Packet Reassembly & Overlapping IP Fragments

By **Haboob Team**

Packet Reassembly & Overlapping IP Fragments

Table of Contents

Introduction

This Paper will discuss how intrusion detection systems work. After getting a solid understanding of the working mechanism of IDS, we will discuss how packet reassembly works. Then will move forward to look into different policy implemented for packet reassembly where it is dependent on the operating system implementation of the RFC. Also, will explain how packet get fragment on the IP level not the TCP and will look into an example.

What is IDS?

An intrusion detection system IDS comes as a device or an extendable module that can be attached into Next Generation Firewall or even as a plain software running on your custom hardware of choice. Intrusion Detection Systems monitors network traffic for malicious behavior or violation to an established network policy. Violation alerts or evasion activity are reported to the network security administrators where admins choose what violation to report. In some organizations those violations are reported to the security information and event management system "SIEM "correlating intrusion from multiple log sources across your infrastructure. You can create a correlation rule in you SIEM system to monitor for a certain series of activities which can be a possible indication of compromise "IOC".

Intrusion Detection Systems come in multiple types ranging in scope from one single computer device to multiple large cluster of network devices. Intrusion Detection System can be deployed as Network Intrusion Detection System "NIDS" or event as Host-based Intrusion Detection System "HIDS". A system that monitor intrusion to a single workstation machine and deployed in the same machine is a clear example of HIDS. A deployment of multiple network system that monitor network activity in a standalone monitoring device is a clear example of NIDS. Intrusion Detection System are classified or can be broken down into the following classification. First, it can be classified based on detection approach. Most of the time are categorized as a signature based which involves having a pattern of strings or passive IOCs. Second, anomaly based detection detecting any deviation of a normal or what considered as a good traffic. Anomaly based detection will need to consume more computer resources to detecting any malicious or abnormal network activity with the help of machine learning. Third, reputation-based detection where it uses what it known to be a real threat and build a scoring mechanism to classify the reputation levels. There are IDS products that can take action when detecting a violation to our established network policy and will prevent/block that traffic. We call those systems that have the ability to take action and Intrusion Prevention System.

IDS vs Firewall

Both devices are network security equipment that monitor network traffic activity, But Intrusion Detection System is different from a firewall in a since that firewall does not have the preprocessing capability as in snort for example. We have multiple preprocessing engines that can simulate end system behavior to find a possible violation that is occurring in the network environment. Modern Next Generation Firewall integrate those functionalities in the same device equipment. Noting the preprocessing engine when enabled will consume too much hardware resources and need to have high hardware specs to keep up with the huge network traffic.

How IDS work

Intrusion Detection Systems come in multiple types ranging in scope from one single computer device to multiple large clusters of network devices. IDS systems can use different methods for detecting suspected intrusions. The two most common broad categories are by pattern matching and detection of statistical anomalies.

Signature Based Pattern Detection

Pattern matching is used to detect known attacks by their "signatures," or the specific actions that they perform. It is also known as signature-based IDS or misuse detection. The IDS looks for traffic and behavior that matches the patterns of known attacks. The effectiveness is dependent on the signature database, which must be kept up to date.

Pattern matching is analogous to identifying a criminal who committed a particular crime by finding his fingerprint at the scene. Fingerprint analysis is a type of pattern matching. The biggest problem with pattern matching is that it fails to catch new attacks for which the software doesn't have a defined signature in its database.

Anomaly Based Detection

Anomaly-based detection watches for deviations from normal usage patterns. This requires first establishing a baseline profile to determine what the norm is, then monitoring for actions that are outside of those normal parameters. This allows you to catch new intrusions or attacks that don't yet have a known signature.

Anomaly detection is analogous to a police officer who walks or drives a particular beat every day and knows what is "normal" for that area. When he sees something that's out of the ordinary, it creates reasonable suspicion that criminal activity may be going on, even though he may not know exactly what crime is being committed or who is responsible.

There are several different anomaly detection methods, including:

- Metric model
- Neural network
- Machine learning classification

A problem with anomaly-based IDS is the higher incidence of false positives, because behavior that is unusual will be flagged as a possible attack even if it's not.

IP Fragmentation

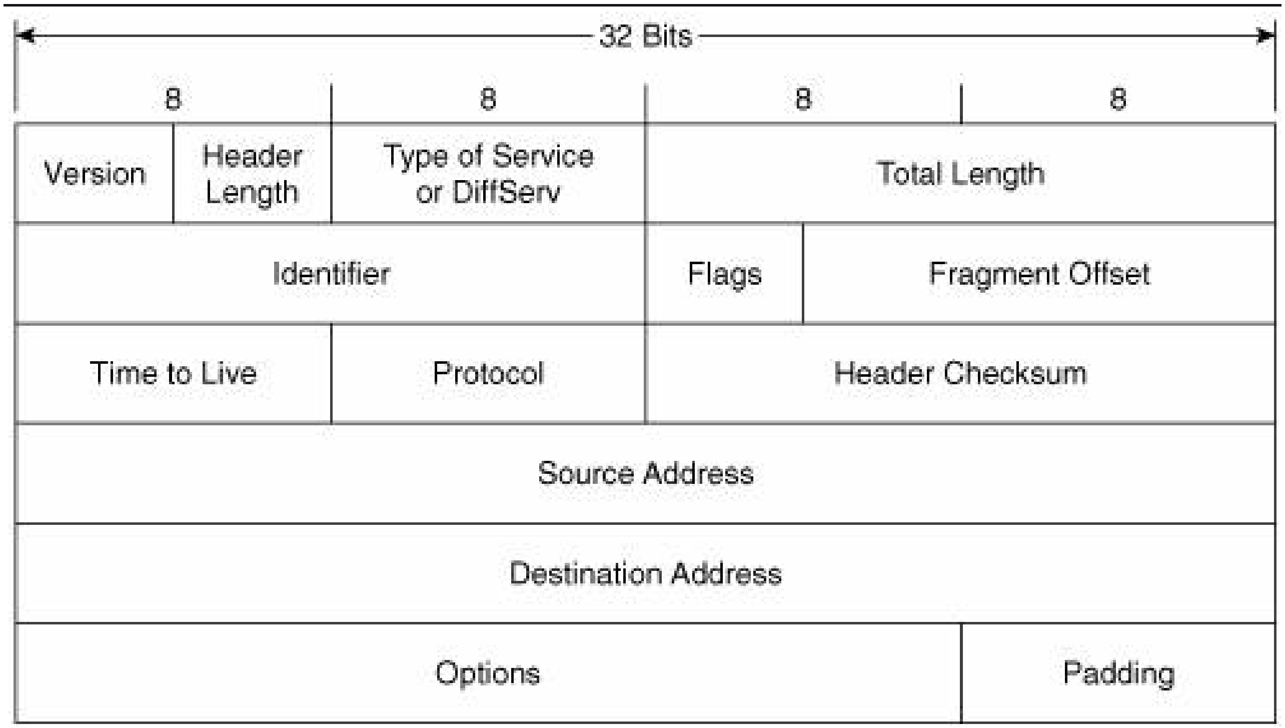
When a packet is being sent to a destination it should be fragmented according to MTU Maximum Transmission Units mostly is set to 1500 Byte MTUs. When packet is sent it will go through different routing devices each device has their own MTUs set up and a packet must not exceed the lower MTU if so it will be fragmented again to match the lower MTU in its path. The following will show the field used to control fragmentation:

Identification: every packet has a unique ID among the fragmented packets

Flags: this field used to control packet fragmentation. For example:

a. Bit 0: Reserved
b. Bit 1: Do not Fragment “DF”
c. Bit 2: More Fragments is coming “MF”

Fragment Offset: Used to reassemble packet regardless if arrived in order or not.



32 Bits

8		8		8		8	
Version	Header Length	Type of Service or DiffServ		Total Length			
Identifier				Flags	Fragment Offset		
Time to Live		Protocol		Header Checksum			
Source Address							
Destination Address							
Options						Padding	

Figure 1: IPv4 Header

Example of packet Fragmentation

Let us say that there is 1440-byte packet that need to be routed through 576 MTU. Taking into consideration that IP header is 20 bytes meaning the original packet contain 1420 Byte. Then how will the packet be fragmented?

The first Fragment:

- ID will be shared among all the packets
- Flag will be set to MF
- Total length will be 572

The second Fragment:

- ID will be shared among all the packets
- Flag will be set to MF
- Total length will be 572

The third Fragment:

- ID will be shared among all the packets
- Flag will be set to DF
- Total length will be 336

Fragment	Identification	Flags	Fragment Offset	Total Length
First Fragment	1	1 (More Fragment)	0	572
Second Fragment	1	1 (More Fragment)	69	572
Third Fragment	1	0 (Do not Fragment)	138	336

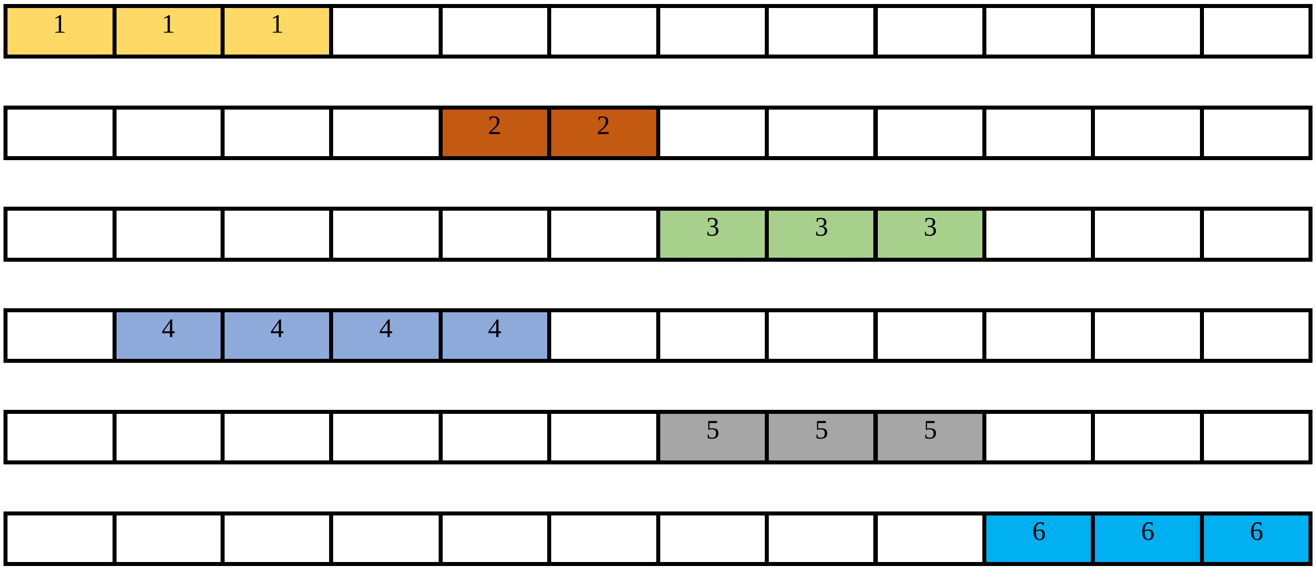
Figure2: This Table shows the resulted fragments of the previous example

We can verify the previous fragmentation result by calculating the total length of the all fragment and then subtracting the IP header from each

(572 – 20) + (572 – 20) + (336 – 20) = 1420

How Operating Systems Reassemble Packets

Intrusion Detection Systems are not giving accurate result and that because they are not processing and analyzing packets the same as the operating system that receives those packets. This is a problem which found in multiple parts of the packet header such as IP header, transport header, several layers of evaluation and processing of the packets including the IP, protocol, and application layers. As an example of the problem, consider traffic that has overlapping fragments that are sent to a given host. Because every operating system reassembles packets differently, depending on the policy that the operating system uses. An Intrusion Detection System cannot use only one policy for all, and if that so it will not be able to reassemble the packets as the operating system does. The adversary can take advantage of that to evade and bypass the IDS because of the mismatch between the two. Or even can manipulate the network analyst



1	1	1									
				2	2						
						3	3	3			
	4	4	4	4							
						5	5	5			
									6	6	6

Figure3: this is an example of a packet that will be sent in order

The following figure shows how each operating system reassemble packet differently and not following the written RFC because of the different interpretation of the RFC. For Example, some implementation will favor the first arrived packet where other operating systems favors the last arrived packet for an offset.

BSD “FreeBSD”

1	1	1	4	4	2	3	3	3	6	6	6

BSD-RIGHT “HP Printer”

1	4	4	4	2	2	5	5	5	6	6	6

FIRST “Windows / MacOS”

1	1	1	4	2	2	3	3	3	6	6	6

LINUX

1	1	1	4	4	2	5	5	5	6	6	6

LAST “CiscoIOS”

1	4	4	4	4	2	5	5	5	6	6	6

Figure4: How would each operating system reassemble fragmented packets

Comparison between Operating System Reassembly Mechanism

BSD policy

BSD policy left-trims an incoming fragment to existing fragments with a lower or equal offset, discarding it if it is overlapped entirely by existing fragments. All remaining octets are accepted; overlapping fragments with a greater offset are discarded or trimmed accordingly. This policy is documented more thoroughly in Wright and Stevens. This Policy Used by FreeBSD, AIX, SUN, IRIX, OpenVMS and Wireshark.

BSD-right Policy

This policy is similar to BSD, except fragments are right-trimmed (new fragments take precedence over those with a lower or equal offset). This Policy Used by HP JetDirect (Printer).

Linux Policy

The Linux policy is almost the same as the BSD policy, except that incoming fragments are trimmed only to existing fragments with a strictly lower offset; that is, existing fragments with the same offset will be overwritten, at least in part. This policy Used by Linux OS, OpenBSD.

First Policy

Always accept the first value received for each offset in the packet. This policy used by Windows, MacOS, SUN, HP-UX, Tektrnoix Phaser Printer.

Last/RFC791 Policy

Always take the last value received for each offset in the packet. This policy used by Cisco IOS, some model from Tektrnoix Phaser Printer.

Mitigation

1- Some Modern IDS system includes Fragmentation Protection but it is disabled by default. Even though some IDS include Fragmentation Preprocessor, it is still plausible that the analyst be deceived by an adversary when sending two exploits for two different operating systems

2- You have to configure your IDS to support Fragmentation Preprocessing.

3- You have to make your IDS speaks the operating system language.

4- Train your network analyst to get a solid understanding on how reassembly engine works.

5- For Snort enable Frag3 and configuring/Fine-tuning it to reassemble packets as target based IP defragmentation.

6- Other modern IDS let you choose between IP Reassemble modes.

Conclusion

Every operating system reassemble packets differently, depending on the policy that the operating system uses. And Intrusion Detection System cannot use only one policy for all, and if that so it will not be able to reassemble the packets as the operating system does. The adversary can take advantage of that to evade and bypass the IDS because of the mismatch between the two or even can manipulate the network analyst. To tackle this challenge, we need to implement what is called target based detection. Identifying your infrastructure assets is essential in implementing protection mechanism properly.

References

[1] https://en.wikipedia.org/wiki/Intrusion_detection_system

[2] https://resources.infosecinstitute.com/network-design-firewall-idsips/#:~:text=Though%20they%20both%20relate%20to,attack%20from%20inside%20the%20network.

[3] https://www.techrepublic.com/article/solutionbase-understanding-how-an-intrusion-detection-system-ids-works/#:~:text=Pattern%20matching,the%20patterns%20of%20known%20attacks.

[4] https://tools.ietf.org/html/rfc791

[5] https://packetpushers.net/ip-fragmentation-in-detail/

[6] http://www.snort.org/assets/165/target_based_frag.pdf

[7] https://digitalassets.lib.berkeley.edu/techreports/ucb/text/CSD-03-1246.pdf

[8] https://forum.netgate.com/topic/72043/snort-spp_frag3-fragmentation-overlap-again-and-again-and-again