something else, a message will be displayed that will allow you to change it back.

Most people are not aware of this, but when you delete an e-mail message with most e-mail clients, you do not actually delete the message. The message is simple flagged so that it does not appear in the e-mail list. Thunderbird never actually deletes a message unless you take some extra steps. First, select File, Empty Trash. Second, select File, Compact Folders. This deletes all of the flagged messages in the data files. This issue keeps coming up. It looks like you can use Thunderbird with a Yahoo account if you subscribe to Yahoo Mail Plus. The standard freebie Yahoo account does not offer POP3 access. See the following articles. You will find all of the proper configuration information posted there.

- Can I use Thunderbird to read and send my Yahoo! Mail?
- Can I POP my mail into a different e-mail client (like Thunderbird)?
- Yahoo outgoing e-mail problems with Thunderbird.

Preventing Thunderbird

This was a feature that was turned on by default in older versions of Mozilla Thunderbird. I think the default is turned off in newer versions. It is something that is not only annoying, but also presents security issues.

Turning the embedding feature off is easy. Just select the View menu. Make sure that Display Attachments Inline is unchecked. Changing Thunderbird's behaviour with outgoing e-mail attachments is easy to do, but is buried in the Config Editor. Make sure that you are using a current version of Thunderbird.

The location of the Config Editor has changed several times with different versions. Select the Tools menu, then Options, select the Advanced button, then the General tab and click on the Config Editor button. Scroll down until you find mail. content_disposition_type. If the value is not set to 1, double-click on that line and an Enter Integer Value dialog box will pop up. Change the value to 1 and click OK. Shut down Thunderbird and restart it for the change to take effect.

W3C

You may find it interesting to learn how to eliminate every error message the W3C validator produces. It can be an educational process even if you are a very experienced developer. Although you may think you know a lot about HTML or XHTML, you will likely find that you do not know as much as you thought you knew about the W3C's coding standards. Before you use the validator, a page needs to have a Doctype declaration.

The Doctype declaration, or DTD, needs to be the very first line of code on a Web page. The DTD sets the standard to be used in the validation. It also serves another purpose. Many modern browsers contain multiple rendering engines. If you do not set a standard to be used for rendering, the page may not look consistent across different browsers.

Note that all valid DTDs from HTML 4.01 on contain a link to a standards document. If your web page editor already adds a DTD with every Web page, it may not be valid if it does not contain a URL to the actual standards document.

Using the W3C validator is very easy. Just cut-and-paste or enter a complete URL to the Web page you wish to test. You may find some of the validation messages to be a bit cryptic, but the W3C provides links with most error messages that lead to additional information.

There are some common page elements that always produce errors or warnings, even though they do not negatively affect either browsers or search engine spiders. Microsoft-specific entity codes, such as copyright and other symbols may not be code compliant. The validator handles these errors well and typically offers the code compliant equivalent.

Body tag attributes such as leftmargin, right margin (Internet Explorer), marginheight, marginwidth (Netscape) are browser extensions that are not part of the official coding standards. These types of issues will not typically produce negative effects with either browsers or spiders, but do show up as errors because they do not comply with the standard. One type of nuisance error is a missing alt attribute in image tags. Current standards do require the use of the alt attribute in order to push designers and developers to use this attribute because it is beneficial for Internet users with vision impairments who utilize special browsers that describe images though the use of alt attributes. It may not make sense, however, to include text with every alt attribute. The code compliant workaround is to simply include an empty string value (alt="") for lines and other page elements that need no explanation.

It can sometimes be very beneficial to eliminate every error message displayed, and to assure complete and 100% compliance with the DTD standard you chose. Once a page has been validated as compliant, the W3C offers a cut-and-paste snippet of code that can be added to your Web pages as both a "gold star" certification of compliance and as a link to re-validate the code when you make changes.

AVOIDING SPAM

Despite anti-spam laws, the amount of spam e-mail messages I receive has grown enormously. You probably feel the same way. The problem with spam laws is that someone has to enforce them, but the e-mail spam problem is so enormous that only the largest and most aggressive USA spammers get prosecuted. Foreign spammers are not subject to USA laws and are much harder to identify. There are, however, some fundamental things that you can do to lessen the amount of spam that you receive, particularly if you have a business.

First and foremost, the best way to prevent your e-mail address from finding its way to a spam list is to never, under any circumstances, publish the e-mail address on a web page. Why? Because most e-mail ends up on spam lists because one of the hundreds of spam bots–also called e-mail harvesters–found it on a web site and automatically added it to a spam list. If you need to publish an e-mail address, create an image file for the e-mail address and display that. Spambots cannot read text in images, so they will not find the e-mail address if you display it in an image.

Second, never, ever click on a spam message or any links in a spam message. This includes links that assure that the message will be removed from the spammer's database. While this may work for legitimate sources of e-mail, I am willing to bet that few spammers really care about the laws, especially if they are in a foreign country. When you click on any link, you are very likely just verifying that your e-mail address is valid, which pretty much guarantees that it will be added to hundreds of spam lists.

Third, much of the spam that we receive comes from spammer programmes that randomly generate e-mail account names using the domain names of popular ISPs and then send an e-mail to that address. If it bounces back, they know the e-mail address is not valid. If it does not bounce back, it may be a valid e-mail address.

Many spammers use a library of account names that are attached to other libraries of domain names. The account name if the portion of the e-mail address found to the left of the @ symbol. The domain is the part to the right.

Due to the fact that we run several web-based businesses, we have noticed a pattern of account names that many spammers use to see if an e-mail address is valid. The account names are attached to lists of domain names that can easily be found on the web and are used to generate spam for commonly used e-mail addresses.

7

IP and IPX: Network Layer

The network layer is in charge of routing network messages (data) from one computer to another. The common protocols at this layer are IP (which is paired with TCP at the transport layer for Internet network) and IPX (which is paired with SPX at the transport layer for some older Macintosh, Linus, UNIX, Novell and Windows networks). Because of the growth in Internet-based networks, IP/TCP are becoming the leading protocols for most networks.

Every network device (such as network interface cards and printers) have a physical address called a MAC (Media Access Control) address. When you purchase a network card, the MAC address is fixed and cannot be changed. Networks using the IP and IPX protocols assign logical addresses (which are made up of the MAC address and the network address) to the devices on the network, This can all become quite complex — suffice it to say that the network layer takes care of assigning the correct addresses (via IP or IPX) and then uses routers to send the data packets to other networks.

TCP AND SPX (TRANSPORT LAYER)

The transport layer is concerned with efficient and reliable transportation of the data packets from one network to another.

In most cases, a document, e-mail message or other piece of information is not sent as one unit. Instead, it is broken into small data packets, each with header information that identifies its correct sequence and document. When the data packets are sent over a network, they may or may not take the same route — it doesn't matter. At the receiving end, the data packets are re-assembled into the proper order. After all packets are received, a message goes back to the originating network. If a packet does not arrive, a message to "re-send" is sent back to the originating network. TCP, paired with IP, is by far the most popular protocol at the transport level. If the IPX protocol is used at the network layer (on networks such as Novell or Microsoft), then it is paired with SPX at the transport layer.

HTTP, FTP, SMTP AND DNS (SESSION/ PRESENTATION/APPLICATION LAYERS)

Several protocols overlap the session, presentation, and application layers of networks. There protocols listed below are a few of the more well-known:

- DNS- Domain Name System- translates network address (such as IP addresses) into terms understood by humans (such as Domain Names) and *vice-versa*
- DHCP- Dynamic Host Configuration Protocol-can automatically assign Internet addresses to computers and users
- FTP- File Transfer Protocol- a protocol that is used to transfer and manipulate files on the Internet
- HTTP- HyperText Transfer Protocol-An Internet-based protocol for sending and receiving webpages
- IMAP- Internet Message Access Protocol- A protocol for e-mail messages on the Internet
- IRC- Internet Relay Chat- a protocol used for Internet chat and other communications
- POP3- Post Office protocol Version 3- a protocol used by e-mail clients to retrieve messages from remote servers
- SMTP- Simple Mail Transfer Protocol- A protocol for e-mail messages on the Internet.

ADDRESS RESOLUTION PROTOCOL

The address resolution protocol (arp) is a protocol used by the Internet Protocol (IP) [RFC826], specifically IPv4, to map IP network addresses to the hardware addresses used by a data link protocol. The protocol operates below the network layer as a part of the interface between the OSI network and

OSI link layer. It is used when IPv4 is used over Ethernet. The term address resolution refers to the process of finding an address of a computer in a network. The address is "resolved" using a protocol in which a piece of information is sent by a client process executing on the local computer to a server process executing on a remote computer. The information received by the server allows the server to uniquely identify the network system for which the address was required and therefore to provide the required address. The address resolution procedure is completed when the client receives a response from the server containing the required address.

An Ethernet network uses two hardware addresses which identify the source and destination of each frame sent by the Ethernet. The destination address (all 1's) may also identify a broadcast packet (to be sent to all connected computers). The hardware address is also known as the Medium Access Control (MAC) address, in reference to the standards which define Ethernet. Each computer network interface card is allocated a globally unique 6 byte link address when the factory manufactures the card (stored in a PROM). This is the normal link source address used by an interface. A computer sends all packets which it creates with its own hardware source link address, and receives all packets which match the same hardware address in the destination field or one (or more) pre-selected broadcast/multicast addresses.

The Ethernet address is a link layer address and is dependent on the interface card which is used. IP operates at the network layer and is not concerned with the link addresses of individual nodes which are to be used. The address resolution protocol (arp) is therefore used to translate between the two types of address. The arp client and server processes operate on all computers using IP over Ethernet. The processes are normally implemented as part of the software driver that drives the network interface card. There are four types of arp messages that may be sent by the arp protocol. These are identified by four values in the "operation" field of an arp message. The types of message are:

- ARP request
- ARP reply
- RARP request
- RARP reply.

The format of an arp message is shown below:

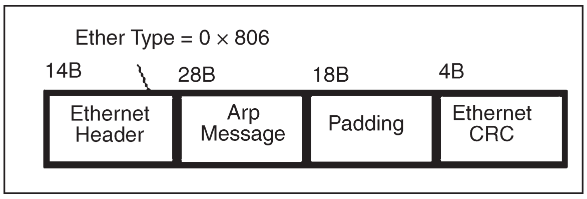
0	8 15	16 31
Hardware Type		Protocol Type
HLEN	PLEN	Operation
Sender HA (octets 0-3)		
Sender HA (octets 4-5)		Sender IP (octets 0-1)
Sender IP (octets 2-3)		Target HA (octets 0-1)
Target HA (octets 2-5)		
Target IP (octets 0-3)		

Format of an arp message used to resolve the remote MAC Hardware Address (HA).

To reduce the number of address resolution requests, a client normally caches resolved addresses for a (short) period of time. The arp cache is of a finite size, and would become full of incomplete and obsolete entries for computers that are not in use if it was allowed to grow without check. The arp cache is therefore periodically flushed of all entries. This deletes unused entries and frees space in the cache. It also removes any unsuccessful attempts to contact computers which are not currently running. If a host changes the MAC address it is using, this can be detected by other hosts when the cache entry is deleted and a fresh arp message is sent to establish the new association. The use of gratuitous arp (*e.g.* triggered when the new NIC interface is enabled with an IP address) provides a more rapid update of this information.

EXAMPLE OF USE OF THE ADDRESS RESOLUTION PROTOCOL (ARP)

The use of arp when a computer tries to contact a remote computer on the same LAN (known as “sysa”) using the “ping” program. It is assumed that no previous IP datagrams have been received form this computer, and therefore arp must first be used to identify the MAC address of the remote computer.



The arp request message (“who is X.X.X.X tell Y.Y.Y.Y”, where X.X.X.X and Y.Y.Y.Y are IP addresses) is sent using the Ethernet broadcast address, and an Ethernet protocol type of value 0 × 806. Since it is broadcast, it is received by all systems in the same collision domain (LAN). This is ensures that is the target of the query is connected to the network, it will receive a copy of the query. Only this system responds. The other systems discard the packet silently.

The target system forms an arp response ("X.X.X.X is hh:hh:hh:hh:hh:hh", where hh:hh:hh:hh:hh:hh is the Ethernet source address of the computer with the IP address of X.X.X.X). This packet is unicast to the address of the computer sending the query (in this case Y.Y.Y.Y). Since the original request also included the hardware address (Ethernet source address) of the requesting computer, this is already known, and doesn't require another arp message to find this out.

GRATUITOUS ARP

Gratuitous ARP is used when a node (end system) has selected an IP address and then wishes to defend its chosen address on the local area network (*i.e.* to check no other node is using the same IP address). It can also be used to force a common view of the node's IP address (*e.g.* after the IP address has changed). Use of this is common when an interface is first configured, as the node attempts to clear out any stale caches that might be present on other hosts. The node simply sends an arp request for itself.

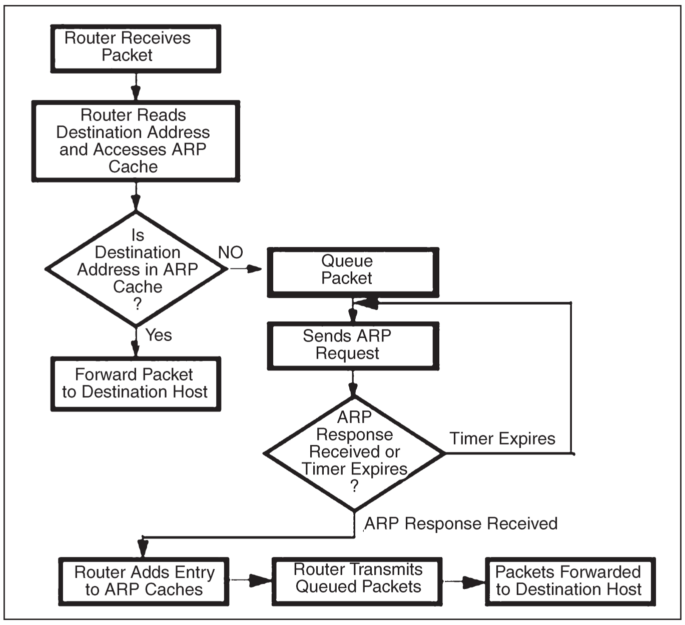
The Address Resolution Protocol (ARP) is a low-level protocol that dynamically learns and maps network layer IP addresses to physical Medium Access Control (MAC) addresses, for example, Ethernet. Given only the network layer IP address of the destination system, ARP lets a router find the MAC address of the destination host on the same network segment. For example, a router receives an IP packet destined for a host connected to one of its LANs. The packet contains only a 32-bit IP destination address. To be able to forward the packet on the LAN, the router must construct the Data Link layer header using the physical MAC address of the destination host. The router must acquire this physical MAC address of the destination host and map that address to the 32-bit IP address.

To obtain the physical address of the host, the router broadcasts an ARP request to all host of the network. Only the host with that IP address responds with its physical MAC address. The router saves the IP/MAC address mapping in a table called ARP cache and it can use this mapping in the future when forwarding packets to the destination host.

RFC : RFC 826 documents the ARP protocol.

ARP Physical Address Broadcast

Note : If the ARP cache does not contain an entry for a destination, the packet is queued pending an ARP Response. This means that the first packet sent between IP Hosts is queued until the expiration of the Time to Retry timer. If an ARP Response is not received within this time an ARP Request is retransmitted. All IP-based protocols perform this function.


Router Receives Packet
Router Reads Destination Address and Accesses ARP Cache
Is Destination Address in ARP Cache ?
NO
Yes
Queue Packet
Sends ARP Request
Forward Packet to Destination Host
ARP Response Received or Timer Expires ?
Timer Expires
ARP Response Received
Router Adds Entry to ARP Caches
Router Transmits Queued Packets
Packets Forwarded to Destination Host

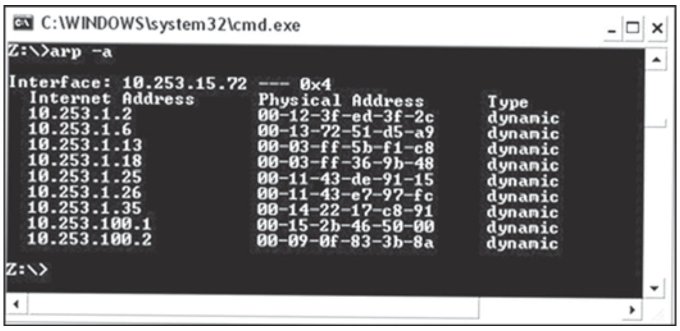
Note: If a second IP packet, intended for the same Destination Address, arrives while the device is awaiting an ARP Response, the packet is queued but a second ARP Request is not sent. When another IP packet, intended for a different Destination Address, arrives while the device is awaiting an ARP Response for the first packet, an ARP Request for the second Destination Address is immediately broadcast to the network.

ARP CACHE

There are two types of ARP entries-static and dynamic. Most of the time, you will use dynamic ARP entries. What this means is that the ARP entry (the Ethernet MAC to IP address link) is kept on a device for some period of time, as long as it is being used. The opposite of a dynamic ARP entry is static ARP entry. With a static ARP entry, you are manually entering the link between the Ethernet MAC address and the IP address. Because of management headaches and the lack of significant negatives to using dynamic ARP entries, dynamic ARP entries are used most of the time.

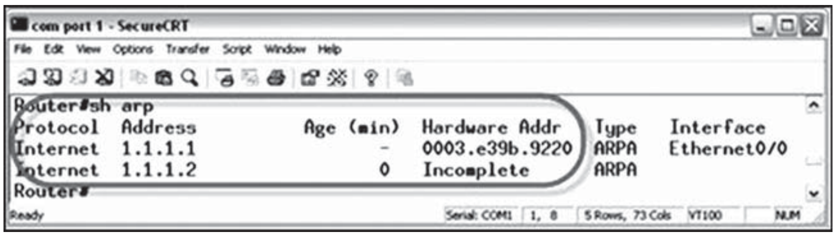
So how is the dynamic ARP entry created? The answer is that the ARP protocol is used. Let's say that a PC wants to communicate with host Myserver.Bluecrabfood.com. Before it can do that, it has to first resolve the hostname with the DNS server. Let's say that it is successfully resolved to 10.10.10.10. Before the PC can communicate with that IP address, it must first resolve the IP address to the MAC address. To do this, it does an ARP request. This is a broadcast to the local LAN that says who has IP address 10.10.10.10 and what is your Ethernet MAC address? Say that server responds and says I have IP address 10.10.10.10 and my MAC address is 1234.4567.890A.

The PC will put that entry into its local ARP cache and it will stay there until the entry has not been used and the ARP cache timeout has expired. Here is an ARP cache looks like on a Windows PC:



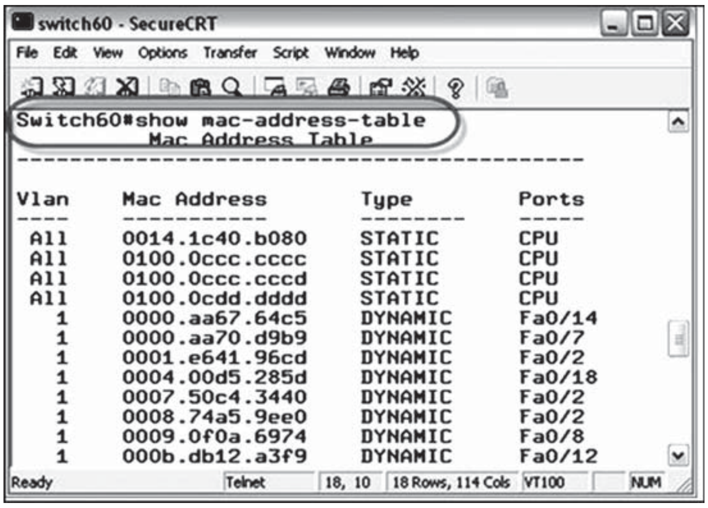
```
Z:\>arp -a

Interface: 10.253.15.72 --- 0x4
  Internet Address      Physical Address      Type
  10.253.1.2            00-12-3f-ed-3f-2c     dynamic
  10.253.1.6            00-13-72-51-d5-a9     dynamic
  10.253.1.13           00-03-ff-5b-f1-c8     dynamic
  10.253.1.18           00-03-ff-36-9b-48     dynamic
  10.253.1.25           00-11-43-de-91-15     dynamic
  10.253.1.26           00-11-43-e7-97-fc     dynamic
  10.253.1.35           00-14-22-17-c8-91     dynamic
  10.253.100.1          00-15-2b-46-50-00     dynamic
  10.253.100.2          00-09-0f-83-3b-8a     dynamic

Z:\>
```

If a router is communicating with a device, it will have its own ARP cache. Here is an example of the show arp command on a Cisco IOS router:



```
Router#sh arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  1.1.1.1                 -   0003.e39b.9220  ARPA   Ethernet0/0
Internet  1.1.1.2                 0   Incomplete      ARPA
Router#
```

In this example, you can see how IP address 1.1.1.1 is mapped to Ethernet MAC address 0003.e39b.9220. Notice the Incomplete entry; this is a sign of trouble. Switches will have their own ARP cache but they will also keep track of which MAC address is connected to which port on the switch. This can be seen with the show mac-address-table commands on a Cisco IOS switch:



```
Switch60#show mac-address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
 All    0014.1c40.b080    STATIC      CPU
 All    0100.0ccc.cccc    STATIC      CPU
 All    0100.0ccc.cccd    STATIC      CPU
 All    0100.0cdd.dddd    STATIC      CPU
   1    0000.aa67.64c5    DYNAMIC     Fa0/14
   1    0000.aa70.d9b9    DYNAMIC     Fa0/7
   1    0001.e641.96cd    DYNAMIC     Fa0/2
   1    0004.00d5.285d    DYNAMIC     Fa0/18
   1    0007.50c4.3440    DYNAMIC     Fa0/2
   1    0008.74a5.9ee0    DYNAMIC     Fa0/2
   1    0009.0f0a.6974    DYNAMIC     Fa0/8
   1    000b.db12.a3f9    DYNAMIC     Fa0/12
```

PROXY ARP

Proxy ARP is the name given when a node responds to an arp request on behalf of another node. This is commonly used to redirect traffic sent to one IP address to another system. Proxy ARP can also be used to subvert traffic away from the intended recipient. By responding instead of the intended recipient, a node can pretend to be a different node in a network, and therefore force traffic directed to the node to be redirected to itself. The node can then view the traffic (*e.g.* before forwarding this to the originally intended node) or could modify the traffic. Improper use of Proxy ARP is therefore a significant security vulnerability and some networks therefore implement systems to detect this. Gratuitous ARP can also help defend the correct IP to MAC bindings.

Modern IP hosts, such as workstations and PCs, transmit directly to either a destination host or router. If the destination is on the same IP network and subnetwork as the sender's, the sender transmits an ARP request to determine the destination MAC address and then transmits directly to it over the LAN.

If the destination's net/subnet is not the same as the sender's, the sender transmits the packet to a router. Hosts are usually configured manually with a default router, which is the IP address of a router on their LAN. Older hosts may always attempt to ARP for a destination address, even if it is not on the local LAN. The older host expects the router to respond to the ARP request with the router's MAC address. This is called

Hosts With No Subnet Support

If the host attempts to send a packet to a network subnet, it sends an ARP request to find the MAC address of the destination host. If the subnet is not on the local wire, a router configured for ARP subnet routing may respond to the ARP request with its own MAC address if the following conditions exist:

- The router has the location of the subnet in its routing table.
- The router sends packets to that subnet via a different interface than the interface that received the ARP request.

Because of the second condition, configure all routers on a local wire for ARP subnet routing when you use hosts without network subnet support.

Proxy ARP Request Example

The following list describes the sequence when a station requiring Proxy ARP wants to send an IP packet to a host on a remote network:

- The host issues an ARP request that contains the destination IP address.
- Any router enabled to respond looks at the IP address for a match in its routing table.
- If there is a match and the route does not pass back through the same LAN port where the ARP host resides, the router responds with an ARP response

supplying its MAC address. Finding a match without passing back through the ARP host port implies another router is present, has a shorter path to the destination, and replies to the ARP itself.

- The host then sends the packet to the router using the newly learned MAC address.
- The host stores this information (that is, the mapping of the IP address to the MAC address) in a local cache so that if it sends another packet to the same destination, it can do so without sending an ARP Request.
- The information is not used. The information is aged out of the cache and may be relearned by resending an ARP Request.

Caution When Using Proxy ARP

The use of proxy ARP is discouraged in modern IP operation. Few hosts require it.

PROXY SUBNET ARP

Proxy Subnet ARP is the same as Proxy ARP except that the router responds to ARP requests for hosts it knows are on other subnets remote from the local subnetwork.

Sometimes hosts forward to a router for destinations with different class A, B, or C addresses, but ARP for any destination with the same class A, B, or C address as their own. They do not know about subnets of the class A, B, or C addresses. They expect the router to respond to the ARP for all subnets of the local class A, B, and C net and to forward to the proper subnet.

Proxy Subnet ARP Example

The following example shows that a host functioning with ARP does not use subnetting (*i.e.*, subnetting is not configured or software does not include subnetting). Unless the router is enabled to respond using Proxy ARP subnet, it does not respond to this ARP and denies connectivity to other subnets of the same IP network.

Example Addressing Description

A single IP class B network number 128.12.0.0 is used to define two subnetworks connected by a router: 128.12.1.0 and 128.12.2.0 (mask 255.255.255.0). The host is on 128.12.1.0 and is attempting to send to 128.12.2.1.

If the host used subnetting, then it sends a packet to its default router and relies on the router to get the packet delivered to the destination 128.12.2.0.

If the host does not use subnetting then it sees the IP network address as 128.12.0.0 (it only knows IP network addresses and therefore uses a class B mask of 255.255.0.0 to obtain 128.12.0.0) and calculates that the destination is on the local LAN (because it has the same network number as itself). It therefore ARPs for the 128.12.2.1 address.

The router must enable Proxy Subnet ARP in order to respond with the router's MAC address. It sends a packet to its default router and relies on the router to get the packet delivered to the destination 128.12.2.0. The host does not use subnetting It sees the IP network address as 128.12.0.0 (it only knows IP network addresses and therefore uses a class B mask of 255.255.0.0 to obtain 128.12.0.0) and calculates that the destination is on the local LAN (because it has the same network number as itself). It therefore ARPs for the 128.12.2.1 address. The router must enable Proxy Subnet ARP in order to respond with the router's MAC address.

Inverse ARP Description

Inverse ARP is a protocol which allows a device to automatically determine the IP Address of a remote device in a Frame Relay network.

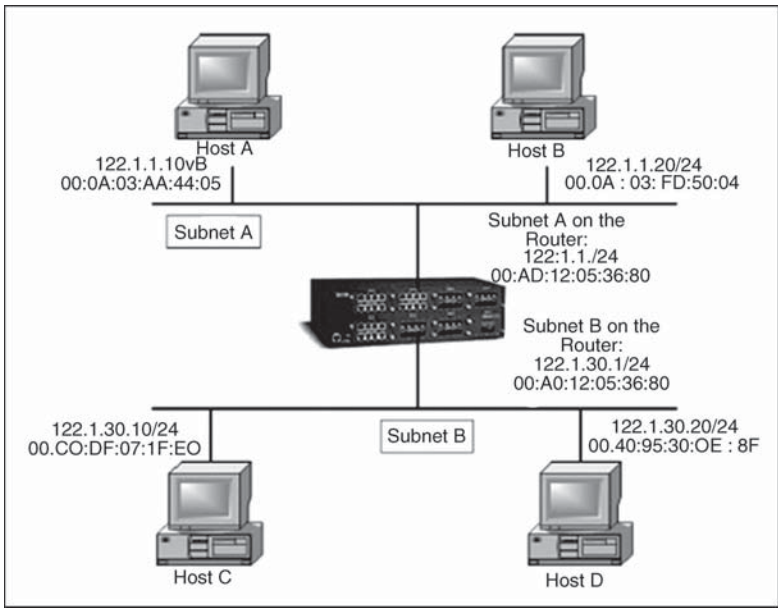
Duplicate IP Address Detection

Duplicate IP Address Detection Defined

Duplicate IP Address Detection is used to detect if the same IP address has been configured on multiple IP devices on the same LAN. If a user configures interface with the same IP address as another device on the same LAN, the network will not work properly. Both devices could receive and respond to packets with that common IP address.

Note: Duplicate IP Address Detection cannot detect all the address duplication problems. There is not a central database to hold all the IP address configurations of a full network. Only unicast addresses are checked.

RARP



RARP (Reverse Address Resolution Protocol) is a protocol by which a physical machine in a local area network can request to learn its IP address from a gateway server's Address Resolution Protocol (ARP) table or cache. A network administrator creates a table in a local area network's gateway router that maps the physical machine (or Media Access Control-MAC address) addresses to corresponding Internet Protocol addresses. When a new machine is set up, its RARP client program requests from the RARP server on the router to be sent its IP address. Assuming that an entry has been set up in the router table, the RARP server will return the IP address to the machine which can store it for future use.

A reverse address resolution protocol (RARP) is used for diskless computers to determine their IP address using the network. The RARP message format is very similar to the ARP format. When the booting computer sends the broadcast ARP request, it places its own hardware address in both the sending and receiving fields in the encapsulated ARP data packet. The RARP server will fill in the correct sending and receiving IP addresses in its response to the message. This way, the booting computer will know its IP address when it gets the message from the RARP server.

RARP request packet is usually generated during the booting sequence of a host. A host must determine its IP address during the booting sequence. The IP address is needed to communicate with other hosts in the network. When a RARP server receives a RARP request packet, it performs the following steps:

- The MAC address in the request packet is looked up in the configuration file and mapped to the corresponding IP address.
- If the mapping is not found, the packet is discarded.
- If the mapping is found, a RARP reply packet is generated with the MAC and IP address. This packet is sent to the host, which originated the RARP request.

When a host receives a RARP reply packet, it gets its IP address from the packet and completes the booting process. This IP address is used for communicating with other hosts, till it is rebooted. The length of a RARP request or a RARP reply packet is 28 bytes.

The ‘operation’ field in the RARP packet is used to differentiate between a RARP request and a RARP reply packet. In an RARP request packet, the source and destination IP address values are undefined. In a RARP reply packet, the source IP address is the IP address of the RARP server responding to the RARP request and the destination IP address is the IP address of the host that sent the said RARP request.

Since a RARP request packet is a broadcast packet, it is received by all the hosts in the network. But only a RARP server processes a RARP request packet, all the other hosts discard the packet. The RARP reply packet is not broadcast, it is sent directly to the host, which sent the RARP request. If more than one RARP server responds to a RARP request, then only the first RARP reply received is used. All other replies are discarded. If a RARP reply is not received within a reasonable amount of time, the host, which sent the RARP request, will not be able to complete its booting sequence. Usually, the host will again retry sending the RARP request after a timeout period.

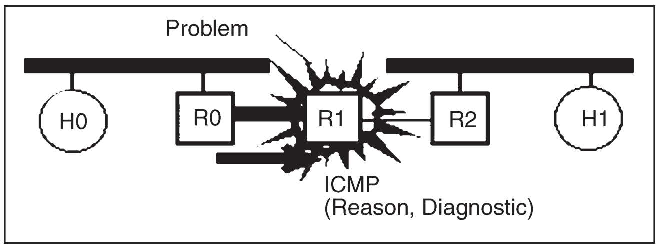
The BOOTP and DHCP protocols can be used instead of RARP to get the IP address from the MAC address.

DIFFERENCE BETWEEN ARP AND RARP

The address resolution protocol (ARP) is used to associate the 32 bit IP address with the 48 bit physical address, used by a host or a router to find the physical address of another host on its network by sending a ARP query packet that includes the IP address of the receiver. The reverse address resolution protocol (RARP) allows a host to discover its Internet address when it knows only its physical address.

INTERNET MESSAGE CONTROL PROTOCOL (ICMP)

The Internet Control Message Protocol (ICMP) [RFC792] protocol is classic example of a client server application. The ICMP server executes on all IP end system computers and all IP intermediate systems (*i.e.* routers). The protocol is used to report problems with delivery of IP datagrams within an IP network. It can be sued to show when a particular End System (ES) is not responding, when an IP network is not reachable, when a node is overloaded, when an error occurs in the IP header information, etc. The protocol is also frequently used by Internet managers to verify correct operations of End Systems (ES) and to check that routers are correctly routing packets to the specified destination address.



ICMP messages generated by router R1, in response to message sent by H0 to H1 and forwarded by R0. This message could, for instance be generated if the MTU of the link between R0 and R1 was smaller than size of the IP packet, and the packet had the Don't Fragment (DF) bit set in the IP packet header. The ICMP message is returned to H0, since this is the source address specified in the IP packet that suffered the problem. A modern version of Path MTU Discovery provides a mechanism to verify the Path MTU [RFC4821].

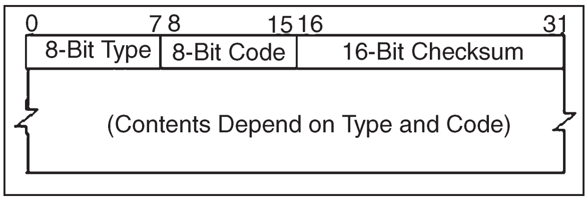


Figure: An ICMP Message Consisting of 4 Bytes of PCI and an Optional Message Payload.

The format of an ICMP message. The 8-bit type code identifies the types of message. This is followed by at least the first 28 bytes of the packet that resulted in generation of the error message (*i.e.* the network-layer header and first 8 bytes of transport header). This payload is, for instance used by a sender that receives the ICMP message to perform Path MTU Discovery so that it may determine IP destination address of the packet that resulted in the error. Longer payloads are also encouraged (which can help better identify the reason why the ICMP message was generated and which program generated the original packet).

The encapsulation of ICMP over an Ethernet LAN using an IP network layer header, and a MAC link layer header and trailer containing the 32-bit checksum:

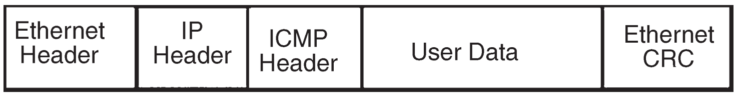


Figure: Encapsulation for a Complete ICMP Packet.

It is the responsibility of the network layer (IP) protocol to ensure that the ICMP message is sent to the correct destination. This is achieved by setting the destination address of the IP packet carrying the ICMP message. The source address is set to the address of the computer that generated the IP packet (carried in the IP source address field) and the IP protocol type is set to "ICMP" to indicate that the packet is to be handled by the remote end system's ICMP client interface.

RFC792 specifies the Internet Control Message Protocol (ICMP) that is used with the Internet Protocol version 4 (IPv4). It defines, among other things, a number of error messages that can be used by an end-system and intermediate systems to report errors back to the sending system. The host requirements [RFC1122] classifies ICMP these error messages into those that indicate "soft errors" (advising of problems), and those that indicate "hard errors" (which need to be responded to).

A version of ICMP has also been defined for IPv6, called ICMPv6 [RFC4443]. This subsumes all the equivalent functions of ICMP for IPv4 and adds other network-layer functions. ICMP error messages are up to 1280 bytes in size, and therefore always carry a substantial number of bytes from the packet that generated the error being reported.

THE PING APPLICATION

The "ping" program contains a client interface to ICMP. It may be used by a user to verify an end-to-end Internet Path is operational. The ping program also collects performance statistics (*i.e.* the measured round trip time and the number of times the remote server fails to reply. Each time an ICMP echo reply message is received, the ping program displays a single line of text. The text printed by ping shows the received sequence number, and the measured round trip time (in milliseconds). Each ICMP Echo message contains a sequence number (starting at 0) that is incremented after each transmission, and a timestamp value indicating the transmission time.

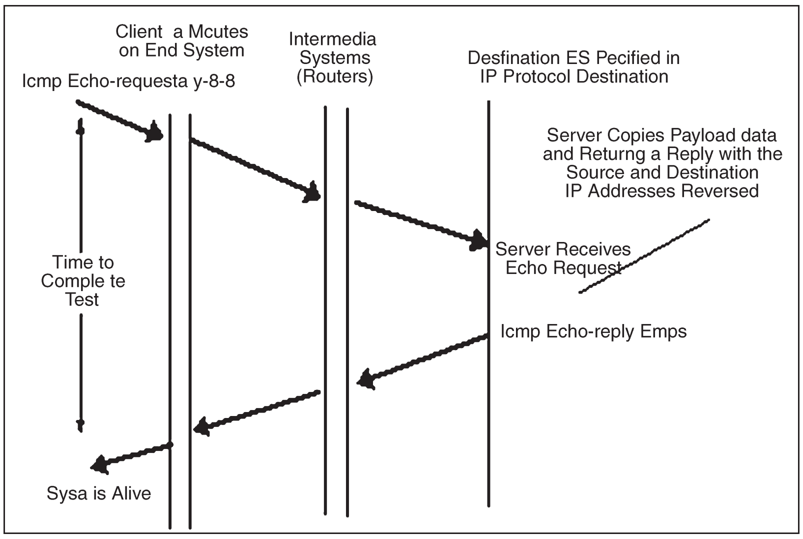

Client a Mcutes on End System
Intermedia Systems (Routers)
Desfination ES Pecified in IP Protocol Destination
Icmp Echo-requesta y-8-8
Server Copies Payload data and Returng a Reply with the Source and Destination IP Addresses Reversed
Server Receives Echo Request
Time to Comple te Test
Icmp Echo-reply Emps
Sysa is Alive

Figure: Use of the Ping Program to Test Whether a Particular Computer ("Sysa") is Operational.

The operation of ICMP is illustrated in the frame transition. In this case there is only one Intermediate System (IS) (*i.e.* IP router). In this case two types of message are involved the ECHO request (sent by the client) and the ECHO reply (the response by the server). Each message may contain some optional data. When data are sent by a server, the server returns the data in the reply which is generated. ICMP packets are encapsulated in IP for transmission across an internet.

THE TRACEROUTE APPLICATION

The "traceroute" program contains a client interface to ICMP. Like the "ping" program, it may be used by a user to verify an end-to-end Internet Path is operational, but also provides information on each of the Intermediate Systems (*i.e.* IP routers) to be found along the IP Path from the sender to the receiver. Traceroute uses ICMP echo messages. These are addressed to the target IP address. The sender manipulates the TTL (hop count) value at the IP layer to force each hop in turn to return an error message.

- The program starts by sending an ICMP Echo request message with an IP destination address of the system to be tested and with a Time To Live (TTL) value set to 1. The first router that receives this packet decrements the TTL and discards the message, since this now has a value of zero. Before it deletes the message, the system constructs an ICMP error message (with an ICMP message type of "TTL exceeded") and returns this back to the sender.

Receipt of this message allows the sender to identify which system is one link away along the path to the specified destination.

- The sender repeats this two more times, each time reporting the system that received the packet. If all packets travel along the same path, each ICMP error message will be received from the same system. Where two or more alternate paths are being used, the results may vary.
- If the system that responded was not the intended destination, the sender repeats the process by sending a set of three identical messages, but using a TTL value that is one larger than the previous attempt. The first system forwards the packet (decrementing the TTL value in the IP header), but a subsequent system that reduces the TTL value to zero, generates an ICMP error message with its own source address. In this way, the sender learns the identity of another system along the IP path to the destination.
- This process repeats until the sender receives a response from the intended destination (or the maximum TTL value is reached).

Note: Some Routers are configured to discard ICMP messages, while others process them but do not return ICMP Error Messages. Such routers hide the "topology" of the network, but also can impact correct operation of protocols.

Some routers will process the ICMP Messages, providing that they do not impose a significant load on the routers, such routers do not always respond to ICMP messages. When "traceroute" encounters a router that does not respond, it prints a "*" character.

An example:

```
>traceroute bbc.co.uk traceroute to bbc.co.uk
(212.58.224.131), 64 hops max, 40 byte packets
1 10.10.10.1 (10.10.10.1) 51.940 ms 18.491 ms 1.260 ms
2 lo0-plusnet.ptn-ag2.plus.net (195.166.128.53) 49.263
ms 55.061 ms 53.525 ms
3 ge1-0-0-204.ptn-gw2.plus.net (84.92.3.106) 139.647 ms
52.525 ms 127.196 ms
4 gi1-1-22.ptn-gw5.plus.net (212.159.4.6) 76.505 ms
57.524 ms 52.404 ms
5 rt0.thdo.bbc.co.uk (212.58.239.25) 89.200 ms 49.666
ms 144.629 ms
6 212.58.238.133 (212.58.238.133) 48.786 ms 68.650 ms
51.599 ms
```

ICMP Type Numbers

The Internet Protocol [IP] is not designed to be absolutely reliable. The purpose of these control messages [ICMP] is to provide feedback about problems in the communication environment, not to make IP reliable. There are still no guarantees that a datagram will be delivered or a control message will be returned. Some datagrams may still be undelivered without any report of their loss. The higher level protocols that use IP (the transport layer, if TCP, or the application, if UDP) must implement their own reliability procedures if reliable communication is required.

The ICMP messages typically report errors in the processing of datagrams. To avoid the infinite regress of messages about messages etc., no ICMP messages are sent about ICMP messages. Also ICMP messages are only sent about errors in handling fragment zero of fragemented datagrams. (Fragment zero has the fragment offeset equal zero).

The Internet Control Message Protocol (ICMP) has many messages that are identified by a "type" field, these are defined by RFCs.

Many of the types of ICMP message are now obsolete and are no longer seen in the Internet. Some important ones which are widely used include:

Echo Reply (0), Echo Request (8), Redirect (5), Destination Unreachable (3), Traceroute (30), Time Exceeded (11).

A list is shown below (the full list may be retrieved form the link at the base of this page):

Type	*Name*	*Reference*
0	Echo Reply	[RFC792]
1	Unassigned	[JBP]
2	Unassigned	[JBP]
3	Destination Unreachable	[RFC792]
4	Source Quench	[RFC792]
5	Redirect	[RFC792]
6	Alternate Host Address	[JBP]
7	Unassigned	[JBP]
8	Echo	[RFC792]

9	Router Advertisement	[RFC1256]
10	Router Selection	[RFC1256]
11	Time Exceeded	[RFC792]
12	Parameter Problem	[RFC792]
13	Timestamp	[RFC792]
14	Timestamp Reply	[RFC792]
15	Information Request	[RFC792]
16	Information Reply	[RFC792]
17	Address Mask Request	[RFC950]
18	Address Mask Reply	[RFC950]
19	Reserved (for Security)	[Solo]
20-29	Reserved (for Robustness Experiment)	[ZSu]
30	Traceroute	[RFC1393]
31	Datagram Conversion Error	[RFC1475]
32	Mobile Host Redirect	[David Johnson]
33	IPv6 Where-Are-You	[Bill Simpson]
34	IPv6 I-Am-Here	[Bill Simpson]
35	Mobile Registration Request	[Bill Simpson]
36	Mobile Registration Reply	[Bill Simpson]
37	Domain Name Request	[RFC1788]
38	Domain Name Reply	[RFC1788]
39	SKIP	[Markson]
40	Photuris	[RFC2521]

PATH MTU DISCOVERY (PMTUD)

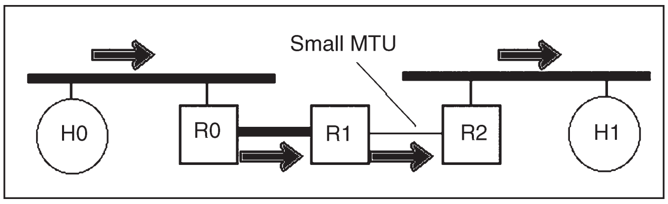
The IP MTU is the largest size of IP datagram which may be transferred using a specific data link connection The MTU value is a design parameter of a LAN and is a mutually agreed value (*i.e.* both ends of a link agree to use the same specific value) for most wide area network links. The size of MTU may vary greatly between different links. Note, people who design lower-layer networks (below IP), often define the MTU in a difference way to the IP-oriented people. This can catch you out, if you are not careful.

IP fragmentation could be used to break larger packets into a sequence of smaller packets for transmission across the network. This however, lowers the efficiency and reliability of Internet communication. The loss of a single fragment results in the loss of an entire fragmented packet, because even if all other fragments are received correctly, the original packet cannot be reassembled and delivered and has to be resent. This is one example were router IP fragmentation can be considered harmful.

In addition, some Network Address Translators (NATs) and firewalls drop IP fragments. Often network address translation performed by a NATs only operates on complete IP packets Some firewall policies also require inspection of a complete IP packet.

Even with these being the case, some NATs and firewalls simply do not implement the necessary reassembly functionality, and instead choose to drop all fragments. These fundamental issues with fragmentation exist for both IPv4 and IPv6.

Instead of making routers fragment packets, an end system could try to find out the largest IP packet that may be sent to a specific destination. The Path MTU Discovery algorithm operates at the sender at the boundary of the Transport Layer and Network Layers, generating probe messages and responding to ICMP error reports that indicate a low MTU. In Intermediate Systems (IS), *i.e.* routers, this operates in the Network Layer, returning ICMP error messages based on the link-layer configuration.

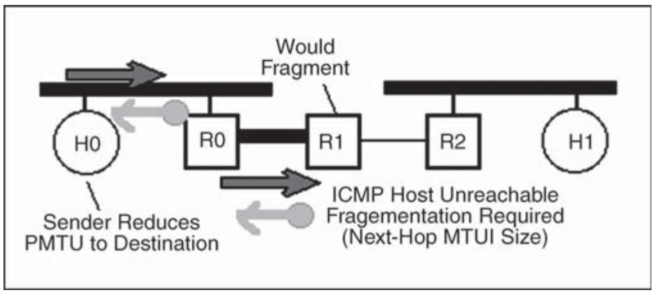


The way in which an end system finds out this large packet size associated with a specific path (*i.e.* on the series of links used to reaxh a destination), is to send a large IP packet (up to the MTU of the link to which it is connected). The packet is sent with the Don't Fragment (DF) flag set in the IP protocol header.

ICMP-BASED PMTUD

The simplest form of PMTUD relies on the ICMP server that executes on all IP end system computers and all IP intermediate systems (*i.e.* routers). If a router receives a packet that is less than or equal to the MTU of the link that it wishes to forward the apcket to, then it sends the packet. If however, a router finds that the MTU of the next link exceeds the packet size and the DF flag is set, this tells the router not to segment the packet. Instead, the router will discard the packet. An Internet Control Message Protocol (ICMP) message is returned by the router back to the sender (H0), with a code saying the packet has been discarded, but importantly, this message also says the reason (*i.e.* the fragmentation would have been required) and indicates the maximum MTU allowed (in this case the MTU of the link between R1 and R2).

Occasionally the end system will generate a large packet, just to see if a new Internet path has been found (*i.e.* a different route). The new path may allow a larger P-MTU.


Would Fragment
H0
R0
R1
R2
H1
Sender Reduces PMTU to Destination
ICMP Host Unreachable Fragementation Required (Next-Hop MTUI Size)

If an IPv4 end system receives an ICMP message saying a packet is too large (type = 3, code = 4), it sets a variable called the Path-MTU (P-MTU) to the appropriate maximum size and then itself fragments the packet to make sure it will not be discarded next time. The end system keeps (caches) a set of P-MTU values for each IP address in use.

When there are a series of links along the path, each with increasingly smaller MTUs, the above process may take place a number of times, before the sender finally determines the minimum value of the P-MTU. Once the P-MTU has been found, all packets are sent segmented to this new value.

In this model, Routers do not therefore have to do any additional processing for these packets. This is much more efficient than router fragmentation. However, practical problems are being experienced within the current Internet, caused by systems (*e.g.* some firewalls) that do not return the required ICMP messages back to the sender. The widely deployed version of PMTUD relies on messages received from the ICMP protocol, specified by RFC1191.

PMTUD WITHOUT RELYING ON ICMP

There are issues with deploying PMTUD in practical networks, you may need to think harder, and the best method is to avoid having to receive ICMP messages (RFC2923). One standard method is to sends probes and validate which get through at the transport layer- this is known as Packet-Layer PMTUD (PLPMTUD) and is standardised in RFC4821. PLPMTUD can of course also utilise ICMP messages (if it wishes) as a part of the process of learning what size of MTU will work across an entire Internet path.

PMTUD with IPv6

RFC4443, which defines ICMPv6 specifies a "Packet Too Big" (type 2, code 0) error message, that is analogous to the ICMP "fragmentation needed and DF bit set" (type 3, code 4) error message of ICMP for IPv4. RFC1981 defines the Path MTU Discovery mechanism for IP Version 6, that makes use of these messages to determine the MTU of an arbitrary internet path. PLPMTUD also works with IPv6.