

Forum:[Hack Forums](#)

Thread: [Beginner's networking guide](#)

REDRIFLE - November 19, 2011, 9:40 pm

I posted this once before but i don't think even one person ever saw it because i posted it at like 4 a.m so here it is again

Beginner's Networking

Alright we're gonna start from the beginning here, you know what that means.

http://en.wikipedia.org/wiki/OSI_model

Try to think of the OSI model as a support structure almost like that of a building.

Memorize every protocol, learn the meaning of them. With proper knowledge of this structure it can be used practically and ETHICALLY to either secure or compromise a network.[/color]

The OSI model consists 7 layers.

1.the physical layer

2.the data link layer

3.the network layer

4.the transport layer

5.the session layer

6.the presentation layer

7.the application layer

Lets talk about IP from the network layer.

IP (Internet Protocol) is a dotted decimal numerical identification assigned to all devices connected to a network. IP addresses can either be internal or external.

an internal address will almost always have 192.168 in them. An example of a internal IP would look like this 192.168.1.1

An external IP is assigned to a router or access point by your ISP (Internet Service Provider)

IP addresses have a limit of 255 per octet. this is because an IP must consist of 32 bit (or 1 byte) strings, that means 255.255.255.255 in binary would look like 111.111.111.111 (8 one's per octet but it won't let me put that many)

Sometimes you will see an ip that looks like xx.xx.xx.xx/24

LISTENING : In case of a server, waiting for a connection request from any remote client.
 SYN-SENT : waiting for the remote peer to send back a TCP segment with the SYN and ACK flags set. (usually set by TCP clients)
 SYN-RECEIVED : waiting for the remote peer to send back an acknowledgment after having sent back a connection acknowledgment to the remote peer. (usually set by TCP servers)
 ESTABLISHED : the port is ready to receive/send data from/to the remote peer.
 FIN-WAIT-1 :
 FIN-WAIT-2 : Indicates that the client is waiting for the server's fin segment (which indicates the server's application process is ready to close and the server is ready to initiate its side of the connection termination)
 CLOSE-WAIT :
 LAST-ACK : indicates that the server is in the process of sending its own fin segment (which indicates the server's application process is ready to close and the server is ready to initiate its side of the connection termination)
 TIME-WAIT : represents waiting for enough time to pass to be sure the remote peer received the acknowledgment of its connection termination request. According to RFC 793 a connection can stay in TIME-WAIT for a maximum of four minutes known as a MSL (maximum segment lifetime).
 CLOSED : connection is closed

..

I think we have TCP down pretty good.
Now we're on to UDP or User Datagram Protocol.

In contrast to TCP, UDP focuses more on reduced latency rather than reliability because UDP needs no handshake to initiate a connection. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for delayed packets. UDP is more unreliable because with no three way handshake, the connection may not be perfectly in sync and thus resulting in the possibility of malformed packets and/or packets may not arrive in order, meaning a UDP connection is more likely to be interrupted than a TCP connection, however, because UDP treat's each request as an independent transaction, UDP supports packet broadcast (Unlike TCP).

(From wiki)

Comparison of UDP and TCP

TCP

Transmission Control Protocol is a connection-oriented protocol, which means that it requires handshaking to set up end-to-end communications. Once a connection is set up user data may be sent bi-directionally over the connection.

Reliable – TCP manages message acknowledgment, retransmission and timeout. Multiple attempts to deliver the message are made. If it gets lost along the way, the server will re-request the lost part. In TCP, there's either no missing data, or, in case of multiple timeouts, the connection is dropped.

Ordered – if two messages are sent over a connection in sequence, the first message will reach the receiving application first. When data segments arrive in the wrong order, TCP buffers the out-of-order data until all data can be properly re-ordered and delivered to the application.

Heavyweight – TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.

Streaming – Data is read as a byte stream, no distinguishing indications are transmitted to signal message (segment) boundaries.

UDP

UDP is a simpler message-based connectionless protocol. Connectionless protocols do not set up a dedicated end-to-end connection. Communication is achieved by transmitting information in one direction from source to destination without verifying the readiness or state of the receiver.

Unreliable – When a message is sent, it cannot be known if it will reach its destination; it could get lost along the way. There is no concept of acknowledgment, retransmission or timeout.

Not ordered – If two messages are sent to the same recipient, the order in which they arrive cannot be predicted.

Lightweight – There is no ordering of messages, no tracking connections, etc. It is a small transport layer designed on top of IP.

Datagrams – Packets are sent individually and are checked for integrity only if they arrive. Packets have definite boundaries which are honored upon receipt, meaning a read operation at the receiver socket will yield an entire message as it was originally sent.

No congestion control – UDP itself does not avoid congestion, and it's possible for high bandwidth applications to trigger congestion collapse, unless they implement congestion control measures at the application level.

Many online games utilize UDP for its improved speed and simplicity over TCP.

..

Now that you've got some basic knowledge on IP, TCP, and UDP, we're going to have a look at ICMP (Internet Control Message Protocol). ICMP (not like TCP or UDP) is not used for data transmission between computers and is not usually employed by end-user applications. ICMP is used in diagnostic tools such as ping and traceroute. It is used most commonly for sending error messages through a network.

(From wiki)

Internet Control Message Protocol is part of the Internet Protocol Suite as defined in RFC 792. ICMP messages are typically generated in response to errors in IP datagrams (as specified in RFC 1122) or for diagnostic or routing purposes. ICMP errors are always reported to the original source IP address of the originating datagram.[1]

An example ICMP error message is the Time To Live Exceeded message. Every machine (such as an intermediate router) that forwards an IP datagram has to decrement the time to live (TTL) field of the IP header by one. If the TTL reaches 0, an ICMP Time to live exceeded in transit message is sent to the source of the datagram.

Each ICMP message is encapsulated directly within a single IP datagram, and thus, like UDP, ICMP is unreliable.

Although ICMP messages are contained within standard IP datagrams, ICMP messages are usually processed as a special case, distinguished from normal IP processing, rather than processed as a normal sub-protocol of IP. In many cases, it is necessary to inspect the contents of the ICMP message and deliver the appropriate error message to the application that generated the original IP packet, the one that prompted the sending of the ICMP message.

Many commonly-used network utilities are based on ICMP messages. The tracert (traceroute), Pathping commands are implemented by transmitting UDP datagrams with specially set IP TTL header fields, and looking for ICMP Time to live exceeded in transit (above) and "Destination unreachable" messages generated in response. The related ping utility is implemented using the ICMP "Echo request" and "Echo reply" messages.

ICMP can be a dangerous protocol because of the way users can manipulate it. Sending a large amount of ICMP ping query's to a target can cause a DoS of the system. this is called the "Ping Of Death".

~.~.~.~.~.~.~

Well i hope some of you people actually learned something from this.

This tutorial was my first so it's not the best but hopefully it will get some beginners pointed in the right direction and maybe some networking vets can learn a thing or to.

And please if i'm wrong about anything or you spot a mistake in my grammar, please for the love of god, tell me.

Oh...and one more thing,

I WANNA SEE SOME POSTIN!

Techie??? - November 19, 2011, 10:35 pm

Nice tutorial for beginners.

Because you like posting. :D

nzpwnedu - November 19, 2011, 10:43 pm

nice guide bro will use this in the future

Spagnum - November 20, 2011, 8:50 am

Good basic stuff, every hacker should know this...

REDRIFLE - November 20, 2011, 3:41 pm

thank you all, i didn't think i would get such nice replies.

r3pt01d - November 20, 2011, 7:47 pm

Exactly what I've been looking for, many thx

Abs0lute - November 20, 2011, 9:09 pm

Thanks man =)

-Abs0lute

maxcharplzz

(Going on a tutorial spree, and thanking all the people I didnt yesterday)

john10294 - November 20, 2011, 10:53 pm

Nice guide it will help me out alot ?