REMnux Tutorial-4.1: Datagrams, Fragmentation & Anomalies

Rhydham Joshi

M.S. in Software Engineering, San Jose State University

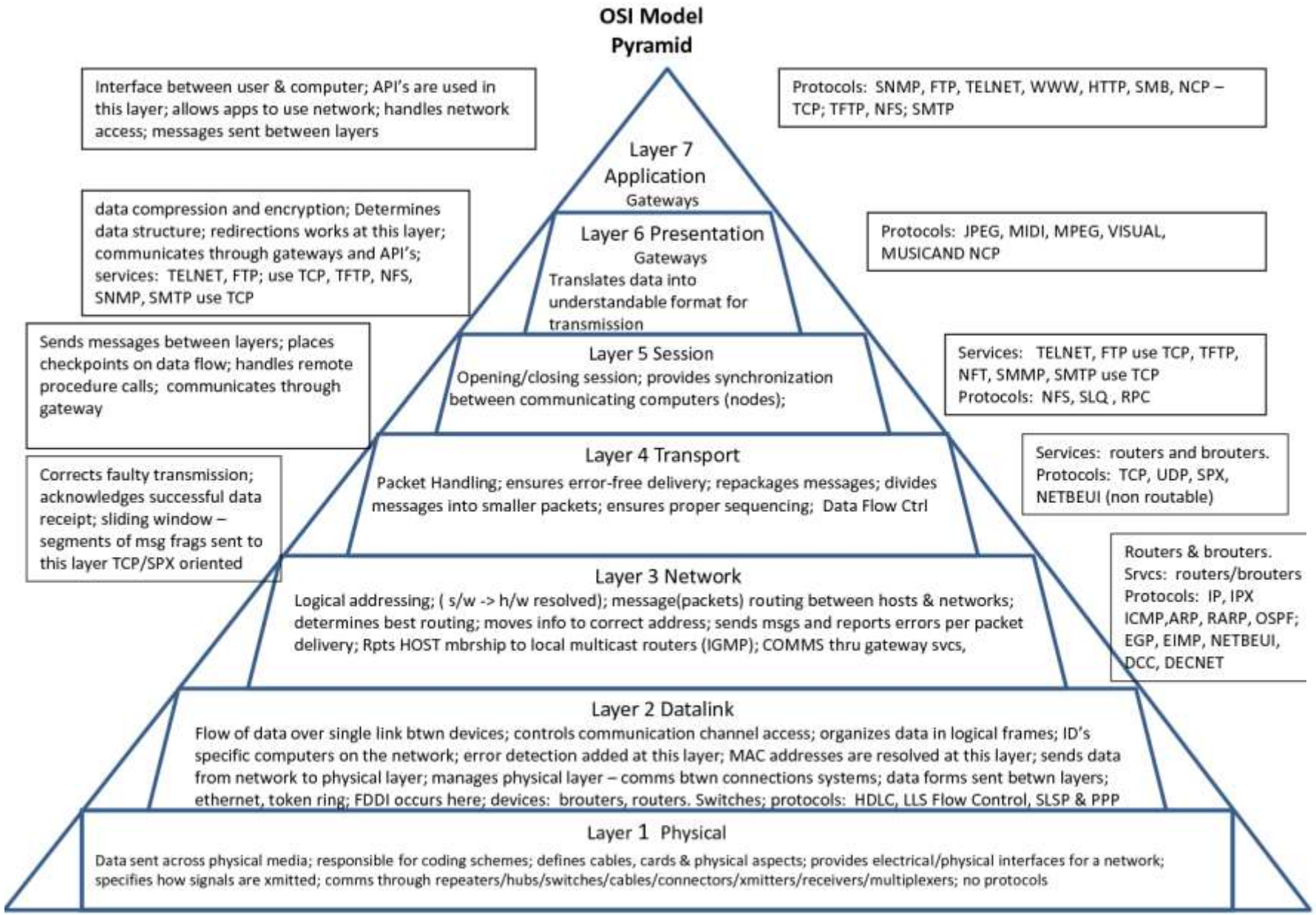
Phone : (+1) 408-987-1991 | Email : rhydham.joshi@yahoo.com

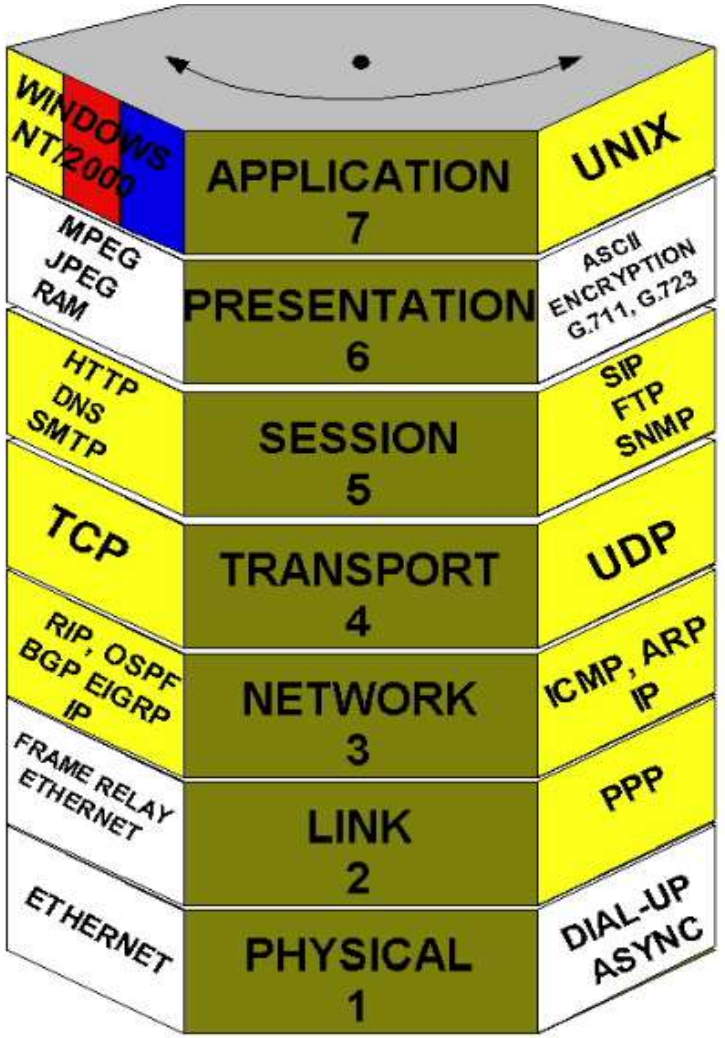
Blog : malwareforensics1.blogspot.com | Linkedin : www.linkedin.com/in/rhydhamjoshi

Contents :

OSI Model and Data Encapsulation

OSI Model and Data Encapsulation





Example showing common protocols used by Windows & Unix OS

Internet Protocol

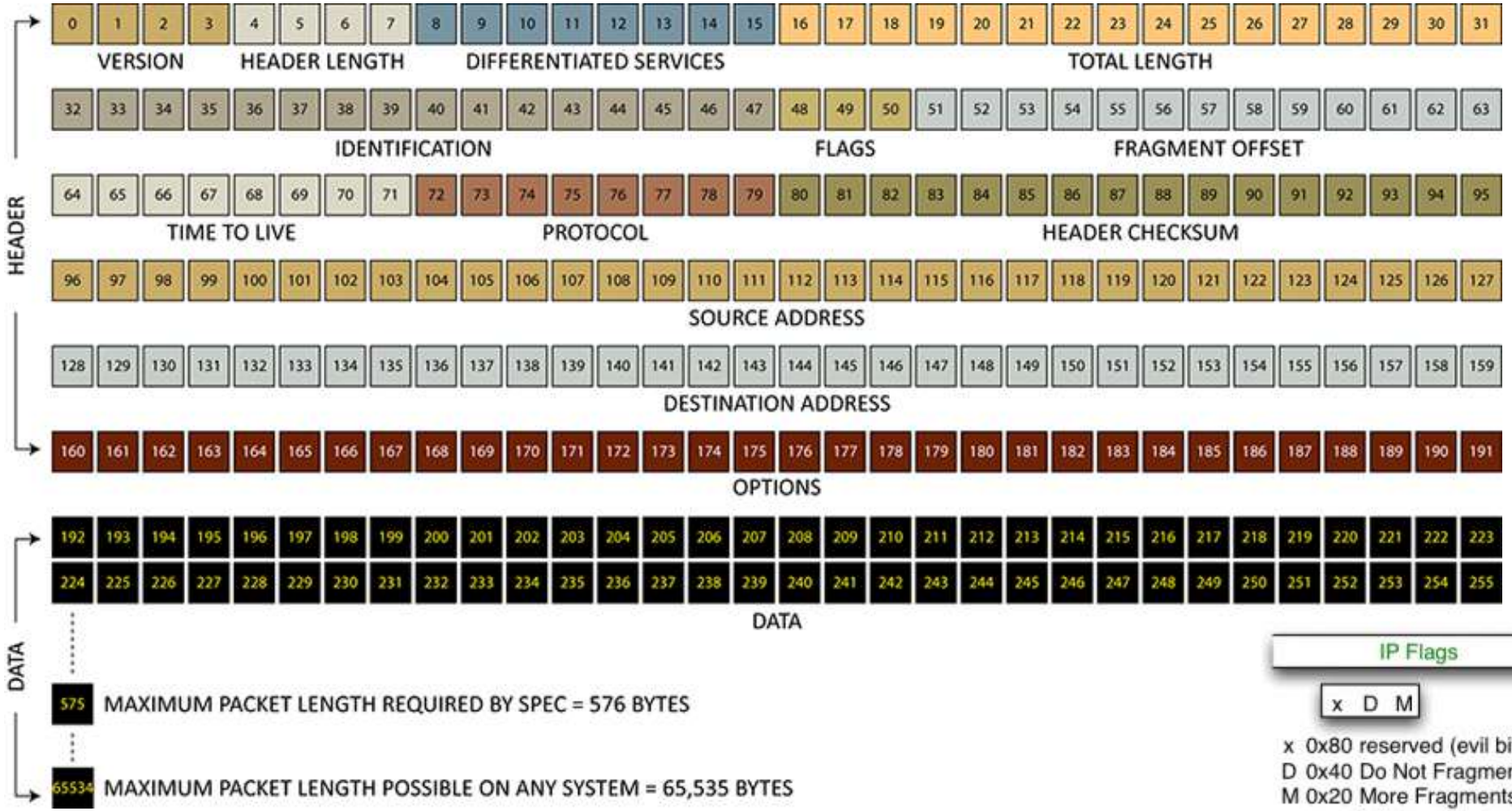
Internet Protocol

- The Internet Protocol (IP) is the principal communications protocol in the Internet protocol suite for relaying datagrams across network boundaries.
- IP delivers packets from the source host to the destination host solely based on the IP addresses in the packet headers.
- IP packet or datagram(a basic transfer unit associated with a packet-switched network) consist of IP Header and Payload(data).
- IP was the connectionless datagram service in the original Transmission Control Program. But since TCP is connection oriented protocol, TCP/IP is often referred as connection oriented protocol.
- Three major classes of routing protocols on IP networks:
  - **Interior gateway protocols - type 1**, link-state routing protocols, such as OSPF and IS-IS
  - **Interior gateway protocols - type 2**, distance-vector routing protocols, such as Routing Information Protocol, RIPv2, IGRP.
  - **Exterior gateway protocols** are routing protocols used on the Internet for exchanging routing information between Autonomous Systems, such as Border Gateway Protocol (BGP), Path Vector Routing Protocol.

IPv4 Packet in brief:

- Traditional point-to-point telecommunications links, simply transmit data as a bit stream.
- A network packet is a formatted unit of data carried by a packet-switched network.
- When data is formatted into packets, the bandwidth of the communication medium can be better shared among users than if the network were circuit switched.
- The Internet Protocol (IP) implements datagram fragmentation, breaking it into smaller pieces, so that packets may be formed that can pass through a link with a smaller maximum transmission unit (MTU) than the original datagram size.
- A packet consists of two kinds of data: **control information** and **user data** (also known as payload).
  - The control information provides data the network needs to deliver the user data, for example: source and destination network addresses, error detection codes, and sequencing information.
  - Control information is found in packet headers and trailers, with payload data in between.
  - Payload refers to actual data that is exchanged between source and destination.

IPv4 Packet



Version

Version of IP Protocol. 4 and 6 are valid. This diagram represents version 4 structure only.

Header Length

Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.

Protocol

IP Protocol ID. Including (but not limited to):

1 ICMP	17 UDP	57 SKIP
2 IGMP	47 GRE	88 EIGRP
6 TCP	50 ESP	89 OSPF
9 IGRP	51 AH	115 L2TP

Total Length

Total length of IP datagram, or IP fragment if fragmented. Measured in Bytes.

Header Checksum

Checksum of entire IP header

Fragment Offset

Fragment offset from start of IP datagram. Measured in 8 byte (2 words, 64 bits) increments. If IP datagram is fragmented, fragment size (Total Length) must be a multiple of 8 bytes.

IP Flags

x D M

x 0x80 reserved (evil bit)
D 0x40 Do Not Fragment
M 0x20 More Fragments follow

RFC 791

Please refer to RFC 791 for the complete Internet Protocol (IP) Specification.

IPv4 Packet

Version	The version of IP used to create the packet. Currently, we use IPv4, so it contains the value 4. This is a 4-bit field, so values of 0 to 15 are allowed.
Header Length	Also called the Internet Header Length (IHL), it simply describes the length of the packet's header, in units of 32-bit words. This is a 4-bit field, so values of 0 to 15 are possible. Since headers must contain all the required fields, the minimum possible header length is 160 bits (5 words). So this field will contain a minimum value of 5. The maximum value is 15, which corresponds to 15 x 32 = 480 bits, or 60 bytes. That is the maximum header size possible. This field can be used as an offset to read the data. Simply multiply the value by 32, count off that many bits from the start of the packet, and start reading data.
Differentiated Services	This 8-bit field was originally known as the TOS (Type of Service) field. It's basically a way for the host to express a preference for how it wants that packet handled - fast and less reliable, slower but more reliable, using a more expensive route, using a cheaper route, etc. It's not been much implemented so far, but will probably be important in IPv6 when there's a lot of real time traffic involved (audio/video stuff). The 8 bits of this field are apportioned as shown below.
	0 - 2 — Precedence, or priority: 3 bits, 8 possible values from very high to very low.
	3 — Delay: two values possible, normal delay or low delay
	4 — Throughput: two values possible, normal or high throughput
	5 — Reliability: two values possible, normal or high reliability
	6 — Cost: two values possible, normal cost or minimize cost
	7 — Undefined
	Some switches may read the information in these bits, and route traffic accordingly. Others may totally ignore this information and handle all packets in exactly the same manner. This is therefore a feature of how a particular network is implemented.
Total Length	This 16-bit field defines the total length of the packet (header + data). The unit is bytes. It can hold values from 0 to 65535 in theory. Legally, the minimum value is 20 (minimum header plus no data). Legally, the maximum must be at least 576, but it can be much larger up to a limit of 65535. This means that the maximum possible size for any IP packet is 65535 bytes, or half a megabit. In practice, it may be much smaller, depending upon the software/hardware generating the packets. However, the maximum allowable must be at least 576 bytes to meet specs.
Identification	This 16-bit field is not always used. Its intended purpose was originally as a unique ID or identifier for different fragments of an IP packet. It's not much used for that purpose today. Some people suggest that it could be used to add information to prevent source address spoofing.

IPv4 Packet

Flags	This is a 3-bit field which contains flags that help manage fragmentation. Fragmentation is described in more detail in a section below, but for now, remember that IP packets can be fragmented if they are too large to go through lower layers (which may have packet size limitations imposed by their own protocol). Therefore, information is needed that helps those layers to fragment these packets. This information is provided via the Flags field and the Fragment Offset field. The 3 bits reserved for the Flags are used as follows.		
		0	Reserved bit. Not used. Must be a 0.
		1	two possible values, 1 means "don't fragment", which lower layers understand as "just trash this packet if it's too big to send without fragmentation". 0 means "okay to fragment".
		2	two possible values: these values are actually set by the lower layers if fragmentation is needed. A value of 1 means "more fragments", meaning that this packet is a fragment of a larger packet, and more packets are to follow. A value of 0 means that this is the last fragment of the series. So if a large IP packet is fragmented into 5 smaller IP packets, the first 4 will have the "more fragments" flag set and the 5th one won't.
Fragment Offset	This 13-bit field tells the host at the receiving end how to re-assemble a packet that was fragmented. The unit is a block of 8 bytes. Since it can have values from 0 to 8191 (13 bits), it can provide a maximum offset of 8191 x 8 = 65528 bytes. This is sufficient to cover the maximum possible length of an IP packet (65535 bytes minus header). This field is again used by lower layers, in case a large IP packet needs to be fragmented. Each fragment is stamped with a fragment offset, which is used to re-assemble the original packet at the other end. For example, a fragment offset of 107 means that the data contained in this fragment belongs to position 107 x 8 = 856, measured from the beginning of the original packet.		
Time to Live (TTL)	This 8-bit field specifies exactly what it says: how long a packet should "live". The units are seconds, so values can be 0 to 255. However, 0 is not a legal value, and any value less than 1 is rounded up to 1. So the legal values are 1 to 255 seconds. The purpose of this field is to prevent a packet from being forwarded forever in a circle, since routes to hosts aren't always known exactly in advance. Every time a packet encounters a switch or router, the TTL is decremented by 1 before it's passed on. When it hits 0, the packet is discarded and no longer forwarded. The switch/router that decrements it to 0 sends an ICMP message back to the sender informing him that the package was discarded. This feature can be used to implement traceroutes.		
Protocol	This 8-bit field defines the protocol used for the data or payload of the packet. It can have values from 0 to 255, each of which specifies a certain Transport Layer protocol. The Internet Assigned Numbers Authority maintains a list which assigns a specific number to a certain protocol. For example, a value of 6 means TCP, or Transport Control Protocol, which is commonly used for the data in TCP packets. A value of 1 means ICMP (Internet Control Message Protocol), which is used for standard messages used for housekeeping/control functions in networks.		
Header Checksum	This is simply a checksum calculated for the header portion of the packet, which is used for error checking. Note that the data is not included in calculating the header checksum. Integrity of the data is the responsibility of the data protocol used. For example, TCP has its own separate checksum for verifying the data integrity. Each switch/router along the route calculates a checksum for the header and compares it against the value in this field. If they don't match, it requests re-transmission of the packet. Note that since each switch changes the header (by decrementing the TTL field), it must calculate and embed a new checksum in each packet before forwarding it.		

IPv4 Packet

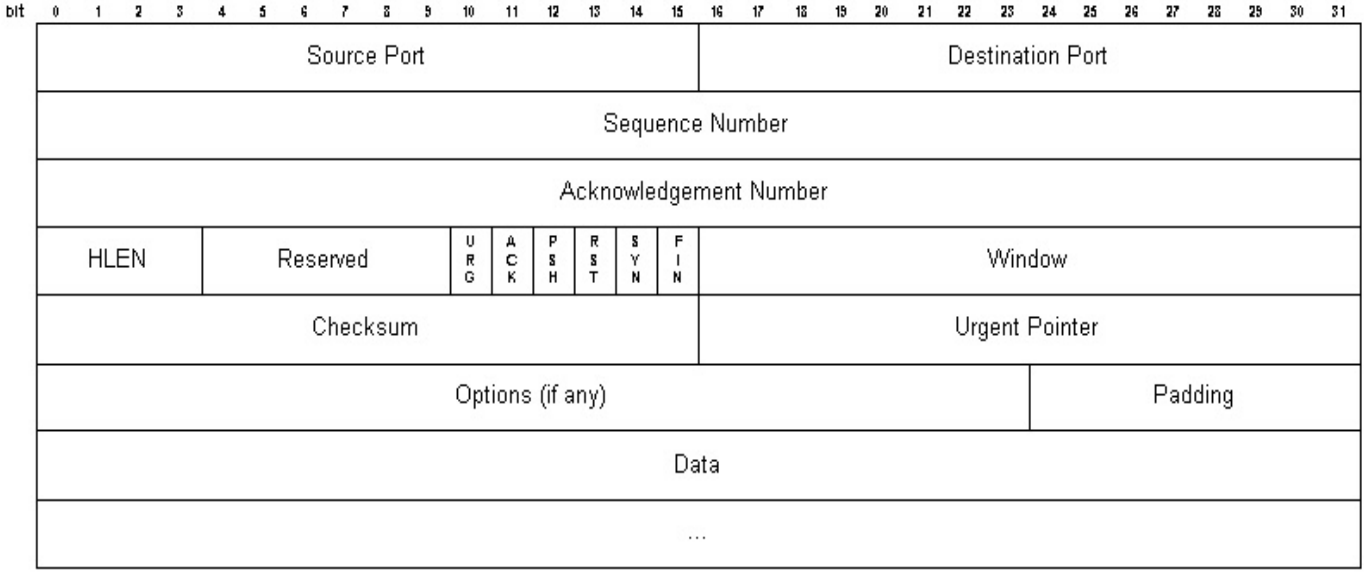
Source Address	This is usually an IPv4 address in binary format. The address field is 32 bits, and can be thought of as a series of four 8-bit fields. Each 8-bit field can contain a value from 0 to 255. Therefore, an IPv4 address of the form 202.134.227.153, for example, can be contained by first converting each of its four parts into binary, then concatenating them together. So it would be: 202 = 11001010; 134 = 10000110; 227 = 11100011; 153 = 10011001. So the value of source address would be obtained by concatenating these four binary numbers: 11001010100001101110001110011001. Note that because of NAT (Network Address Translation) the source address might not be the address of the actual host where the packet originated, but rather the address of the NAT machine. Replies will therefore be sent back to the NAT machine, which will forward them to the appropriate host.		
Destination Address	This is the IPv4 address of the destination machine, again in binary format as described for the Source Address field.		
Options	This is an optional part of the header, and is hardly ever used in IPv4. Since the minimum header length is 20 bytes (with no options) and the maximum header length is 60 bytes (determined by the maximum value possible in the header length field), these options can use up to a maximum of 40 bytes. The first 2 bytes (16 bits) are reserved for the options header, and the remainder for the options data. The bit assignment for the header is shown below.		
	Copied	1 bit	Should options be copied into all fragments if the packet is fragmented. 1 means yes.
	Class	2 bits	4 possible values. Generally used to categorize the options. 0 means "control options" 2 is for "debugging and measurement options", while 1 and 3 are currently not used.
	Number	5 bits	The option number, uniquely identifies each option contained in this part of the header. A maximum of 32 options are possible.
	Length	8 bits	The length of the option in bits. Includes the length of this field as well.
	Data	Variable	Any data used by the options. The length is variable, with the constraint that the total optional part of the header can't be more than 40 bytes. Simple options might not have any data associated with them.
Data	The actual data or payload of the packet. This is not part of the header and not included in the header checksum. The data can be any of the Transport Layer protocols, as defined in the Protocol field of the header. Usually over the Internet, the data will be a Layer 4 packet, either a TCP packet or a UDP packet.		

Transmission Control Protocol (TCP)

Transmission Control Protocol

- Transmission Control Protocol (TCP) is one of the main protocols in TCP/IP networks. Whereas the IP protocol deals only with packets, TCP enables two hosts to establish a connection and exchange streams of data.
- TCP provides reliable, ordered, and error-checked delivery of a stream of octets between applications running on hosts communicating over an IP network.
- TCP guarantees delivery of data and also guarantees that packets will be delivered in the same order in which they were sent.
- TCP adjusts its segment size to be smaller than the MTU(Maximum Transmission unit)
- TCP is the protocol that major Internet applications such as the World Wide Web, email, remote administration and file transfer rely on.
- Applications that do not require reliable data stream service may use the User Datagram Protocol (UDP), which provides a connectionless datagram service that emphasizes reduced latency over reliability.

TCP Header



Source Port

- A 16-bit number identifying the application the TCP segment originated from within the sending host.
- The port numbers are divided into three ranges, well-known ports (0 through 1023), registered ports (1024 through 49151) and private ports (49152 through 65535).
- Port assignments are used by TCP as an interface to the application layer.
- Ex. TELNET server is always assigned to the well-known port 23 by default on TCP hosts. A complete pair of IP addresses (source and destination) plus a complete pair of TCP ports (source and destination) define a single TCP connection that is globally unique.

Destination Port

- A 16-bit number identifying the application the TCP segment is destined for on a receiving host.

TCP Header

Sequence Number

- A 32-bit number identifying the current position of the first data byte in the segment within the entire byte stream for the TCP connection. After reaching 2^{32} -1, this number will wrap around to 0.

Reserved

- A 6-bit field currently unused and reserved for future use.

Control Bits

- *Urgent Pointer (URG)*. If this bit field is set, the receiving TCP should interpret the urgent pointer field (see below).
- *Acknowledgement (ACK)*. If this bit field is set, the acknowledgement field described earlier is valid.
- *Push Function (PSH)*. If this bit field is set, the receiver should deliver this segment to the receiving application as soon as possible. An example of its use may be to send a Control-BREAK request to an application, which can jump ahead of queued data.
- *Reset the Connection (RST)*. If this bit is present, it signals the receiver that the sender is aborting the connection and all queued data and allocated buffers for the connection can be freely relinquished.
- *Synchronize (SYN)*. When present, this bit field signifies that sender is attempting to "synchronize" sequence numbers. This bit is used during the initial stages of connection establishment between a sender and receiver.
- *No More Data from Sender (FIN)*. If set, this bit field tells the receiver that the sender has reached the end of its byte stream for the current TCP connection.

TCP Header

Acknowledgement Number

- A 32-bit number identifying the next data byte the sender expects from the receiver. Therefore, the number will be one greater than the most recently received data byte. This field is only used when the ACK control bit is turned on.

Header Length

- A 4-bit field that specifies the total TCP header length in 32-bit words (or in multiples of 4 bytes if you prefer). Without options, a TCP header is always 20 bytes in length. The largest a TCP header may be is 60 bytes. This field is required because the size of the options field(s) cannot be determined in advance. Note that this field is called "data offset" in the official TCP standard, but header length is more commonly used.

Window

- A 16-bit integer used by TCP for flow control in the form of a data transmission window size. This number tells the sender how much data the receiver is willing to accept. The maximum value for this field would limit the window size to 65,535 bytes, however a "window scale" option can be used to make use of even larger windows.

Checksum

- A TCP sender computes a value based on the contents of the TCP header and data fields. This 16-bit value will be compared with the value the receiver generates using the same computation. If the values match, the receiver can be very confident that the segment arrived intact.

TCP Header

Urgent Pointer

- In certain circumstances, it may be necessary for a TCP sender to notify the receiver of urgent data that should be processed by the receiving application as soon as possible. This 16-bit field tells the receiver when the last byte of urgent data in the segment ends.

Options

- In order to provide additional functionality, several optional parameters may be used between a TCP sender and receiver. Depending on the option(s) used, the length of this field will vary in size, but it cannot be larger than 40 bytes due to the size of the header length field (4 bits). The most common option is the maximum segment size (MSS) option. A TCP receiver tells the TCP sender the maximum segment size it is willing to accept through the use of this option. Other options are often used for various flow control and congestion control techniques.

Padding

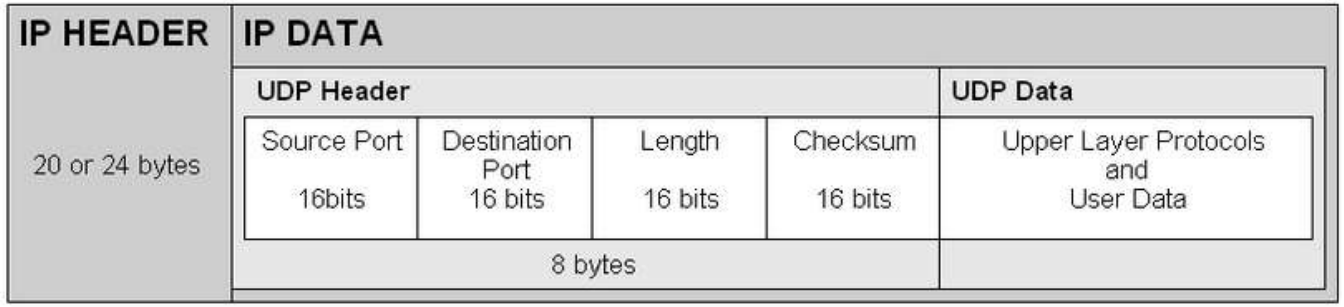
- Because options may vary in size, it may be necessary to "pad" the TCP header with zeroes so that the segment ends on a 32-bit word boundary as defined by the standard [10].

Data

- Although not used in some circumstances (e.g. acknowledgement segments with no data in the reverse direction), this variable length field carries the application data from TCP sender to receiver. This field coupled with the TCP header fields constitutes a TCP segment.

User Datagram Protocol (UDP)

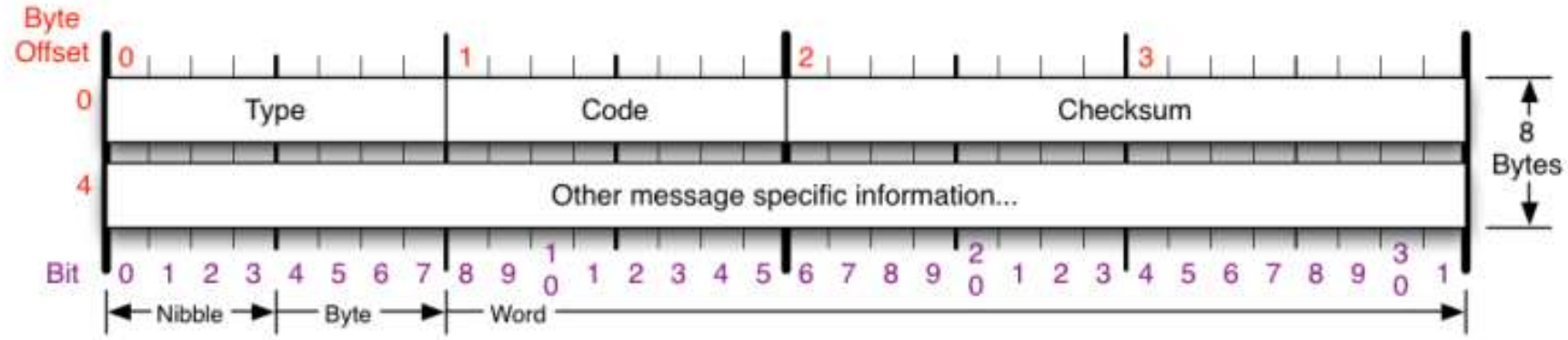
User Datagram Protocol



- UDP (User Datagram Protocol) uses Internet Protocol(IP) to exchange messages between computers in a network.
- UDP uses a simple connectionless transmission model.
- There is no guarantee of delivery, ordering, or duplicate protection.
- UDP provides checksums for data integrity, and port numbers for addressing different functions at the source and destination of the datagram.
- It sents message without prior communications to set up special transmission channels or data paths.
- Unlike TCP, however, UDP does not provide the service of dividing a message into packets (datagrams) and reassembling it at the other end.
- The Trivial File Transfer Protocol (TFTP) uses UDP instead of TCP.

Internet Control Message Protocol (ICMP)

ICMP Header



ICMP Message Types

Type	Code/Name
0	Echo Reply
3	Destination Unreachable
	0 Net Unreachable
	1 Host Unreachable
	2 Protocol Unreachable
	3 Port Unreachable
	4 Fragmentation required, and DF set
	5 Source Route Failed
	6 Destination Network Unknown
	7 Destination Host Unknown
	8 Source Host Isolated
	9 Network Administratively Prohibited
	10 Host Administratively Prohibited
	11 Network Unreachable for TOS

Type	Code/Name
3	Destination Unreachable (continued)
	12 Host Unreachable for TOS
	13 Communication Administratively Prohibited
4	Source Quench
5	Redirect
	0 Redirect Datagram for the Network
	1 Redirect Datagram for the Host
	2 Redirect Datagram for the TOS & Network
	3 Redirect Datagram for the TOS & Host
8	Echo
9	Router Advertisement
10	Router Selection

Type	Code/Name
11	Time Exceded
	0 TTL Exceeded
	1 Fragment Reassembly Time Exceeded
12	Parameter Problem
	0 Pointer Problem
	1 Missing a Required Operand
	2 Bad Length
13	Timestamp
14	Timestamp Reply
15	Information Request
16	Information Reply
17	Address Mask Request
18	Address Mask Reply
30	Traceroute

Checksum

Checksum of ICMP header

RFC 792

Please refer to RFC 792 for the Internet Control Message protocol (ICMP) specification.

- The **Internet Control Message Protocol** (**ICMP**) is one of the main protocols of the Internet Protocol Suite. It is used by network devices, like routers, to send error messages indicating. ICMP has no port numbers; it uses ICMP message types and codes instead.
- ICMP is used to pass an error message between two hosts or a host and a network device such as a router. Since UDP and IP are connectionless protocols, they rely on ICMP to transmit error messages on their behalf.

Anomalies in Datagrams

Anomalies in Datagrams

- Abnormal packets are those which violate IP Protocol standards.
- Abnormal packets may be generated through benign means, such as a malfunctioning router, but they are usually specially crafted by attackers.
- The abnormality is often introduced into the packet purposely so that the packet may avoid being blocked by a firewall or intrusion detection system.
- In other cases, abnormal packets are used to attempt to crash systems by exploring few vulnerabilities in O.S.
- There are dozens of IP protocols including common IP protocols like Transmission Control Protocol(TCP),User Datagram Protocol(UDP), Internet Control Message Protocol(ICMP) such as IGRP, EIGRP and OSPF used for data transmission.
- Each IP protocol type has its own value, called an Internet Protocol number. Some systems log packets by the IP number.
- List of IP Numbers associated with protocols could be found at : http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml
- IP Numbers associated with some common protocols :

  TCP (6) , UDP(17) and ICMP(1)
- This slide describes anomalies associated with TCP, UDP and ICMP Protocol.

Anomalies in Internet Protocol (IP) Packet

Anomalies in Internet Protocol (IP) Packet

- For IP packets, since version 6 is yet to be deployed, Any packet with Version!=4 can be considered as crafted.
- IP Spoofing attacks are made to investigation harder for an attack to be traced back to its real IP address. Attackers often use IP address belonging to other domain and sometimes even Private IP address range.
- **Land IP Spoofing Attack** refers to use of packets with the source and destination address set to the same value. Such packets has to be rejected right away.
- IP spoofing is very limitative since it "looks weird“ and most of the time it gets dropped by ISP only.
  - E.g. :If one of the ISP customer sends a packet with a spoofed external address, this will show up as an outgoing packet, coming from the inside, but with an external source address; the ISP routers will find such an occurrence anomalous and may report it.
  - The ISP can then further track down such packets down its infrastructure and thus pinpoint the culprit.

Anomalies in TCP Segment

Anomalies in TCP Segment

- TCP is a connection-oriented protocol; it uses various flags to indicate that a connection is being started or ended, or that the data carries a high priority.
- Many attacks are based on altering the TCP flags. Certain illegal combinations of TCP flags may be able to help packets avoid detection by firewalls or intrusion detection systems; other illegal combinations may be used to crash operating systems.
- At least one of these six flags must be set in each TCP packet; each flag corresponds to a particular bit in the TCP header.
- The six flags are:
  - SYN (Synchronization) - Initiate a TCP connection.
  - ACK (Acknowledgment) - Indicates that the value in the acknowledgment number field is valid.
  - FIN (Finish) - Gracefully end a TCP connection.
  - RST (Reset) - Immediately end a TCP connection.
  - PSH (Push) - Tells the receiver to pass on the data as soon as possible.
  - URG (Urgent) - Indicates that the urgent pointer is valid; often caused by an interrupt.

Note : The functional specification for TCP is defined in RFC 793. This RFC and others define how systems should respond to legitimate packets, but they don't explain how systems should handle illegal combinations of flags. Consequently, different operating systems respond differently to illegal flag combinations. Attackers can exploit this to determine what operating system a device is using.

Anomalies in TCP Segment

Normal Flag combination:

- SYN, SYN ACK, and ACK are used during the three-way handshake which establishes a TCP connection.
- Except for the initial SYN packet, every packet in a connection must have the ACK bit set.
- FIN ACK and ACK are used during the graceful teardown of an existing connection. PSH FIN ACK may also be seen at the beginning of a graceful teardown.
- RST or RST ACK can be used to immediately terminate an existing connection.

- Packets during the "conversation" portion of the connection (after the three-way handshake but before the teardown or termination) contain just an ACK by default. Optionally, they may also contain PSH and/or URG.
- Packets with any other flag combination can be classified as abnormal.

Anomalies in TCP Segment

Common Abnormal Flag combinations:

- SYN FIN is probably the best known illegal combination. It is nonsensical to perform start and end of an existing connection, both at the same time. Many scanning tools use SYN FIN packets, because many intrusion detection systems did not catch these in the past, although most do so now.
- SYN FIN PSH, SYN FIN RST, SYN FIN RST PSH, and other variants on SYN FIN also exist and they are clearly malicious.. These packets may be used by attackers who are aware that IPS/IDS may be looking for packets with just the SYN and FIN bits set, not additional bits set.
- Packets should never contain just a FIN flag. FIN packets are frequently used for port scans, network mapping and other stealth activities.
- Some packets have absolutely no flags set at all; these are referred to as "null" packets. It is illegal to have a packet with no flags set.
- TCP packets have two additional bits referred as“reserved bits” which are reserved for future use. Any packet which has either or both of the reserved bits activated is almost certainly crafted.

Anomalies in TCP Segment

Other common characteristics of abnormal TCP traffic:

- Packets should never have a source or destination port set to 0.
- The acknowledgment number should never be set to 0 when the ACK flag is set.
- A SYN only packet should only occur when a new connection is being initiated and it should not contain any data.
- Packets should not use a destination address that is a broadcast address, usually ending in .0 or .255. Broadcasts are normally not performed usin

Anomalies in UDP Segments

Anomalies in UDP Segment

- Unlike TCP, UDP is a connectionless protocol. UDP does not have the flag and reserved bits that TCP does. However, both TCP and UDP rely on source and destination ports.

- Like TCP, packets, UDP packets should never have a source or destination port set to 0. UDP packets can also be fragmented maliciously.

Anomalies in ICMP Packets

Anomalies in ICMP Packets

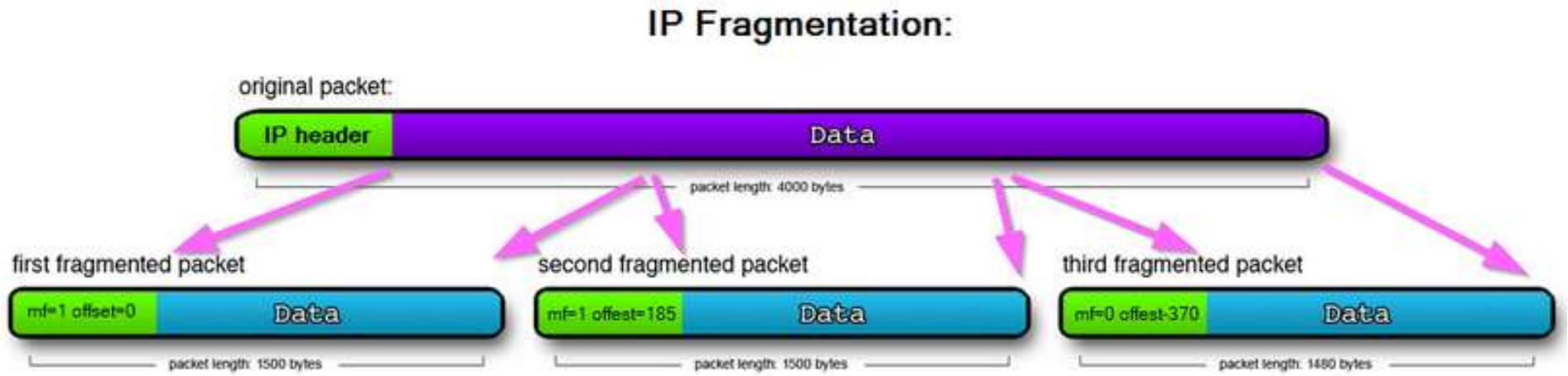
- ICMP redirect messages are intended to be sent from a router to a host to inform host that different router is more optimal than it is when contacting a particular destination address. Some attacks such as WinFreeze use false ICMP redirect messages to attempt to convince a host to use itself as the optimal router. Any packet which tells a device to route everything to itself should be considered highly abnormal.
- Most ICMP packets are composed of a small header and payload; for example, most ICMP echo request packets have an 8-byte header and a 56-byte payload. ICMP packets that are significantly larger than normal should be considered suspicious.
- Also, some ICMP types, such as echo requests, should not be carrying any data.
- Some malicious applications, including various DDOS programs and tunneling programs, use ICMP packets as "containers" that hide other traffic. So an ICMP echo reply might actually contain a completely different IP protocol within its data. If large ICMP packets or for packets of specific ICMP types that should not contain data but do contain is detected, it has to be considered malicious.

Anomalies due to Fragmentation

Anomalies due to Fragmentation:

Fragmentation:

- When an IP packet is too large to be transmitted as one entity, it must be split into two or more smaller pieces that can be sent across networks. Each piece of a packet is referred to as a fragment. Fragmentation occurs for all of the protocols like TCP, UDP and ICMP - although it occurs most frequently for TCP.
- The common packet filtering approach used by filtering systems to deal with fragmentation is to do packet filtering only on the first fragment of a packet and to allow any non-first fragments through. Packet can then be re-assembled using fragmented packets.



Common Exploits due to Fragmentation:

IP fragment overlapped

- Some operating systems do not properly handle fragments that overlap and may throw exceptions or behave in other undesirable ways upon receipt of overlapping fragments.
- **Teardrop attack**: Overlapping fragments may also be used in an attempt to bypass Intrusion Detection Systems. In this exploit, part of an attack is sent in fragments along with additional random data; future fragments may overwrite the random data with the remainder of the attack. If the completed datagram is not properly reassembled at the IDS, the attack will go undetected.

IP fragmentation buffer full

- The IP fragmentation buffer full exploit occurs when there is an excessive amount of incomplete fragmented traffic detected on the protected network. This could be due to an excessive number of incomplete fragmented datagrams, a large number of fragments for individual datagrams or a combination of quantity of incomplete datagrams and size/number of fragments in each datagram.

IP fragment overrun

- The IP Fragment Overrun exploit is when a reassembled fragmented datagram exceeds the declared IP data length or the maximum datagram length. By definition, no IP datagram should be larger than 65,535 bytes. Systems that try to process these large datagrams can crash, and can be indicative of a denial of service attempt.

Common Exploits due to Fragmentation:

IP fragment too many datagrams

- The Too Many Datagrams exploit is identified by an excessive number of incomplete fragmented datagrams detected on the network. This is usually either a denial of service attack or an attempt to bypass security measures. An example of "Too Many Datagrams", "Incomplete Datagram" and "Fragment Too Small" is the Rose Attack.

IP fragment incomplete datagram

- This exploit occurs when a datagram can not be fully reassembled due to missing data. This can indicate a denial of service attack or an attempt to defeat packet filter security policies.

IP Fragment Too Small

- If an IP fragment is too small it indicates that the fragment is likely intentionally crafted. Any fragment other than the final fragment that is less than 400 bytes could be considered too small. Small fragments may be used in denial of service attacks or in an attempt to bypass security measures or detection.

References:

- **OSI Model and Data Encapsulation : (http://www.apkmodgame.net/tag/seven-layers-of-osi-model-and-functions-of-seven-layers-of-)**
- **Internet Protocol (http://www.wikipedia.com)**
- **IPv4 Packet (http://essayweb.net/miscellany/datatransmission.shtml)**
- **TCP (http://www.wikipedia.com)**
- **Packet Anomalies (http://www.symantec.com/connect/articles/abnormal-ip-packets)**
- **Fragmentation Anomalies (http://docstore.mik.ua/orelly/networking_2ndEd/fire/ch04_02.htm)**

Thank you.!