0 4 8	16 19	32
Vesion	IHL	ToS	Total len
Identifier		Flags	Fragment offset
TTL	Protocol	Header Check sum
Source IP Address
Destination IP Address
Options
Data
(Payload)

base header (20 byte)

بکار IPV4

برای frag , reassembling

استفاده میشه

< 60 Byte

> 20 Byte

قابلیت های برای Fragmentation برای آنکه اپلیکیشن است که بتونیم بسته و نتیجه مربوطه

multiplexing یا

- Version : نسخه IP که IPv4 یا IPv6

- Internet Header Length (IHL) : طول header بر اساس تعداد

(راوار = کلان 32bit) مقدار = $15 \times 4 = 60$

$2^4 - 1$

- Type of Service (ToS): داده ایی که بخواهیم اولویت سرویس مربوط رو اضافه کنیم به سرویس (کیفیت)

(کیفیت) یک نوع : delay , یک نوع اهمیت packet loss , یک نوع (کیفیت)

Cost   Throughput   Reliability   Delay

X P3 P1 P0 DRTC

نوع سرویس ادویت

در غالب این فیلد استفاده میشود چون اینکه غالب به یک routing table داشته باشیم بس از این فیلد برای

$2^8 = 256$ (یک فیلد) Class of Service

برای می تواند میشه در نظر گرفته باشه ولی اینکه خاص سرویس کردم بار آلکی رو بزاریم

- total length _____ اندازه کل سر بسته data , header (شامل

(واحد = byte) (byte = $2^{16}-1$ = حداکثر اندازه بسته)

64 k Byte

- ___ برای خط مشی دلایل reassembling , frag و سرهم بندی مجدد

- TTL (time to live): loop جلوگیری از ماندن بسته در شبکه و routing مشکل
سرشت در این حالت اگر TTL این بسته منتقل می شود از این پروتکل
توسط ساخته

اول برای این زمان بود ولی بعدا تعریف داده شده براساس بعداد گام یا hops (hop limit)
معمولا 64 یا 128
همواره گام به گام عبور می کند مقدار TTL کم می شود

- Protocal: مشخص می سازد که گروه از چه پروتکلی این داده دریافت می کنم
میتوانیم با پروتکل IP کار کنیم

TCP → 6   UDP → 17

ICMP → 1   IPsec → 4

IP پروتکل میزبان

- header checksum . مثلا به خطای ممکن بودن IP check جلوگیری
sum
از خطای میزبان ___ (فقط جلوگیری header خطای بارگذاری)

- Source IP Address
دستگاه مبدا خطاب به کلاس __ reply برای پاسخ، پاکت ها و داده
یا ___ پیام را پردازش می کنیم و ___ در این ذخیره می کنیم

- Destination IP Address
آن ___ مقصد ___

<u>IP Address</u> → 32 bit   8bit, 8bit, 8bit
dot - decimal 100 . 50 . 10 . 1

کنیم چون حتی زمانی می خواهیم از آدرس دهی سلسله مراتبی استفاده کنیم. مثلاً سرا اگر می خواهیم ایراد معنی دار به forwarding table بدهیم، سنگ آدمکش شویم

Fragmentation & Reassembling

ethernet به زیرشبکه
MTU = 1500 Byte

ADSL فرض
MTU = abc Byte

SONET مثلاً مرزی
MTU = dek Byte

هرشبکه یک ربط (max transfer unit) MTU

fragmentation مثلاً وقتی بزرگتر میاد و می رساند به شبکه هایی که اندازه [MTU] کمتری دارند: reassembling
در گره مقصد [MTU] یکی می شود

گره مبدأ می تواند fragment کند مربوط به های ... به بسته ها را جمع به ترتیب بیند دوباره (جون IP)
ترتیب حفظ نمیشه (یکسان) بسته بازسازی کند.

سه تا ابدیندکس ها کمک می کند باشناسایی با درست شدن frag ها مشخص شود که این frag از کجا اومده.
یک identifire کنار هرباکت
identifier ← 16 bit ← 64k دوباره
چون از 64k زیاد ← بصورت
دوبه سرار برسیم سری کوار می شود
اول رسیده ← هرباکتی id یکه unique مشخصی
جزو هابلانندکس دوباره ← frag یک جون frag مشخص
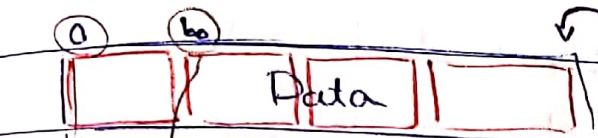
16 bit offset

64K ا بایت

Data

offset هر داده ای بی در هر frag مرکز می‌شم

می‌کنیم با offset ماند header ۱۳ بیتی است ⟵ ۳bit جزء اولین مینیمم

هر اندازه هر frag مربوط به اضرب ۸ سیستم مینیمم (⟵ ۱۳bit اول

frag هشتمین صفر است (

از ۱۳ بیت می‌مونیم frag ۱۶K را که تش‌ها S

Flags ⟶ MF سیت = 0   برای سیم آخری

(more frag)   ↳ سیم اول میانی‌ها = 1

نمونه‌ای frag ۲ می‌باید در مبدل بعد از S می‌شن از هر گرفتیم دل چیزی که برشته مینیم
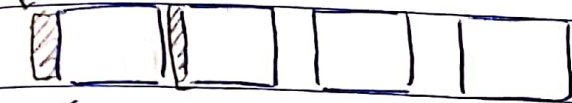
| H | Data | length = 1504   ID = X

(X) فاینتیک

MTU = 576
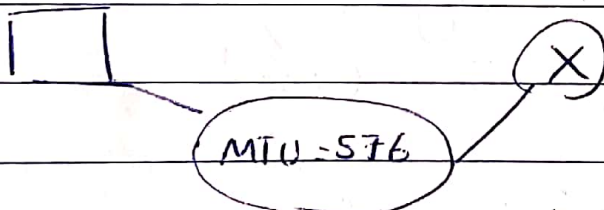
1504 - 20 = 1484 ⟶ Data

576 - 20 = 556 ⟶ Data های frag

556 ⟶ سیت ۸ است ⟹ 556 مضرب ۸ نیست می‌بریم   552 کوچک‌ترین مضرب ۸ صفر است

استاندازه data دینار می‌کنیم

1484	552
1104	2
380	

	total len	ID	MF	offset	
original packet	1504	x	0	0	سرآمد بسته
					تعداد Frag شده
Frag 1	552 + 20	x	1	0	
Frag 2	572	x	1	$552 \div 8 = 69$	
Frag 3	380 + 20	x	0	138	

سرویس در Flag که شامل تعدادی از اسلایدهای استفاده یعنی MF و یکی DF

DF (Don't Fragment)   اگر مسیر این بسته که کسر که تونست این سیستمها رو تونست این
کرد Frag   اگر MTU کم تر بود سم lost می شه با پروتکل ICMP
خطا به مبدا اطلاع داده می شود

$4 \times 10^9 = 2^{32}$

32 سری که در سرم

جستجو و آدرس کردن جدول با 2^{32} ردیف سخت و زمان بر است!

راهی از روش آدرس دهی سلسله مراتبی این مقاله می کنم   XX00

آدرس دهی سلسله مراتبی:
host کلاس ← 4 بیت
آدرس
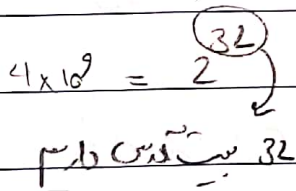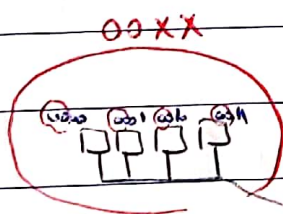
Forwarding table R1   درواقع آدرس شبکه است

م س معتبر	N.H
0 0 0 0	1
0 0 0 1	1
0 0 1 0	1
0 0 1 1	1
0 1 0 0	2
0 1 0 1	2
0 1 1 0	2
0 1 1 1	2
1 0 0 0	R2
1 0 0 1	R2
1 0 1 0	R2
1 0 1 1	R2
1 1 0 0	R2
1 1 0 1	R2
1 1 1 0	R2
1 1 1 1	R2

آدرس معتبر →

آدرس معتبر	N.H
00 XX	1
01 XX	2
1XXX	R2

Prefix   آدرس های سلسله مراتبی

Net ID	host ID

32 bit

در هنگام Forwarding آدرس شبکه

راهنماندہ ما را به سایز جدول

اندازه NetID به اندازه شبکه وابسته است

0	Net	host
8	24	

Class A   جنس بزرگ

1	0	Net	Host
16	16		

Class B   بزرگ   شبکه ها بر اساس

1	1	0	Net	host
24	8			

Class C   کوچک   اندازه

با این روش تعداد آدرس ها زود پر شد! (هدر رفت آدرس ↑)

راه حل این مشکل: تعداد بیت هارو میبریم بیت بیشتری (128) به آدرس تخصیص

bit

بدهیم ←——— IP V6

راه حل دارد