CHAPTER 1

PRINCIPLES OF NETWORK MONITORING

Jawwad Shamsi and Monica Brocmeyer

Department of Computer Science, Wayne State University
5143 Cass Avenue, 431 State Hall, Detroit, MI 48202, USA
E-mail:{ jshamsi, mbrockmeyer }@wayne.edu

Efficient and cost-effective measurement of network characteristics is pivotal for distributed systems deployed on the Internet. The network characteristics are utilized by Internet-based distributed systems to provide better service to the user and enhance performance for the application.

This chapter provides a detailed analysis of existing techniques for the measurement of the four important network characteristics which include latency, bandwidth, path detection and loss rate. The chapter describes key concepts related to network measurements, including techniques for clock synchronization, strategies for time stamping of probes, methods for network analysis, difference between active and passive measurements and comparison of round trip vs. one way delay measurements. It elaborates the usefulness of different transport and network layer protocols (i.e. TCP, UDP and ICMP) for obtaining network measurements and continue this discussion to describe some important measurement tools such as Ping, Traceroue, Pathchar, Sting, Scriptroute, Spruce and Paris Traceroute. The chapter explains the effectiveness and limitations of these tools with respect to the measurement of network characteristics.

1. Introduction

The effectiveness of a distributed application providing service to its end users depends largely on the characteristics of the underlying network. A web server is of no use if its immediate router fails and all the paths leading to the web server becomes unavailable. Similarly, a VoIP (Voice over IP) service has restricted functionality if the available bandwidth is limited.

The two examples mentioned above are not infrequent. Internet based systems observes enough uncertainty and unpredictability that effect the quality and

effectiveness of distributed systems. This uncertainty strive the need for monitoring of various network characteristics, both at the level of the service provider and the application. An ISP (Internet Service Provider) may monitor the bandwidth used by its customers, in order to allow a fair share of the bandwidth usage, whereas an application could periodically measure latency to its servers in order to detect congestion and failure.

1.1 Which network characteristics are relevant for measurements?

The significance and relevance of network characteristics varies with applications. A content distribution network may be interested in measuring latency from its clients to the application server, whereas a load balancing application may find available bandwidth as a useful criterion to adjust the load on the servers. Other network characteristics such as congestion, queuing delay, network failure, topology discovery, and bottleneck determination also have great significance. However, there are four network characteristics that constitute the basic paradigm of network measurements and could be utilized in the computation of other related network characteristics. They are mentioned below

Latency: Latency refers to the time taken by the message to traverse from the sending host to the destination host. It is usually measured in milliseconds (ms). Latency of a path has a great significance for many applications. For many applications latency is used to determine the availability of the servers and select nearest available server in terms of latency, specifically in the content distribution networks. Further, variation in latency (i.e. the deviation between the successive latency measurements) is used to determine the quality of the path and detect network congestion.

Packet Loss: If the rate of packets sent from the source host is not equal to the rate of packets received at the destination host then the path experiences packet loss. Loss rate of a path is the rate (in percentage) at which packets are being lost while traversing through the network path. Lost packets affect the performance of the application as they are often required to be retransmitted. Even when the packets cannot be retransmitted (for example, such as the live transmission of audio or video streams) a high packet loss could lead to low application performance. Due to these reasons it is always desirable to select paths with low loss rate.

Path Detection: When a message is sent across the Internet, it traverses through various intermediate routers to reach the destination. Path detection is the process of determining the actual path taken by the message in reaching from source to destination. Since there are many possibilities for a path from one host to another, path detection enables an application to determine the actual path taken by the message during transmission. Path changes are possible on the Internet, therefore path detection also facilitate in determining a change in the path between two nodes. It can also be used as a sanity check to determine or isolate network failure.

Bandwidth: Bandwidth refers to the capacity of the path to transfer data in a unit time. It is measured in bits per second (bps). Bandwidth has great implication for multimedia applications. Such applications generally have high bandwidth requirements. If the available path has limited bandwidth than the multimedia application suffers degraded performance. Bandwidth detection tools can be used to infer the path quality and select appropriate path.

The purpose of the chapter is to develop reader skills and proficiency related to network measurements. The chapter elucidates basic concepts about network monitoring and explains several approaches and tools for the measurement of the four basic network characteristics. The chapter also discusses practical issues related to development of network monitoring tools. The chapter assumes that the reader has the understanding of basic networking terms such as the seven layer OSI model and familiarity with the transport layer protocols such as TCP and UDP.

2. Background Information

Network monitoring is a very vast and rigorous topic. In order to thoroughly explain the concepts, this section mentions some background information. The information includes description about different types of network monitoring and explanation of underlying protocols used for monitoring. Readers may also refer to Section 9 of this chapter that explains some key terms associated with network monitoring.

2.1. Types of Network Monitoring

There are two basic approaches for network monitoring, active monitoring and passive monitoring.

Active Monitoring

Active monitoring involves injecting probes in the network, specifically for the purpose of monitoring. Active monitoring bears the cost of sending additional traffic in the network; however, under most scenarios, the packet size is relatively small compared to the actual capacity of the network, therefore the cost of injecting additional traffic is minimal. The cost of injecting traffic can be reduced by decreasing the probing rate; however, this may reduce the quality of the measured characteristics. Active monitoring provides full control with respect to monitoring interval, packet size and the path to be monitored. Further, the data obtained is not specific to any particular application.

Passive Monitoring

Passive monitoring is the process of observing the existing network traffic and collecting information from it without injecting additional monitoring probes in the network. In a passive measurements based system, network measurement information is retrieved through the packets which are sent as a part of another application. Packets are captured by monitoring application, which is deployed either on the source and the destination hosts or along the path through a tap[a]. The passive mode of monitoring avoids the overhead of introducing measurement traffic in the network and also evades the use of stale data for measurements. However, in passive mode the monitoring application has little or no control over probing interval, packet size or the path to be monitored. Further, due to increase in the capacity of the core links it is expensive to identify and measure the packet of a particular flow. Passive monitoring could also raise privacy concerns.

The preference for the type of monitoring varies with the application. For instance, an application measuring the performance of an existing application such as an intrusion detection system might prefer a passive approach, where as an application employing performance enhancement measures such as load balancing may desire an active scheme. An application may also employ a combination of the passive and active schemes for higher efficiency. The Wren grid monitoring system utilizes such a scheme, in which the passive mode is used

[a] A tap is a hardware device which provides a way to capture data along a path. A tap usually has three connection points. Two connection points are connected to each end of the path, where as the third point is connected to a monitor which listens for the information. Tap is not a sink, i.e., it copies all the information to the monitor and the communication between the two ends of the path remains uninterrupted.

when an application is sending messages, whereas the active mode is used when the application is silent [18].

2.2. Network protocol

For passive monitoring, the measurement system has no choice for the underlying protocol of network measurement, i.e. the monitoring application monitors packets that are sent as a part of another application. However, in the active mode, the measurement system has to decide about the underlying protocol used for sending probes. Many choices are available and they are discussed below.

ICMP

ICMP or Internet Control Message Protocol is the underlying protocol for the famous ping utility. It operates at the networking layer. The principle advantage of the ICMP based measurement tools is simplicity and ease of use. ICMP-based schemes do not require connection setup or handshake. Further, they do not require any specific implementation at the recipient end, i.e. only the host sending the active probes implements mechanism for sending the ICMP probes. Since ICMP messages are control messages, an ICMP request is automatically replied by the destination host. However, the automatic response capability of the host could be exploited by attackers and is seen as security vulnerability. ICMP will be discussed further when we analyze the ping utility.

TCP

TCP (Transmission Control Protocol) is the underlying protocol for many internet applications including WWW and FTP. It operates at the transport layer. Unlike ICMP based tools, TCP-based utilities bear the overhead of the TCP handshake[b]. TCP-based tools also require that the destination host runs a service on a designated port. The sending host establishes TCP connection to the destination through the TCP handshake and sends probes to the destination host at the server port. The server at the destination replies with a response that is used for network measurement.

[b] TCP handshake is a three-way process. In the first step, the host initiating the connection sends a TCP SYN packet. The recipient host responds with an ACK packet. Finally, the original host sends a SYN ACK packet. All the three packets contain sequence numbers for proper identification.

An important limitation of the TCP based tools is that since TCP is not a packet based protocol, it makes its own decisions about sending packets. By using various optimizations, including Nagle algorithm [34], TCP coalesces packets of small sizes. Since TCP has a large overhead including a 40 byte header (20 byte for IP and 20 for TCP), these optimizations allow TCP to conserve bandwidth. At the user level, it is difficult to force TCP to send packets at a specific interval [44].

In some cases, the requirement of a running service at the destination can be eliminated using TCP ACK and RST. According to RFC 793 [33], if a sender sends a TCP ACK packet to a closed[c] TCP port, then the TCP at the destination will reply with the TCP RST packet and the same sequence number as of the TCP ACK packet in the request. In such a case, the TCP RST can be used as a response. However, this type of request-response method can be used only if the response from the destination host is not specific to the request, i.e. the response is not generated based on the contents in the request.

UDP

UDP (User Datagram Protocol) is a datagram-based protocol that operates at the transport layer. Unlike TCP, UDP does not require a handshake. Additionally, UDP has an 8 byte header (plus 20 byte IP header) that results in lower overhead than TCP. Since UDP is a packet protocol it allows sending probe packets at controlled intervals. Moreover, UDP is a connectionless protocol in which a lost request (or a response) can be effectively used to detect packet loss.

Like TCP, UDP can also eliminate the need to run a service at the destination port. If a UDP request is received at a port at which the service is not running, then it responds by sending *ICMP destination unreachable* message to the sender.

Many other choices such as DCCP (Datagram Congestion Control Protocol) and SCTP (Stream Control Transmission Protocol) also exist. The selection of underlying protocol for network monitoring is based on the requirements and restrictions from an application.

3. Network Characteristics

Having identified the basic principles of monitoring, this section describes various approaches for the measurement of network characteristics. Each of the

[c] A port that is not running any TCP service

four network characteristics, latency, loss-rate, path detection and bandwidth, are discussed in a subsection which explains the basic concepts about the characteristic and describes various approaches for its measurements.

3.1. Latency

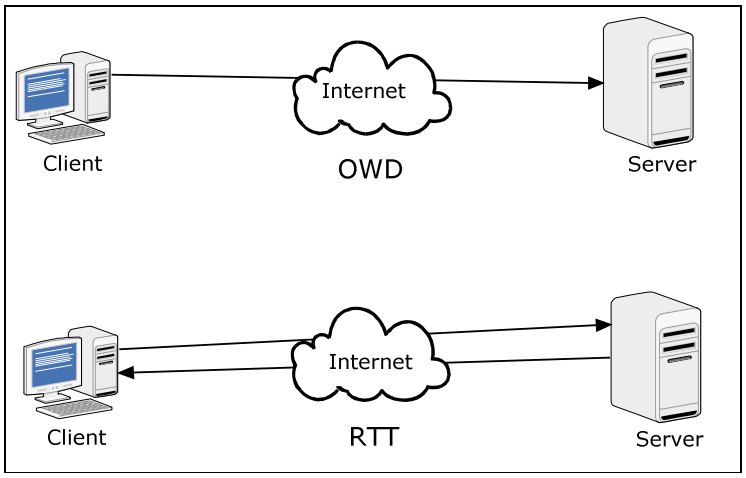
Latency has been the most widely measured network characteristic. It refers to the time (in ms) taken by the packet to traverse from the sending host to the destination host. Latency between two hosts could vary due to two reasons: First, variation is possible due to changes in queuing delay or congestion and second due to route changes. When a packet is sent from sender to receiver, it traverses through multiple intermediate routers, each of which has variable queuing delay. Therefore, two successive packets could follow different routes, resulting in differing latency for each. . Route changes are quite common due to congestion or network failure. Therefore, the variation in latency can be utilized to depict congestion between the two end points, where paths with low congestion observes more consistency in latency.

The notion of latency mentioned here represents the one-way latency from the sender to the receiver (commonly refers to the **One-Way Delay** or OWD). It is measured by subtracting the time that packet is sent by the sending host, from the time that packet is received at the recipient host.

Besides OWD, the word latency is also sometimes used to refer to the **Round Trip Time** (RTT), between the two hosts. This involves the time taken by the packet to reach the recipient host from the sending host and the reply from the recipient host to the sending host. Figure 1 depicts the difference between OWD and RTT.

Fig. 1. OWD vs. RTT.

Example 1: Assume the round trip time between two Internet hosts is 200 ms. What is the one way latency (OWD) between them?
Solution: Unknown. The internet is asymmetric, i.e. the forward and the backward routes between two hosts are not necessarily the same. Therefore OWD is not necessarily equal to one half of RTT.



Example 2: What is the RTT between two internet hosts A and B, if the one way latencies of the forward and backward paths between them are 91 ms and 92 ms? The host B takes 200 usec on average to respond to the probes.
Solution:
RTT = 91+92+0.2= 183.2 ms.

RTT implies the effective communication delay for applications that are request-response applications such as WWW and FTP. In comparison, OWD is useful

communication tool for applications that have one way communication such as multicast.

In case of a large message transfer, the end-to-end latency of a message is higher then the packet latency. Large messages are usually disassembled into small packets and transferred to the destination. At the destination, the message is reassembled into its original form. The processes of disassembling and assembling are called as fragmentation and de-fragmentation, respectively.

The capacity of a path to transfer a message without fragmentation depends upon its MTU or **Maximum Transmission Unit**, where MTU of a path denotes the size of the IP datagram that can be transferred. It usually varies for different networks; however for Ethernet it is equal to 1500 bytes. The end to end path between two internet hosts may compose of links with different MTUs. In such a scenario, intermediate routers may perform the fragmentation according to the capacity of each link. We now describe some important latency measurement techniques.

Measurement Strategies

The measurement strategy for latency is simple. The sending host sends a packet and waits for the response. When the response is received, it subtracts the sent time stamp from the received time stamp and computes the round trip latency. For OWD, however, the destination host notes the time at which the request is received at the destination and includes it in its response to the original sender which computes OWD by subtracting the sending time from the time at which the request was received at the destination. A major requirement for the measurement of OWD is to synchronize clocks of the two nodes. Note that the underlying protocol could be ICMP, TCP or UDP.

The methodology mentioned above is confined to active monitoring. For passive monitoring, the monitoring host can monitor the existing flows form an application and compute RTT. For instance, a web client can compute its RTT with a web server by comparing time stamps of the HTTP request and HTTP response.

3.2. Loss rate

The loss rate of a path is the rate at which packets are being dropped while traversing through the path. There could be multiple reasons for packet loss. Most commonly, for a path with limited bandwidth, the intermediate routers buffer the packets in the queue. However, if a queue at the router approaches its

storage capacity then the router drops the packet. Packet loss could also occur due to hardware failure at the router or at the destination host. It can also happen due to the packet being corrupted because of noise in the path.

The detection of the packet loss is challenging. In theory, packet loss can be detected if a response to a request is not received by the sender within a certain amount of time. However, the sender (the entity sending the request) faces a challenge in determining the duration of the timeout. A response from the destination host could be delayed due to multiple reasons including high RTT, congested path, heavily loaded server or even low connection speed at the sender. If a sender adopts a safe-passage by implementing a large timeout time then it suffers from poor performance, since a lost packet might be detected earlier; on the other hand if the waiting time is small, then the sender might end up causing

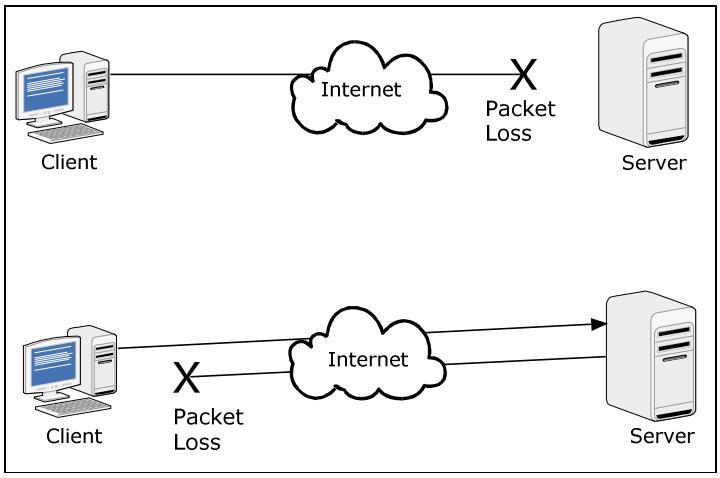


Fig. 2. Examples of Packet Loss.

congestion in the network by sending multiple requests, or might erroneously conclude that a packet is lost when, in fact, the packet is just delayed. While there is no perfect answer to determine the duration of the waiting time, most of distributed applications adopt a safe approach and keeps a longer waiting time. For instance, the time out for a DNS query is 15 seconds.

Measurement Strategies

Packet loss is measured by initiating a timer for each probe and noting the probes for which the response is not received. Note that in such a case, the direction of

packet loss is unknown *i.e.,* if the request from sender is lost or the response from the destination has failed to reach the sender (see Figure 2). This information is important for many applications that use one-way communication such as multicast. In the next section, we will analyze the sting tool [37], which computes loss rate in both directions.

3.3. Path Detection

An internet path consists of various hops connected by routers. Path detection is the process of determining information (including name and IP address) about all intermediate routers traversed when a packet is sent across an Internet path.
Note that path detection is different from the topology discovery. The former is the process of determining only the intermediate routers that are traversed by a packet when it is sent along the path, whereas the later is the process of determining all the combinations of a path between two Internet hosts. Path detection tools can be utilized to discover topology.
Figure 3 illustrates a scenario where several routes exist for a path from client *X* to the server *Y*.

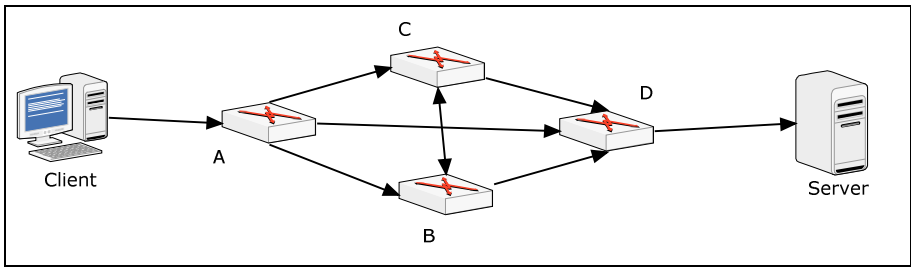


Fig. 3. Possibilities for a path.

The computation of Internet paths is achieved through routing protocols, which are categorized in two broad categories: **Internet Gateway Protocols** and **Exterior Gateway Protocols** such as BGP. The former refers to the routing within an autonomous system[d], whereas the later is used when the routing is done across autonomous systems.

[d] An autonomous system (AS) is a collection of networks and routers that maintain a single routing policy. An AS could be under control of one or more entities and is uniquely identified by an AS number [35].

Measurement Strategies

Path detection strategies are based on using TTL (Time-to-live) values in the IP header. The purpose of the TTL is to avoid infinite looping (IP routing cycle) of IP packets. TTL is an 8 bit field which indicates the number of hops or routers a packet can be traversed, before it must reach its destination or be discarded. When a packet traverses a path, each intermediate router examines the TTL value of the packet. If the TTL value is greater than 1, then the TTL value is decremented by 1 and the packet is sent to the next hop, which could be another router or the final destination host. However, if the TTL valuee is 1, then the router discards the message and sends an ICMP *time exceeded* message to the original sender. The ICMP *time exceeded* message indicates that the TTL field has expired, the packet has failed to reach the destination and it has been discarded. The *ICMP time exceeded* message contains the address of the router that generates the message. Path detection tools work by sending a series of packets with increasing TTL to the final destination. Since the TTL value is gradually increased in the series, each intermediate router once receives a packet having TTL value equal to 1 and sends a reply to the original sender with ICMP *time exceeded* message. The sender can thus obtain information about all the routers in the path.

Example 3: Can path detection tools be used to measure RTT to a specific hop?
Solution: Path detection tools can be used to measure RTT. When the original sender receives an ICMP *time exceeded message* it can compare the send time stamp of the probing packet, with the received time stamp of the ICMP message to compute RTT to a particular hop

3.4. Bandwidth

Bandwidth quantifies the data rate at which a network link or a network path can transfer [31]. It is measured in bps or bits per second. Unlike latency, where the end-to-end characteristic of a path is computed by aggregating the latency of the individual links, the bandwidth of a path is the capacity of the link with minimum bandwidth.

[e] When the destination host receives a packet with a TTL value of 1, it delivers the packet to the application.

The term bandwidth can be used to refer to two different types of speeds. Upload speed denotes the rate at which data is being sent to the destination, while download speed refers to the rate at which data is being received. If a path has similar upload and download speed then it is called as bandwidth-symmetric. Most of the internet backbone is bandwidth symmetric. However, this is not necessarily true for end users, including home users relying on cable or ADSL. The reason for this difference is that most home users are typically interested in downloading, therefore ISPs (Internet Service Providers) allocates more resources for downloading. Note that a bandwidth-symmetric path is not necessarily symmetric in route or vice versa. The two entities are separate and must not be confused with each other.
For the measurement of bandwidth two terms are relevant, capacity and available bandwidth.

Capacity

Capacity refers to the maximum data rate a link can transfer. It is measured as the number of data units (bits) transferred in a unit time (sec). The capacity of a link depends upon the transmission medium and is usually constant at the link-layer (OSI layer2). For instance, for a gigabit Ethernet, the capacity of a link is equal to one gigabit per second. However, this capacity refers to the capability of a link excluding any overhead. Remember, at the IP-layer (layer 3), IP packets are encapsulated in link-layer frames such as Ethernet frames, therefore the actual capacity must also consider the overhead associated in transmitting data in frames. For instance, for Ethernet the overhead consists of 38 bytes in that 14 bytes for Ethernet header, 4 bytes for CRC[f] (cyclic redundancy check) 8, bytes for the frame preamble[g] and the 12 bytes for the inter frame gap[h] [31].
The actual capacity of the path considering the overhead can be expressed as

[f] Cyclic Redundancy Check- Checksum to detect errors in the frame.
[g] A preamble is a 7 byte field, which is a combination of ones and zeros. It is used for synchronization purposes. The preamble field is followed by a 1 byte start of frame delimiter (SFD). The combination of both the preamble and SFD fields takes 8 bytes.

[h] Interframe gap is a delay a network device must wait before it can attempt to transmit a frame. This ensures that a device which has previously transmitted a frame has reverted back to the listening mode. The interframe gap must be 96 bit times. For 10 Mbps, it takes 100 nanoseconds to transmit one bit, therefore 96 bit times is equal to 9.6 micro seconds, which is equal to 12 bytes.

$$C = C_L \frac{L}{O + L} \tag{1}$$

where, C is the actual capacity at the IP level. C_L is the capacity at the link level, O is the overhead needed to encapsulate IP packet and L is length of the Ethernet frame. Therefore the actual capacity of a link varies with the transmission technology. Example 4 further clarifies the concept.

Example 4 Compute the capacity of a path for a 10 base T link with an Ethernet frame size of 1400.
Solution
Given: C_L = 10 Mbps, O=38 bytes and L= 1400 bytes
Placing values in equation (1), we get
C = 9.73 Mbps

The capacity of a path that consists of several links is equivalent to the capacity of the link with minimum capacity. Due to this reason minimum capacity link is also called as the **narrow link**.

Available Bandwidth

The available bandwidth is the unused capacity of the link. At any moment a link is either transmitting at full speed or is idle [31]. Therefore, the utilization (μ) of a link is expressed as an average which is computed over a series of time intervals. Therefore the available bandwidth (A) is equal to the total capacity minus average usage, where usage is expressed as a fraction of the total capacity.

$$A = C(1 - \mu) \tag{2}$$

The link with the minimum available bandwidth is known as the **tight link**. The available bandwidth of a path equals to the available bandwidth of the tight link.

Measurement Strategies

The available bandwidth of a path can be computed by measuring the amount of data transferred over a unit time. Strauss [41] has categorized the existing bandwidth measurement tools into two models.

Probe Gap Model:

The Probe Gap Model (PGM) can be explained as follows. The sender sends two probes that are separated by a time interval of t_1. The two probes arrive at the receiver separated by an interval of t_2. If $t_2 > t_1$, it implies that the probes have experienced a delay during their traversal. Assuming a single bottleneck that induces delay, $t_2 - t_1$ is the amount of delay experienced at the bottleneck. If the capacity (total bandwidth) of the bottleneck is known to be C, then the utilization at the bottleneck is

$$\mu = C(\frac{t_2 - t_1}{t_1}) \tag{3}$$

Since the model assumes a single bottleneck, the utilization of the path is also equal to the utilization at the bottleneck. Therefore, the available bandwidth of the path is

$$A = C(1 - \frac{t_2 - t_1}{t_1}) \tag{4}$$

A major limitation of the PGM is that it assumes that there is a single bottleneck with a known capacity.

Probe Rate Model:

The Probe Rate Model (PRM) is based on idea of creating congestion in the network. A series of packets are sent along the path. If the sending rate of packets at the sender matches with the arrival rate of packets at the receiver, then it implies no congestion. Therefore, the sending rate is increased to the point where the arrival rate becomes lower than the sending rate. At this point congestion is detected in the network, and the instance is referred as turning point. The sending rate just before the beginning of congestion denotes the available bandwidth.
Note that PRM has a larger overhead as compared to the PGM as it requires induction of congestion in the path.

4. Measurements Tools

This section describes some important measurement tools. The section is divided in to five subsections. The first four subsections are dedicated to each specific

network characteristic, whereas as the last subsection describes some tools that have multiple usages.

4.1. Latency Measurements

Ping

Ping is a widely used latency measurement utility. It is built in to most operating systems, including UNIX and Windows. Ping has multiple usages: It can be used to measure RTT between two hosts or to detect packet loss along a path. It works by sending ICMP *echo request* packets to the destination host and listening to the ICMP *echo reply* messages. The ICMP *echo request* is a type of message which must be replied by every host or a router via the ICMP *echo reply* message and the data of the *echo request* message must be unaltered in the *echo reply* message.
The sending host sends a series of ping messages that are identified using probe sequence numbers and the corresponding *echo reply* messages are used to compare the time stamps of the probes and compute RTT. Ping infers a statistical summary of the path latency characteristics including minimum, maximum and average RTT and the packet loss to the destination host. Ping is inexpensive and flexible measurement utility since it does not require the destination host to run any service or listen to any particular port.

Fping

Fping [9] is an enhancement to the ping utility. Ping is limited as it can only send messages to a single destination host at a time. Fping improves upon this restriction by sending probes to multiple hosts. The hosts can be specified on the command line or through a file. However, the underlying protocol of fping is still ICMP.
Both Ping and Fping may demonstrate limited accuracy as many routers or firewalls filter or rate-limit ICMP packets due to security concerns [37]. Recall that we mentioned that every host that receives *echo request* sends an *echo reply*. If a firewall filters ping packets then it will drop the packet before it reaches its intended destination. Since the packet will be dropped, the source will not receive *echo request* message and the *echo reply* message will not be sent. In such a case, the usability of ping remains limited.
Several additional tools which measure latency, as well as other network characteristics are described in Section 4.5.

4.2. Loss rate

Latency measurement techniques utilizing active monitoring approach can be utilized to estimate loss rate along the path. Packet loss can be detected by using a timer. Usually, a timer is set whenever a probe request is sent, and is cancelled when the probe response is received. In this manner, a signal is raised whenever a response is not received, which indicates a lost packet. We now describe how the sting tool can be utilized to estimate loss rate in both directions.

Sting

Sting [37] is a TCP based tool which measures packet loss between two Internet hosts. It can precisely detect the direction of the packet loss i.e. if the loss has occurred in the forward direction (from source to target) or in the reverse direction from (target to source). Sting requires the target host to run a TCP-based service such as a web or telnet server. The Sting protocol is divided into two phases.

In the first phase, the source host sends a series of TCP data packets to the service running on the target host. Each packet has a sequence number[i] one greater than the sequence number of the preceding packet in the series. The target host responds with an acknowledgement for each received packet. In the second phase, the source host initially sends a data packet with a sequence number one greater than the sequence number of the last data packet sent in the first phase. The target host responds by sending a probe with an appropriate acknowledgement number such that if packet loss occurs during the first phase then the acknowledgement number indicates the sequence number of the first lost packet, otherwise the acknowledgement number indicates the next expected packet. In case of packet loss during the first phase, the source responds by retransmitting the lost packet. The process continues until all the lost packets are retransmitted. At the end of the second phase, the loss rate can be computed using the total number of packets sent and the number of retransmitted packets.

---

[i] Recall that in TCP each packet has a 32 bit sequence number and a 32 bit acknowledgement number. The sequence number uniquely identifies each packet and is used to arrange out of order arrival of packets at the target. The acknowledgment number is used by the target to acknowledge the receipt of the data packet. When a target host receives a packet, it sends and acknowledgement back to the sender with the acknowledgement number equal to the expected sequence number of the next packet. The acknowledgement number also implies that all the packets with the preceding sequence number have been received by the target. The acknowledgement number is not incremented if an out of order message is received. For efficiency, many implementations of TCP delay sending the acknowledgement until they received a next packet or a time out occurs.

The procedure described above is useful for computing packet loss in the forward direction. For the packet loss in the reverse direction, the sender counts the number of acknowledgements received from the target. In order to ensure that target sends acknowledgement for each packet received from the sender and does not delay sending the acknowledgement, the sender sends the packet with some delay.

Example 5: Consider sting is used to detect packet loss along a path. If the source sends 100 packets during the first phase and three packets during the second phase then compute the packet loss in the forward direction.
Solution:
In the first phase, 100 packets are sent.
In the second phase, the source sends 3 packets. Therefore, two of the three packets must be the retransmitted packets. Therefore 100-2 = 98 packets are received at the target.
Therefore the forward loss rate is 2/100 =2 %.

4.3. Path Detection

Traceroute

Traceroute (tracert in Windows) is a path tracing utility that is used for path detection or network failure diagnosis. Traceroute works by sending a series of IP packets with incremental values of the TTL. At each step, three packets with the same TTL value are sent. For instance, the first three packets have a TTL value of one while the next three packets contain a TTL value of two and so on. When the first series of packets (with a TTL of 1) arrives at the first hop, the router discards the packet and sends ICMP *time exceeded* message with its address as the source address to the original sender. In this manner, a series of traceroute messages are sent in which a TTL value is gradually incremented, the sending host then collects the information about each router along the path and constructs the network path. The traceroute utility prints the information about each hop-router on the screen. During this process, if a message is lost, or if a router is configured not to reply with an ICMP *time exceeded* message then an asterisk (*) is printed.
The choice of the underlying protocol for the traceroute is platform dependent. The original specification proposed by Jacobson [14] uses UDP packets, with

UNIX utilizing the same choice. The UDP packets are destined for ports from 33434 to 33534. The ports are used by Traceroute, since they are unlikely to be selected by an application. Thus, when the destination receives a probe at the port at which no service is running it responds with an ICMP *destination unreachable* message. The Windows implementation of traceroute utilizes ICMP *echo request* (similar to ping). Note that this does not alter the reply from the router when TTL reaches 1, as in both the cases (UNIX or Windows), the router will send the ICMP *time exceeded* message to the original sender.
Due to security issues, some networks block incoming traceroute packets. In such a scenario, traceroute can only display the router hops up to the firewall router of the network.

TCPtraceroute

TCPtraceroute is a TCP-based implementation of traceroute. The motivation behind using TCP is to bypass firewalls that block UDP and ICMP, as most firewalls permit TCP-based inbound traffic. Unlike conventional traceroute schemes which send UDP or ICMP echo messages and wait for ICMP time exceeded messages, TCPtraceroute sends TCP SYN message to the destination port 80 (Since port 80 is the known port for the web, most firewalls allow packets intended for port 80). If the target host is running any service on the port 80 then it responds with a SYN ACK, otherwise it will reply with an RST message. If the sender receives a SYN ACK message then it will sends an RST message to reset the connection.

Paris Traceroute

Paris Traceroute [1] is developed to improve the accuracy of the traceroute utility. Traceroute suffers from two major flaws: First, it can occasionally report false links between routers and second, it can miss existing links between the routers. To understand these two deficiencies, consider the example illustrated in Figure 4.

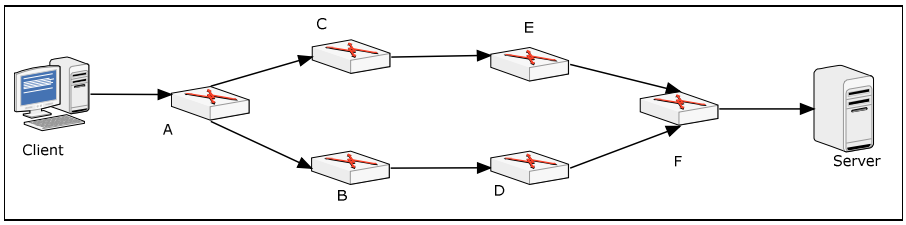


Figure 1 – Traceroute Problems
(based on the example mentioned in [1])

The Traceroute client sends traceroute packets with increasing TTL to the server. Router A implements load balancing[j] and selects either router C or router D based on the load balancing policy. For the first traceroute packet sent by the client with a TTL value of 1, the router A drops the packet and replies with an ICMP *time exceeded* message. If the second packet with a TTL value of 2 is forwarded to router B (where it is dropped) and the third packet with TTL value of 3 is forwarded to router C (where it is dropped at router E), then it would appear that there is a link between router B and router E. Additionally the link between router C and router E remains unnoticed. The first flaw is the appearance of a false link between C and D, whereas the second deficiency is the failure to observe the link between C and E.

Paris traceroute corrects these two problems by maintaining the probe packet header in a manner that permits all probes to follow the same path in case of per flow load balancing [1]. In order to achieve this, Paris traceroute modifies the packet header information such that the load balancing portion of the header remains constant. This implies that the other fields in the header are modified such that the load balancing fields remains constant and the response packets are matched with the probe packets. Paris traceroute achieves this task as follows: In case of ICMP traceroute packets, Paris traceroute modifies the sequence number and the identifier field. For UDP packets, Paris traceroute varies the checksum field. Since checksum is used for data integrity, payload of the packet is also modified.

[j] A Router has multiple interfaces. Load balancing implies that in order to route a message to the destination, a router selects an interface based on the load on the network. Load can be balanced based on per packet, per flow or per destination policy [3] [16]. In per packet balancing, each packet is balanced separately, whereas per flow balancing works by sending all the packets of a particular flow to the same interface. A flow is identified using various fields in the IP header and the transport header. These include the source address, source port, destination address, destination port, the protocol, the type of service (TOS), ICMP code and checksum fields. In per destination policy, load is balanced according to the destination.

However, since per packet load balancing is random, Paris traceroute is unable to completely eliminate these problems when per packet load balancing is used.

4.4. Bandwidth

Pathload

Pathload [15] is a bandwidth measurement tool, based on the PRM model. It induces congestion in the network and monitors the OWD between the source and the target. The moment where the OWD increases marks the start of congestion and the bandwidth can be computed. Note in order to compute OWD, Pathload does not require synchronized clocks at both ends as it is only interested in detecting changes in OWD.

Pathchar

Pathchar [26] is a tool to measure capacity at each hop. Like the traceroute utility, it uses the TTL field in the packet header to get a response from each hop and measure RTT.

The path-capacity estimation is based on the principle that the RTT to each hop consist of three components: (i) serialization delay, which is the time utilized to serialize the packet for transmission along a path (ii) propagation delay, the time taken by a packet to traverse along the path, and (iii) queuing delay, the delay encountered by a packet while waiting in a queue at the router before being processed. Of the three delay components, the serialization delay depends upon the size of the packet (L) and the capacity (C) of the path, *i.e.* L/C, the propagation delay is constant, whereas the queuing delay varies with size of the packet and the available bandwidth of the path.

To measure path capacity at hop n, pathchar sends a series of probes with same packet size and with a fixed TTL value of n. The n^{th} hop router responds by sending an ICMP *time exceeded* message for each received probe. Pathchar obtains the RTT from the ICMP response for each probe. In this manner, a series of RTT values for the n^{th} hop are obtained. Pathchar assumes the minimum RTT observed represents an instance in which the probe and its response did not experience any queuing delay. Therefore, the minimum RTT value consists of only two components: (1) propagation delay and (2) serialization delay. Of the two, the later is a function of path capacity and packet size. Since packet size is

known, the path capacity can be computed via the slope of the minimum RTT up to hop n [31].
One major limitation of pathchar is that it assumes that every router which receives an IP packet with TTL of 1 responds with an ICMP *time exceeded* message. This assumption works fine for layer-3 routers, however, for layer-2 routers, which operate at the link layer and route messages on MAC addresses, this is not true [32]. Layer-2 devices create serialization delays based on packet size and path capacity, however they do not respond with ICMP error messages. As a result, the estimated capacity is not accurate.

Spruce

Spruce [41] is a bandwidth measurement tool that is built on the PGM model. It uses a probe pair to estimate the available bandwidth along a path. It utilizes a series of probe pair responses and computes the average.

TCP throughput or Bulk Transfer Capacity (BTC) measurement

TCP throughput is the capability of the path to transfer TCP traffic in bulk. Note that this capability may differ from the actual capacity of the path due to many reasons. These include TCP flow control, packet loss, TCP congestion avoidance, routing preferences. Since a large portion of Internet activity constitutes TCP traffic, TCP throughput provides an estimate of effective communication capacity in most cases.
Iperf [13] is a tool that measures TCP throughput over an Internet path. It works by sending bulk TCP streams over the path and measuring the amount of payload and the delay encountered for transmission. Iperf requires access to two machines that can act as a sender and a receiver. It provides various configuration options to the user.
Iperf can also be used for measuring UDP throughput. Alternatively, Iperf can also be used to generate bulk amount of TCP/UDP traffic in the network. It is available for major operating systems including UNIX and windows and also has an available GUI interface.

4.5. Multi-purpose tools

In section 4.1, we described the ping tool which can be used to measure RTT and loss rate. We also explained traceroute in section 4.3 which can be used to detect

route, measure RTT or estimate packet loss. Besides the ping and the traceroute utilities there are several other tools which have multiple usages.

Scriptroute

Scriptroute [38] [39] is a public network measurement infrastructure. It allows users to run their network measurement scripts through a web interface and measure latency, detect packet loss or estimate a path. The scriptroute facility maintains a list of active servers which runs the scriptroute daemon.
Each active server runs a web server through which it accepts the user scripts via the CGI post format. The measurement scripts are written in Ruby – a user friendly scripting language. The web server invokes the Ruby interpreter which parses the script and interacts with the scriptroute daemon to facilitate measurement in a controlled environment. The scriptroute daemon sends measurement packets via a raw socket[k] interface. Scriptroute measurement packets are generated based on the type of request submitted by the user. For instance, if a user desires to execute traceroute to the destination host then TTL-limited UDP messages will be generated. Finally, the response from the destination is matched to the probe request and result is displayed on the web interface.
The major advantage of scriptroute is that it aims to provide general availability to the normal users. Another benefit is that the measurements are run in a controlled environment to prevent security threats such as DOS attacks.

Pathping

Pathping [27] is a multi-purpose network utility. It is used to detect the route to the destination and compute hop-level RTT and loss rate, i.e., latency and loss rate for all the intermediary hops from the source to the destination. It is built-in on Windows operating system.
Pathping is a combination of traceroute (tracert) and ping. It first sends the traceroute messages to the destination host and determines all the intermediary routers. It then sends ping messages (through ICMP *echo messages*) to the intermediary routers and compute latency and loss rate. The advantage of Pathping is that it can detect the packet loss to intermediary routers and thus can

[k] Raw socket is a term used to denote for a socket that is delivered to an application without stripping the headers, i.e., without going through the TCP/IP stack.

be used for network diagnosis. Since the analysis is computed over a longer time period, the results are extensive compared to the normal ping. For this reason, Pathping analysis could also take longer time.

5. Indirect Measurements

The measurement aspects explained so far employ direct measurements, i.e., end-to-end characteristics of a path are measured directly, either through actively sending probes to the destination and reading the response, or through passively monitoring existing flows sent by some other application.

Direct mode provides rapid and accurate mechanism for measurements. However, the schemes based on direct measurements have limited scalability. In a network of n nodes, if each node is to measure its characteristics to every other node then there would be $O(n^2)$ end-to-end measurements.

Therefore, often indirect measurements are utilized for improved performance. The indirect mode involves use of heuristics or tomography to estimate the network characteristics of a path. The direct measurements from one or more components of a path are utilized to estimate end to end characteristics. As a result, the indirect method is not precise, however it incurs lower overhead. In large networks and in applications where an approximate of measurement is sufficient (such as locating proximity of a node), indirect measurements are utilized. Figure 5 illustrates an example of indirect measurements, where network characteristics of a path between nodes A and C can be estimated if characteristics of paths AB and BC are known.

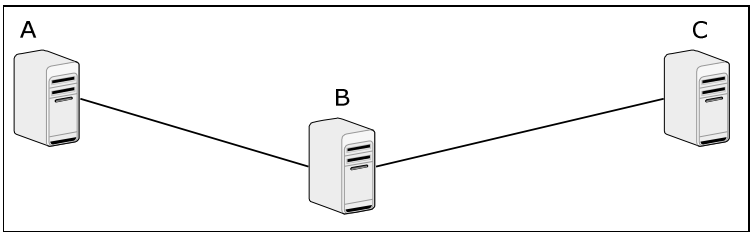


Figure 2 - Example of Indirect Measurement

The end-to-end path characteristics are computed through an aggregation function, which varies with the network characteristic. For instance, in case of latency, the overall latency from node A to node C is the sum of the latencies for

the paths AB and BC. However for bandwidth, the minimum bandwidth of the paths AB and BC will be used.

An important consideration for indirect measurements is that the information about network characteristics of different paths needs to be exchanged among different nodes. For the example mentioned in figure 5, in order to compute the path characteristics for the path AC, node A must obtain information about the path BC.

Figure 5 is a specific example of indirect measurements. Other techniques including inference based on evidence measurements from nearby hosts also exist. A few of such methods are described below.

King

King [11] is a latency estimation tool that utilizes recursive DNS queries. To estimate the latency between two Internet hosts, it first finds the DNS servers that are topologically close to the two end hosts. The network latency between the two DNS servers is then computed using recursive DNS queries, which is also approximated as the latency between the two hosts.

IDMAPS

The IDMAPS [8] project utilizes special hosts, called tracers, deployed widely along the Internet. Each tracer measures its distance with all the other tracers and also with a set of hosts involving nearby IP addresses. The measured distance is sent to set of dedicated servers called HOPS, to build the virtual topology. HOPS servers estimate the latency between two end hosts as the sum of the latencies between the hosts and their nearest traces, plus the distance between the tracers. Figure 6 illustrates the concept of the IDMAPS approach.

Iplane

The iplane [19] project from the University of Washington can be used to detect paths, estimate loss rate or approximate latency. It utilizes a large number of special hosts called vantage points that are distributed throughout the Internet. Vantage points send measurement probes to each other and generate an atlas of the Internet core. The iplane is built along the idea that if there are a large number of vantage points then the latency between two Internet hosts is approximately equal to the latency between their nearby vantage points. The atlas is utilized to predict network characteristics in this manner.

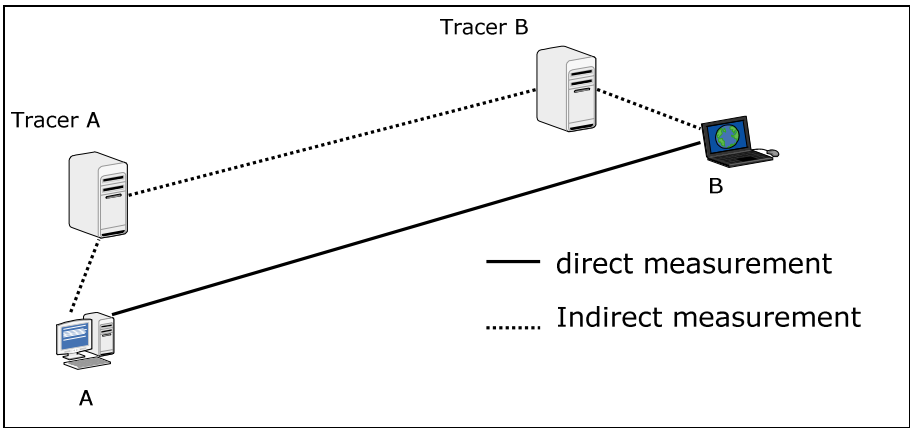


Figure 3 – Idmaps approach for latency measurement

6. Thoughts for Practitioners

Whether developing a tool to measure network behavior, or using an existing technique for measurement, two factors are of particular importance. First, the accuracy of the measurements should meet the requirements of an application, and second, the measurements should bear low cost.

Two aspects are significant for improving accuracy and reducing overhead: (1) Correctness of the measurement or estimation techniques and (2) precise implementation of the algorithm in obtaining accurate results. Sections 2 to 5 were related to explaining basic concepts and enlightening various measurement strategies. In this section, we focus on improving implementation strategies.

In general, many strategies could be significant for developing an efficient measurement utility.

- Efficient design for the measurement tool. This includes better utilization of the available resources such as CPU, memory and the network. For instance, selecting appropriate number of threads and processes, ensuring mutual exclusion for the critical section, choosing suitable methods for inter-process communication, and opting proper mechanism for detecting timeouts, using signals and timers.
- Selecting efficient method for data collection. For instance, in case of active monitoring, use appropriate socket communication for sending probes and receiving responses.
- Proficient strategies for analyzing network data, and
- Accurate and efficient methods for clock synchronization and time stamping

Of the four points mentioned here, the first two are related to operating system concepts and socket communication techniques. Therefore, they will not be discussed further in this chapter. The later two points are related to network measurements and are explained below.

6.1. Analyzing Network Traffic

Tcpdump

Tcpdump is a command line tool used to analyze network traffic. It is built-in to most versions of UNIX including Linux, BSD and Solaris. The Windows version of tcpdump is called WinDump. Tcpdump can be used to analyze network packets including packet header and data (pay load) information. It is a very powerful tool that can be configured for the analysis of packets that match user-specific criteria (for instance, all packets destined for port 80).

Tcpdump utilizes Berkeley Packet Filter (BPF) to capture packets. The BPF filter has two components a network tap, and a user defined filter. The network tap is used to capture packets, whereas the filter is utilized to check if the packet should be accepted, and how much of it should be delivered to the application [20]. The BPF can be used to place the network interface card in the promiscuous mode. In promiscuous[1] mode, the network card captures all the packets going through the network and not just the packets destined for it. When a device driver (of the Ethernet card) senses a packet, it forwards the packet to the BPF, which passes it through the user-specific filter (tcpdump filter in this case). The packets that match the user-specific criteria are forwarded to the user-level program. Note that in this manner any other program (such as ethereal, a network analysis tool) which uses the BPF filter can also retrieve its desired packets. To each packet forwarded to the application, BPF attaches a timestamp for the captured packet.

The use of BPF filter by tcpdump makes it a very powerful tool. A tcpdump program running on a host can be used to analyze network traffic originating from (or destined to) a different host in the network. Note that in this manner, a network device can also be configured as a tap, or a passive monitoring device such that the packets can be copied, while the normal operation between the sender and the destination remains uninterrupted (Section 2.2).

[1] In non-promiscuous mode, network interfaces only capture packets destined for them or the packets that are broadcast to the whole network.

Since tcpdump can be used for network analysis of any host in the network, many implementations of UNIX restricts the configuration of network device in promiscuous mode to the superuser (root).

Libpcap

The back-end of the tcpdump tool is based on libpcap, an application programming interface (API) used for capturing and analyzing packets.

Example 6 : Open the device in promiscuous mode
Solution: The tcpdump tutorial mentions the following system call for opening a device in promiscuous mode.

```
pcap_t *pcap_open_live(char *device, int snaplen, int promisc, int to_ms, char *ebuf)
```

where, `pcap_t` is a handle returned by the function call.
`device` is the name of the name of the Ethernet card sniff into
`snaplen` is an integer which defines the maximum number of bytes to be captured by pcap
`promisc` is 1(true) if the device is to be set in the promiscuous mode
`to_ms` is the timeout in milliseconds, a 0 means no timeout.
ebuf is for storing error messages.

By using libpcap, a programmer can develop his or her own version of tcpdump, build a packet sniffer, or analyze captured data in several ways. Libpcap is based on based on the C language (the windows version is WinPcap). For programming in other languages wrappers are provided. Example 6 shows the function call (of the pcap library) which can be used to open the device in the promiscuous mode. A detailed description and tutorial can be found at the tcpdump/libpcap website [42].

6.2. Clock Synchronization

Accurate CPU clock is needed for precise time stamping as well as for measuring OWD. In general, there are two clocks in a computer, a battery powered hardware clock and a software clock. The hardware clock is used to keep track of time when the system is turned off, while the software clock (also called as CPU

clock) is used to access the time when the computer is in operation. Software clock maintains a timer to generate timer interrupts and keep track of system time.

The software clock generally runs at slightly different rate than the actual time. The rate is also affected due to temperature, aging and environmental effects. This leads to inaccurate time from the software clock (also known as CPU clock). In order to correct this error, CPU clocks are generally synchronized with a timing source. The process of synchronizing the CPU clock with a reference source is called **Clock Synchronization**. The difference in the rate of the two clocks is called as **clock drift**, whereas the difference in actual time is known as **clock skew**. The timing source for clock synchronization is generally through a time server in the network. The standard timing source for computers on the Internet is NIST - National Instituted of Standards and Technology.

Clocks can be synchronized in two ways [7]. A CPU clock is **externally synchronized**, if the difference between it's time and the reference time is bounded by a given constant. Whereas, in a network system a set of clocks are **internally synchronized** if the difference between their times is bounded by a constant. Note that externally synchronized clocks are internally synchronized as well. However, the reverse is not necessarily true.

NTP

NTP [21] [22] is the most widely and universally adopted protocol for synchronizing computer clocks across the Internet. The servers providing clock synchronization are organized in a hierarchical manner, with each level referred to as a stratum. Servers at stratum 0 (root level) are synchronized to an external reference source such as UTC (Universal Time Coordinate) or GPS (Global Positioning System) and servers at the subsequent levels are synchronized with the servers at the preceding level. A client seeking clock synchronization sends NTP probes to the desired NTP server which responds with the time at the server. The NTP algorithm (at the client) compares the time of the software clock with the time at the server, computes the offset of the two clocks and adjusts the offset and rate of the software clock.

NTP offers a low cost and affordable clock synchronization mechanism for most Internet applications. However, for applications that require near-accurate clocks (such as applications measuring latency), the NTP-based clock synchronization mechanism is not acceptable [23] [24]. Due to variable rate of the software clock, the offset of the software clock can be high, which may lead to a large rate adjustments. The problem intensifies when only one-way latency is to be

measured, as one-way latencies are expected to be lower than the round trip latencies and an error of few ms could significantly reduce the accuracy of measurement.

TSC register-based clock

To counter the limitations of NTP, a TSC (time stamp counter register) based solution is proposed [25] [4]. The mechanism involves using a virtual clock based on the smooth oscillations of the TSC register and synchronizing the clock with a nearby NTP stratum1 server.

6.3. Time Stamping

The Timestamp of an event refers to the actual time of occurrence of the event. In network measurements, this normally refers to the time a probe is sent or received. Obtaining precise time stamps is pivotal for accurate network measurements.
In UNIX, time stamp of an event is generally expressed through the C `struct timeval`, which has following members

```
struct timeval
{
            long tv_sec; // second portion
            long tv_usec; // micro second
}
```

Example 7: The C `struct timeval` has microsecond (usec) precision. Is there any method available to obtain higher resolution?
Solution:
The `struct timespec` provides nanosecond (ns) precision. The format is

```
struct timespec
{
           time_t  tv_sec    //seconds ;
           long    tv_nsec   //nanoseconds ;
}
```

Note that the `struct timeval` is useful for denoting the reference time (time elapsed after the occurrence of certain event). In order to utilize this `struct` for the system time, UNIX programmers use the `gettimeofday()` system call to read the CPU clock and obtain the system time since epoch – the time elapsed since midnight UTC of January 1, 1970.
Using `gettimeofday()` system call after receiving a probe packet or before sending a probe is not very accurate. This is due to the fact that in an operating system that supports multi-tasking, the processing of the one task can be delayed due to scheduling of another task. If the `gettimeofday()` is called before sending a packet, then operating system could schedule another task after the `gettimeofday()` call but before sending the packet. Similarly, if the `gettimeofday()` is called after sending the packet then the OS could schedule another task after sending the packet but before calling the `gettimeofday()`. The problem persists when the response of the probe is received.
There are several approaches to obtain higher accuracy.

RTLinux

Since the time stamping error is associated with OS scheduling, a solution lies in minimizing the effect of OS scheduling. One such approach is to use a real-time operating system such as RT-Linux [45] [46]. The RT-Linux system runs above the hardware layer but below the Linux OS. Tasks that need deterministic processing are run as RTLinux tasks and can only be interrupted by hardware interrupt or other high priority tasks, whereas the lower priority tasks are still run by the Linux OS. RTLinux is easily available in both free and commercial versions. One major disadvantage with RTLinux is that its deployment is still limited i.e. not many hosts on the Internet utilize RTLinux.

Kernel Time-stamps

Even though RTLinux can correct the OS scheduling time stamping errors, using `gettimeofday()` before or after the send and receive operation is still not very accurate, as the return time from the `gettimeofday()` denotes the current system time and not the actual send or receive time of the packet. One method to obtain packet sent or receive time stamps is to utilize kernel time stamps.
When a network card receives a packet, the device driver (of the network card) sends an interrupt to the processor, so that kernel can pull the packet from the

queue of the device driver. When Linux OS receives the packet from the driver of the network card, it also measures the time stamp. In [40], the authors have mentioned three ways to obtain these time stamps. These are tabulated in Table 1.

Table 1 – Kernel time stamps

Method	Comments
`ioctl` system call	Use `ioctl` system call with `SIOCGSTAMP` option specified as `ioctl_type` [6]. Ping also uses ioctl. Only time stamps from received packets can be obtained.
recvmsg ()	`recv_msg` system call with `SO_TIMSTAMP`. Can be run as non-root user.
Pcap time stamps	Pcap time stamps are obtained via library libpcap [28]. Tcpdump utility in UNIX also utilizes these time stamps. One advantage is that time stamps of sent packets can also be obtained. However, use of Pcap is restricted for users with root privilege.

Example 8: How can the `iotcl` system call is utilized to read the time stamp of a received packet?
Solution:

```
int recv_fd;
struct timeval recv_ts;

//Create socket and receive the message using recv or recvfrom system call

ioctl(recv_fd, SIOCGSTAMP, &recv_ts);

// The struct recv_ts has the time stamp of the received packet.
```

DAG cards

So far, we have discussed about improving the accuracy of time stamps by minimizing the error due to OS scheduling. However, a time stamping error can also occur by reading an inaccurate clock [25].
DAG card project from the University of Waikato, New Zealand solves both the problems of time stamping i.e. error due to clock synchronization and error due to OS scheduling. The DAG card is a passive mode device that is normally connected to a hub to which the probing machine is connected.
The internal clock of the DAG card is synchronized with the GPS (Global Positioning System) time system. The DAG card can read the timestamps of sent and received packets with an accuracy of 100 ns [5] that can be obtained via tcpdump [43]]. The DAG card timestamps the first bit of the probing packet, while most other network cards will timestamp only after the packet has fully arrived.
The provision of the DAG card requires the placement of a GPS antenna at the roof of the building. This limitation, aided by the high cost of GPS antenna has limited its usage for end-users. However, the measurement facilities such as RIPE [36] use DAG cards for high accuracy.

7. Research directions

Network measurement is an evolving topic and entails many directions for future research. Some of the significant trends are summarized below.

- The growth and expansion of the Internet has lead to the proposal and deployment of new architectures and evaluation platforms such as Internet2 [12], PlanetLab [29] and GENI [10]. Many researchers are now focusing on measuring network characteristics of these architectures [17] to understand their topology and network usage. Thorough understandings of these architectures would promote better planning for the future expansion of the Internet.
- Researchers are also utilizing network measurements to understand the usage and characteristics of popular Internet applications such as VOIP and Bit Torrent [30]. This would lead to improved performance form these applications.
- On the implementation side, improving clock synchronization techniques (specifically needed for OWD measurements) and better time stamping mechanisms that have low cost have also been the focus of many researchers.

- A significant amount of study and research proposes new techniques for indirect measurements. Since indirect measurements promote scalability, effective use of indirect measurements can promote measurements on a large scale. The focus of the current research is to promote indirect measurements and estimation methods [2] that provide near accurate measurements and that have low cost.

8. Conclusion

The growth of the Internet will continue to increase the significance of network measurements in obtaining better performance from distributed applications. In this chapter, we have explained the basic principles of network monitoring.
The chapter presents a detailed analysis of important techniques for the measurement of four important network characteristics, i.e., latency, path detection, loss rate and bandwidth. It also describes key concepts related to network measurements, including techniques for clock synchronization, strategies for time stamping of probes and methods for analyzing packets. The chapter also elaborates on the usefulness of different transport layer protocols (i.e. TCP, UDP and ICMP) for obtaining network measurements and continued this discussion to describe some important measurement tools such as Ping, Traceroue, Pathchar, Sting, Scriptroute, Spruce and Paris Traceroute.

9. References

1. B. Augustin et. al. Avoiding traceroute anomalies with Paris traceroute. Internet Measurement Conference (2006)
2. Y Chen, D Bindel, HH Song, RH Katz Algebra-Based Scalable Overlay Network Monitoring: Algorithms, Evaluation, and Applications. IEEE/ACM Transactions on Networking, (2007).
3. Cisco, “How does load balancing work?” http://www.cisco.com/warp/public/105/46.html
4. E. Corell, P. Saxholm, and D. Veitch, A user friendly TSC clock, *Passive and Active Measurement Workshop* (2006)
5. DAG support documentation.
http://www.cubinlab.ee.unimelb.edu.au/probing/docs/dag_cards.php
6. DCROUTE an improved version of traceroute.
http://www.cubinlab.ee.mu.oz.au/probing/dcroute/
7. C. Fetzer and F. Cristian Integrating Internal and external clock synchronization. *Real Time Systems. Special issue on global time in large scale distributed real-time systems*. (1997)
8. P. Francis, S. Jamin, C Jin, Y Jin, D Raz, Y Shavitt, and L Zhang. IDMAPS: A Global Internet Host Distance Estimation Service. *IEEE/ACM Trans. (On networking*, (2002).
9. Fping measurement utility. http://fping.sourceforge.net/
10. GENI. Global Environment for Network Innovations. http://www.geni.net/

11. K. P. Gummadi, S. Saroiu, and S. D. Gribble. King: Estimating Latency between Arbitrary Internet End Hosts. *Internet Measurement Conference*, (2002).
12. Internet2. www.internet2.edu
13. Iperf – The TCP/UDP bandwidth measurement tool http://dast.nlanr.net/Projects/Iperf/
14. V. Jacobson, Traceroute source code and documentation. Available from: ftp://ftp.ee.lbl.gov/traceroute.tar.Z.
15. M. Jain and C. Dovrolis, End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput. *ACM SIGCOMM,* (2002).
16. Juniper, Configuring load-balance per-packet action. http://www.juniper.net/techpubs/software/junos/junos70/swconfig70-policy/html/policy-actions-config11.html
17. SJ Lee, P Sharma, S Banerjee, S Basu, R Fonseca Measuring Bandwidth Between PlanetLab Nodes. *Passive And Active Network Measurement Workshop* (2005).
18. Combining active and passive network measurements to build scalable monitoring systems on the grid. *ACM Sigmetrics Performance Evaluation Review* (2003).
19. H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy and A. Venkataramani. iPlane: An Information Plane for Distributed Services. *Usenix OSDI* (2006).
20. S. McCanne and V. Jacobson. The BSD packet filter: A new architecture for user-level packet capture. *Winter Usenix Conference*. (1993)
21. D. Mills. Internet Time Synchronization: The Network Time Protocol. *IEEE Transactions on Communications COM, vol 39*, pages 1482{1493, (1991).
22. D. Mills. Precisions Synchronization of Computer Network Clocks. Technical Report 93-11-1, Electrical engineering Department University of Delaware, (1993).
23. A. Pasztor and D. Veitch. High Precision Active Probing for Internet Measurement. *Proceedings of INET* (2001).
24. A. Pasztor and D. Veitch. PC Based Precision Timing Without GPS. *ACM SIGMETRICS international conference on Measurement and modeling of computer systems,* (2002).
25. A. Pasztor and D. Veitch. Robust Synchronization of Software Clocks. *Internet Measurement Conference (IMC)*, (2004).
26. Pathchar, http://www.caida.org/tools/utilities/others/pathchar.
27. Windows XP Pathping.
www.microsoft.com/technet/prodtechnol/windows2000serv/reskit/cnet/cnbd_trb_vxmb.mspx
28. Libpcap tutorial. http://www.cet.nau.edu/~mc8/Socket/Tutorials/section1.html
29. PlanetLab. http://www.planet-lab.org
30. J. Pouwelese , P. Garbacki, D. Epema and H. Sips. The Bittorrent P2P File-Sharing System: Measurements and Analysis. *International Workshop on Peer-to-Peer Systems* –IPTPS. (2005).
31. R. Prasad, C. Dovrolis, M. Murray, and kc claffy. Bandwidth estimation: Metrics, measurement techniques, and tools. *IEEE Network*, vol. 17, no. 6, pp. 27–35, (2003).
32. R. Prasad, C. Dovrolis and B.A. Mah. "The effect of layer-2 store-and-forward devices on per-hop capacity estimation". *IEEE Infocom*, (2003).
33. RFC 793 Transmission Control Protocol. http://www.faqs.org/rfcs/rfc793.html
34. RFC 896. Congestion Control in IP/TCP Internetworks. http://rfc.net/rfc896.html
35. RFC 1930 Guidelines for creation, selection and registration of Autonomous Systems. http://tools.ietf.org/html/rfc1930
36. RIPE Network Coordination Center. http://www.ripe.net/
37. S. Savage. Sting: a TCP-based Network Measurement Tool. *USENIX Symposium on Internet Technologies and Systems* (1999).
38. Scriptroute Network Measurement. www.scriptroute.org

39. N. Spring, D. Wetherall, and T. Anderson. Scriptroute: A facility for distributed Internet measurement .*USENIX Symposium on Internet Technologies and Systems – USITS* (2003).
40. N. Spring, L. Peterson, A. Bavier, and V. Pai. Using PlanetLab for Network Research: Myths, Realities, and Best Practices. *ACM SIGOPS Operating Systems Review*, 40:1, pages 17--24, (2006)
41. J.Strauss, D. Katabi, F. Kaashoek. A measurement study of available bandwidth estimation tools. *Internet Measurement Conference* (2003).
42. Tcpdump/Libpcap Public repository. http://www.tcpdump.org/
43. Tcpdump FAQ. http://www.tcpdump.org/faq.html
44. Unix Socket FAQ. http://www.unixguide.net/network/socketfaq/
45. V. Yodaiken and M. Barabanov. A Real-Time Linux. *Linux Applications Development and Deployment Conference – USELINUX*. (1997).
46. V. Yodaiken. The RTLinux Manifesto. *The 5th Linux Expo*, (1999).

10. Terminologies

Host

A host refers to an entity that sends or receives a message. The entity could refer to an application that sends a message or the actual computer itself on which the application is deployed.

Probe

A Probe refers to an event of capturing network information. Specifically, it involves sending message and reading its response in order to infer information about the characteristics of Internet communication between the sending and receiving host.

Probing Interval

The Probing interval specifies the interval between subsequent probes.

Packet:

A packet refers to a chunk of data being transferred between two hosts. When packets are used to send information, a large message can be disassembled into packets and sent across the network. The maximum size of the packet that can be transferred across the paths is called MTU (Maximum Transfer Unit) of the path.

Flow

A network flow refers to a sequence of packets sent as a part of a specific application from one host to another.

Source

The source implies the host that sends the packet.

Destination

The destination is the host that is intended to receive the packet.

Hop

Hop refers to an intermediary host (or router) being traversed while sending a packet from source to the destination. When a packet is sent from the source host to the destination host, it traverses through various intermediate hosts called routers. Each router computes the path and determines the next immediate router. The number of routers traversed from source to destination is usually refers to the number of hops.

Path:

The path is the end-to-end route from source to destination. It includes all the intermediary routers that are utilized to send a packet. There could be multiple paths from source to the destination, i.e., each path could have different routers.

Queuing Delay

Queuing delay is the time spent by a packet in the queue of the router before being processed. At each hop along a path, packets arrive at a router, which processes it and sends it to the next hop. If at any router, the arrival rate of a packet is greater than the packet processing rate then the packets are stored in a queue. Queued packets increase the path latency. If the queue becomes full then the packets are dropped which causes packet loss.

Congestion

Congestion is the state in which the increase in the load deteriorates the quality of the network. The decrease in quality is usually observed in terms of high queuing delay, increase packet loss and denial of new connections. If the congestion is lasting over a period of time then it is called persistent congestion, otherwise congestion over a short duration or spike is called transient congestion.

Timestamp

The timestamp of an event refers to the actual time of occurrence of the event. In network measurements this normally refers to the time a probe is sent or received.

Clock Synchronization

Clock synchronization is the process of synchronizing the system time with a reference source. A synchronized clock is needed in order to obtain an accurate time stamp.

11. Questions

Question 1: Suppose you modify your web browser to obtain pcap timestamps of HTTP requests and response. Is this an example of active or passive monitoring? Which network characteristic you can measure by using this approach?
Solution: This is an example of passive monitoring. The timestamps can be used to measure Round Trip Time (RTT) from the client to the web server.
However, the RTT measurements may not be accurate if the server is heavily or if it process queries (such as database query) based on the client's requests.

Question 2: What methods can be used to passively collect the data?
Solution: In order to collect the data in a passive mode, a hub, a tap or port-mirroring can be used.

Question 3: What sort of problems could be yielded due to filtration of ping.
Solution: Many routers or networks filter ping. Ping filtration could lead to inaccurate estimate of packet loss or inference of service unavailability by the client.

Question 4: What is the minimum value of TTL required to send a packet from client to the server, in the network configuration depicted in Figure 4? What is the maximum number of TTL that can be used?
Solution: In the network configuration of figure 4, the minimum value of TTL required is 3 which is a path through routers A and D. The maximum TTL can be complicated. Since both the routers B and C have links to each other, a packet can traverse between them indefinitely. However, the purpose of TTL is to avoid infinite loop, therefore, the maximum TTL is the maximum value allowed. For an 8 bit field the maximum value allowed is 255.

Question 5: In traceroute, how does the sending host determine not to send any further packets with increasing TTL?
Solution: The procedure of deciding not to send any further packets with increasing TTL depends upon the underlying protocol. If the traceroute is utilizing UDP, then the destination host will send an ICMP *destination unreachable* message to the original sender, indicating that the destination host is not running any service on the particular port. However in case of ICMP (Windows implementation), the traceroute message is not destined for any particular port, and an ICMP *echo reply* (type 0) message will be sent. In either case, the original sender will recognize that the final destination host is reached.

Question 6: Compute the total capacity for 100 Mbps Fast Ethernet. Assume packet size of 100 bytes.
Solution: The interframe gap must be 96 bit times, which is equal to 12 bytes. The overhead is 38 bytes i.e. 14 byte Ethernet header, 4 bytes CRC, 8 bytes for preamble and 12 bytes for interframe gap.
Therefore the capacity C = 100 (100/138) = 72.46 Mbps.

Question 7: Is it safe to assume that the capacity of a path remains constant? How about the available bandwidth?
Solution: The capacity of a path may remain constant over a long period. However, in case of a route change or path upgrade, the capacity of the path could be changed. In comparison, the available bandwidth of a path may change abruptly. A link may experience sudden change in its utilization, resulting change in the available bandwidth.

Question 8: Is narrow link of a path, same as the tight link? Explain your answer?
Solution: Narrow link is not necessarily same as the tight link. For instance, consider an example where a link between two internet routers A and B is

included in several internet paths. At a given moment, the link AB may appear as a narrow link for a path, even though it may not be the tight link for the path.

Question 9: What is the disadvantage of using TCPtraceroute?
Solution: The foremost disadvantage of TCPtraceroute is that it requires an extra message. Conventional traceroute schemes utilize two messages: a UDP or ICMP echo message from the sender, and an ICMP time exceed message from the target. Comparatively, in TCPtraceroute, a sender has to send an RST message if it receives a SYN ACK from the target.

Question 10: As a class project you are asked to build a latency monitoring tool. Which time stamping mechanism would you use to read time stamps of sent packets?
Solution: Sent time stamps can be obtained via PCAP utility. A precise mechanism is to use DAG cards. If DAG card is not available then you can obtain time stamps from PCAP library.