

IMD0043

---

INTERNET PROTOCOL (IP)



# REVISÃO: CAMADA DE REDE

- ▶ Funcionalidade: **Entregar os dados**
- ▶ Protocolo: *Internet Protocol (IP)*
- ▶ Alcança sua funcionalidade (entregar os dados) usando 3 conceitos:
  - ▶ **Endereçamento** (endereçamento IP)
  - ▶ **Roteamento** (utilizando uma variedade de protocolos)
  - ▶ **Cabeçalho** do pacote como uma interface (encapsulando dados em pacotes)
    - ▶ Forma do pacote passar informações para o roteador
    - ▶ Cabeçalho reflete em informação necessária para realizar tarefas básicas

# COMO É O PACOTE IP?

**Sintaxe:** formato do pacote

- ▶ Cabeçalho (*header*)
- ▶ Dados (*payload*)



**Semântica:** significado dos campos do cabeçalho

# QUAIS TAREFAS PRECISAMOS REALIZAR?

1. Ler os pacotes corretamente;
2. Entregar o pacote ao destino;
3. Entregar respostas ao pacote de volta a origem;
4. Carregar dados;
5. Avisar ao host o que fazer com o pacote quando ele chegar;
6. Especificar qualquer tratamento especial do pacote pela rede;
7. Lidar com problemas que aparecem durante o caminho/rota.

# 1. LER OS PACOTES CORRETAMENTE

- ▶ Aonde o cabeçalho termina?
- ▶ Aonde o pacote termina?
- ▶ Que protocolo estamos usando?

## 2. ENTREGAR OS PACOTES AO DESTINO

- ▶ Endereço de destino
- ▶ Se um host se é movido fisicamente, seu endereço muda?
- ▶ Se não, como construir uma Internet escalável?
- ▶ Se sim, então para que serve um endereço para identificação?

### 3. ENTREGAR RESPOSTAS AO PACOTE DE VOLTA A ORIGEM

- ▶ Endereço de origem
- ▶ Necessário para que roteadores respondam a origem
  - ▶ Quando eles precisam se comunicar com a origem? **Falhas!**
  - ▶ Eles realmente precisam responder a origem?
  - ▶ Como uma origem sabe se um pacote chegou ao destino?

## 4. CARREGAR DADOS

- ▶ Payload!

## 5. AVISAR AO HOST O QUE FAZER COM O PACOTE QUANDO ELE CHEGAR

- ▶ Indicar quais protocolos devem lidar com o pacote
- ▶ Indicar opções desejadas

## 6. ESPECIFICAR QUALQUER TRATAMENTO ESPECIAL DO PACOTE PELA REDE

- ▶ Tipo de Serviço
- ▶ Prioridade
- ▶ Outras opções...

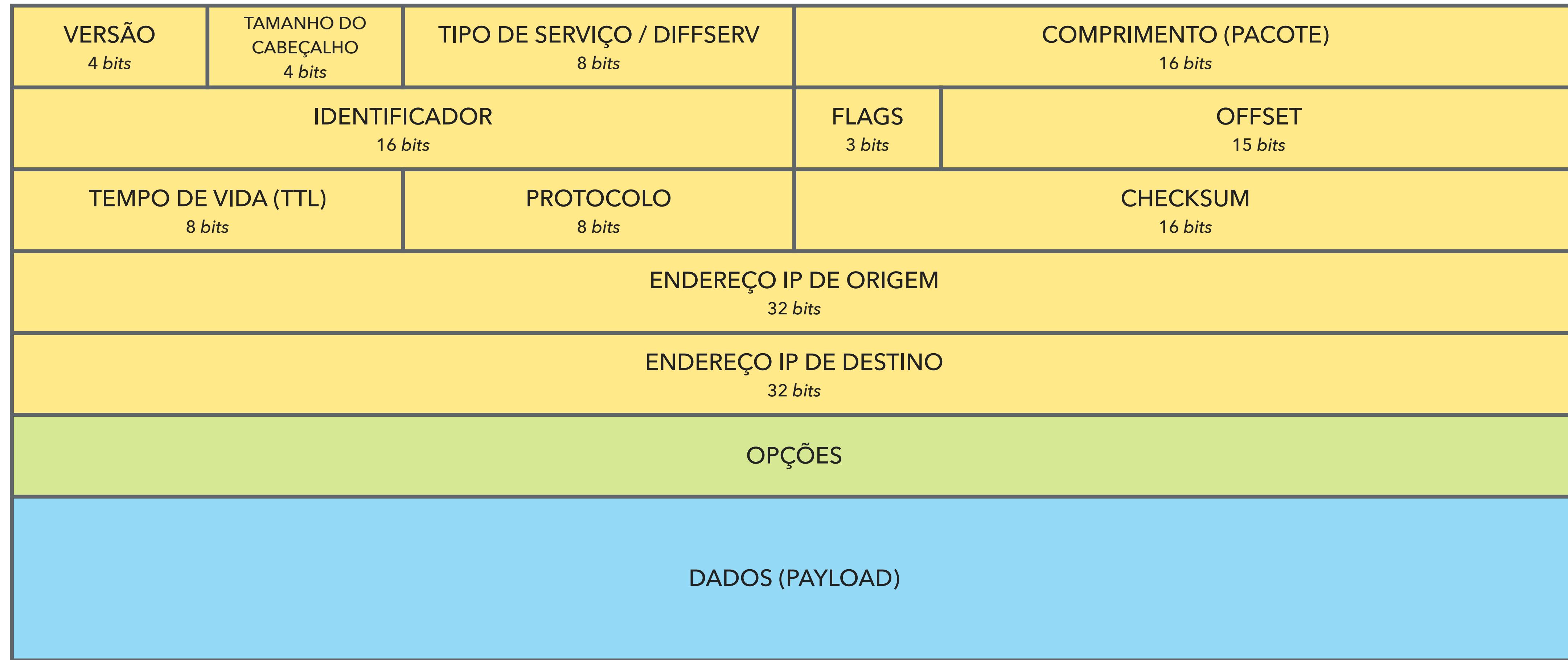
## 7. LIDAR COM PROBLEMAS QUE APARECEM DURANTE O CAMINHO/ROTA

- ▶ E se o pacote ficar em *loop*?
  - ▶ TTL
- ▶ Cabeçalho corrompido:
  - ▶ checksum
  - ▶ e checksum do *payload*?
- ▶ Pacote muito grande?
  - ▶ Lidar com fragmentação
  - ▶ Dividir o pacote em pedaços
  - ▶ Saber como remontar depois...

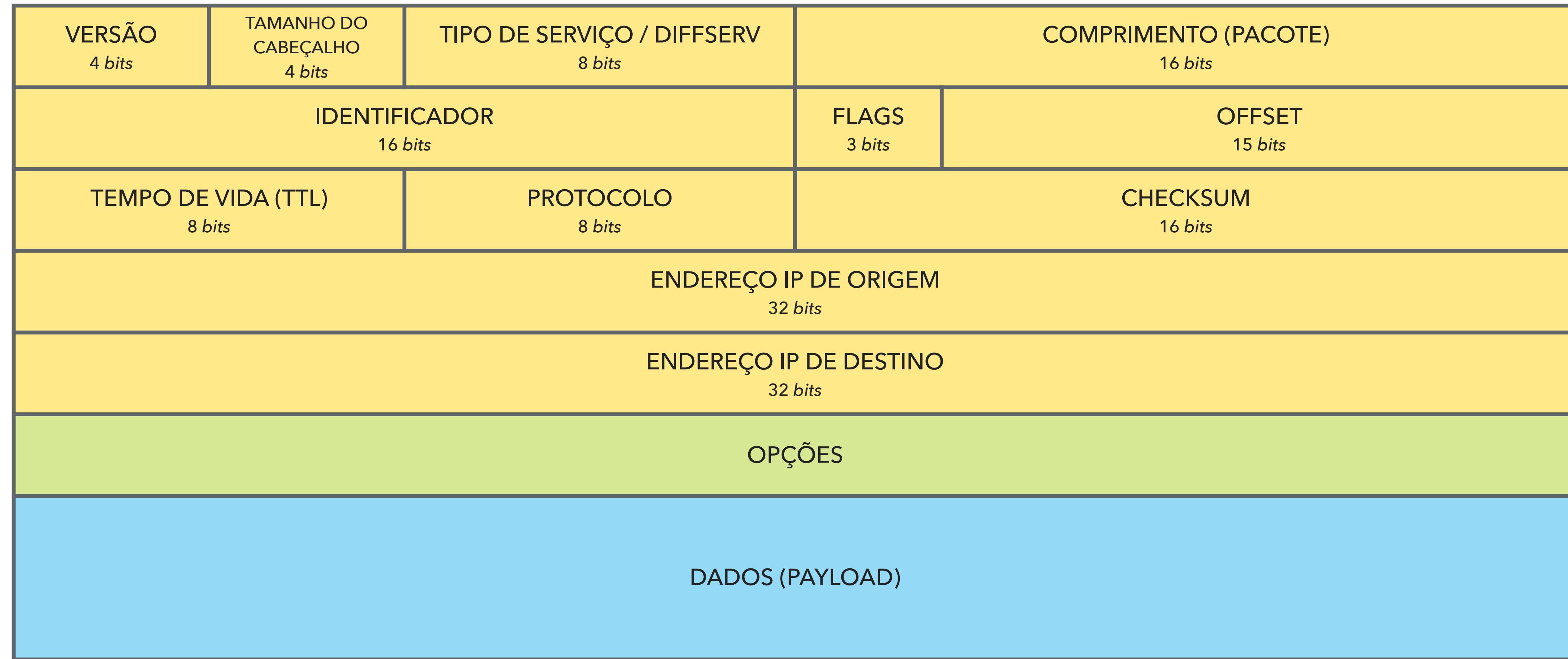
# DA SEMÂNTICA PARA A SINTAXE

- ▶ Falamos sobre as **tarefas** que o cabeçalho deve atender...
- ▶ Agora vamos verificar a sintaxe (formato) do **cabeçalho IPv4**, e também discutir a semântica em mais detalhes

# FORMATO DO PACOTE IPV4



# CABEÇALHO BASE DE 20 BYTES, E ENTÃO OPÇÕES



# 1. LER OS PACOTES CORRETAMENTE

## Versão (4 bits)

- ▶ Indica a versão do protocolo IP
- ▶ Necessário para saber o que se esperar dos próximos campos
- ▶ Tipicamente “4” (para IPv4), e cada vez mais “6” (para IPv6)

## Tamanho do cabeçalho (4 bits)

- ▶ Quantidade de palavras de 32 bits no cabeçalho
- ▶ Tipicamente “5”, para o cabeçalho base de 20 bytes do IPv4
- ▶ Pode ser maior caso o campo de opções seja utilizado

## Comprimento (16 bits)

- ▶ Quantidade de bytes do pacote
- ▶ Tamanho máximo é de 65.535 bytes ( $2^{16} - 1$ )
- ▶ ... porém enlaces podem impor limites menores!

# 1. CAMPOS PARA LER O PACOTE CORRETAMENTE

VERSÃO <i>4 bits</i>	TAMANHO DO CABEÇALHO <i>4 bits</i>	TIPO DE SERVIÇO / DIFFSERV <i>8 bits</i>	COMPRIMENTO (PACOTE) <i>16 bits</i>	
IDENTIFICADOR <i>16 bits</i>		FLAGS <i>3 bits</i>	OFFSET <i>15 bits</i>	
TEMPO DE VIDA (TTL) <i>8 bits</i>	PROTOCOLO <i>8 bits</i>	CHECKSUM <i>16 bits</i>		
ENDEREÇO IP DE ORIGEM <i>32 bits</i>				
ENDEREÇO IP DE DESTINO <i>32 bits</i>				
OPÇÕES				
DADOS (PAYLOAD)				

## 2/3. ENTREGAR OS PACOTES AO DESTINO E DE VOLTA A ORIGEM

Endereço de destino (32 bits)

- ▶ Localizador único para o host destinatário
- ▶ Permite que cada nó realize decisões de encaminhamento

Endereço de origem (32 bits)

- ▶ Localizador único para o host que está enviando o pacote
- ▶ Destinatário pode decidir se irá aceitar o pacote
- ▶ Permite que o destinatário envie uma resposta de volta a origem

## 2/3. CAMPOS PARA ENTREGAR OS PACOTES AO DESTINO E DE VOLTA A ORIGEM

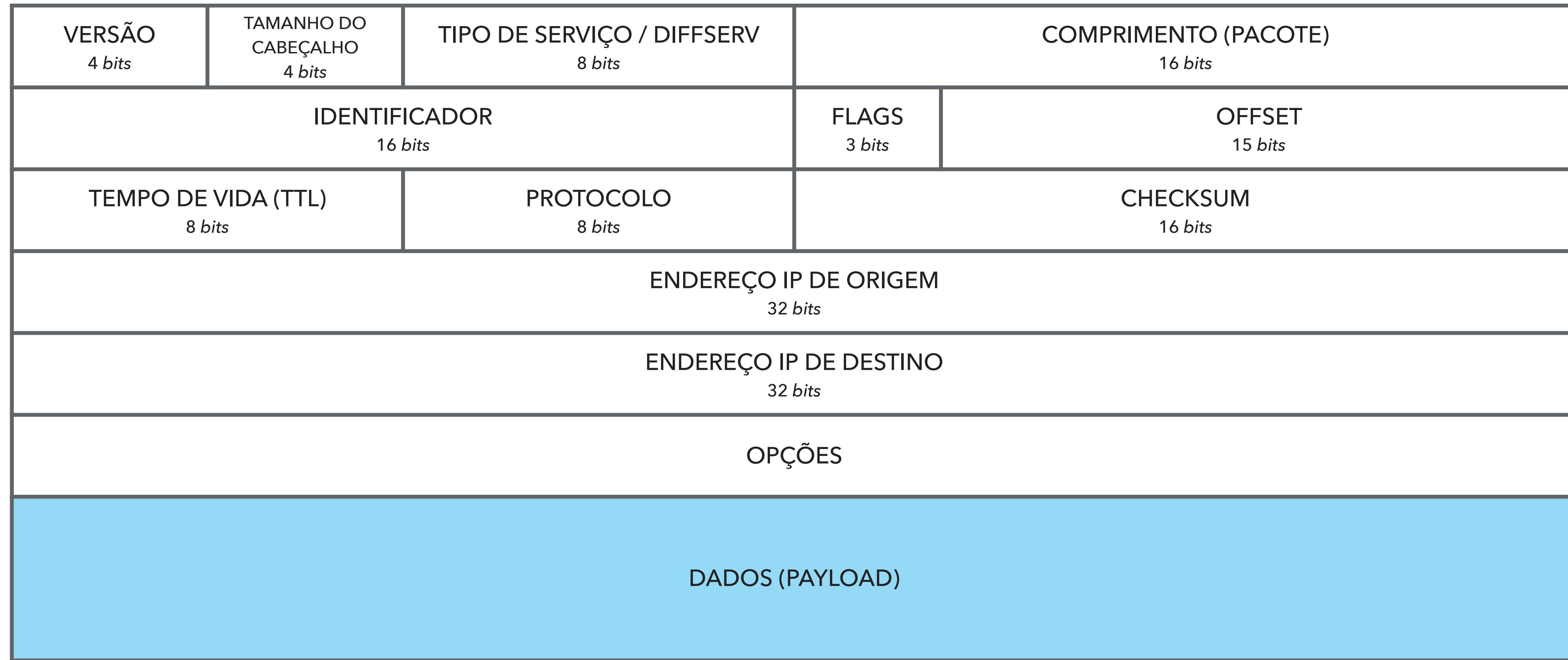
VERSÃO 4 bits	TAMANHO DO CABEÇALHO 4 bits	TIPO DE SERVIÇO / DIFFSERV 8 bits	COMPRIMENTO (PACOTE) 16 bits	
IDENTIFICADOR 16 bits		FLAGS 3 bits	OFFSET 15 bits	
TEMPO DE VIDA (TTL) 8 bits	PROTOCOLO 8 bits	CHECKSUM 16 bits		
ENDEREÇO IP DE ORIGEM 32 bits				
ENDEREÇO IP DE DESTINO 32 bits				
OPÇÕES				
DADOS (PAYLOAD)				

## 4. CARREGAR DADOS

Payload ( $\times$  bits)

- ▶ Não faz parte do cabeçalho!

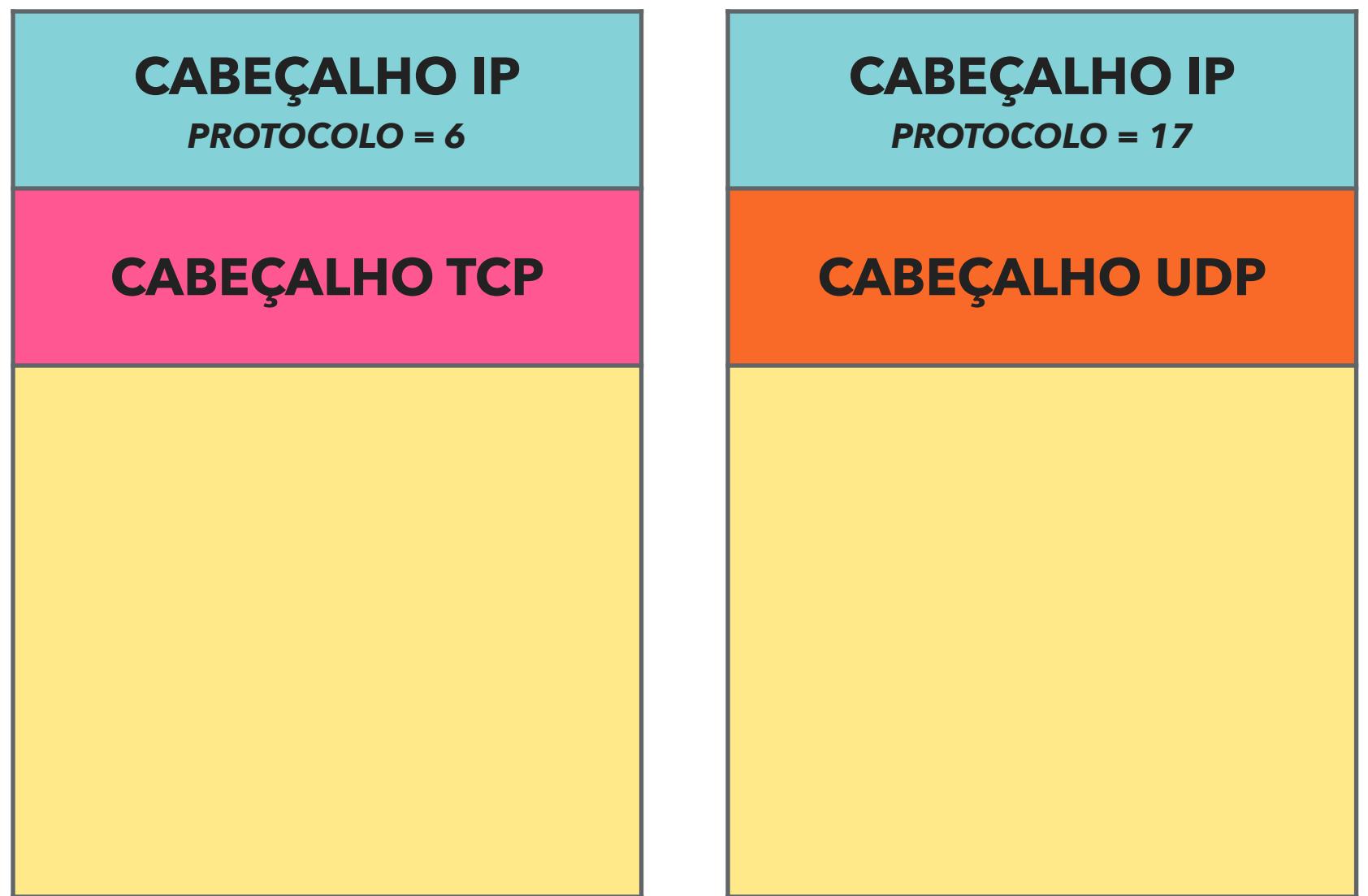
## 4. CAMPO PARA CARREGAR DADOS (NÃO FAZ PARTE DO CABEÇALHO!)



## 5. AVISAR AO HOST O QUE FAZER COM O PACOTE QUANDO ELE CHEGAR

### Protocolo (8 bits)

- ▶ Identifica o protocolo de camada superior
- ▶ Importante para demultiplexação no host de destino
- ▶ Exemplos comuns:
  - ▶ “6” para o protocolo TCP
  - ▶ “17” para o protocolo UDP



## 5. CAMPO PARA AVISAR AO HOST O QUE FAZER COM O PACOTE QUANDO ELE CHEGAR

VERSÃO <i>4 bits</i>	TAMANHO DO CABEÇALHO <i>4 bits</i>	TIPO DE SERVIÇO / DIFFSERV <i>8 bits</i>	COMPRIMENTO (PACOTE) <i>16 bits</i>	
IDENTIFICADOR <i>16 bits</i>		FLAGS <i>3 bits</i>		OFFSET <i>15 bits</i>
TEMPO DE VIDA (TTL) <i>8 bits</i>	PROTOCOLO <i>8 bits</i>	CHECKSUM <i>16 bits</i>		
ENDEREÇO IP DE ORIGEM <i>32 bits</i>				
ENDEREÇO IP DE DESTINO <i>32 bits</i>				
OPÇÕES				
DADOS (PAYLOAD)				

## 6. ESPECIFICAR QUALQUER TRATAMENTO ESPECIAL DO PACOTE PELA REDE

### Tipo de Serviço (*Type of Service*) (8 bits)

- ▶ Permite que pacotes sejam tratados de formas diferentes baseado em necessidades
- ▶ Ex: baixa latência para áudio, alta largura de banda para transferências grandes
- ▶ Foi refinado várias vezes; na prática não foi largamente implementado

### Opções

- ▶ Permite especificar "outras" funcionalidades
- ▶ Formato extensível
- ▶ Não são normalmente utilizados
- ▶ Ex: *Timestamp*, *Record Route*, ...

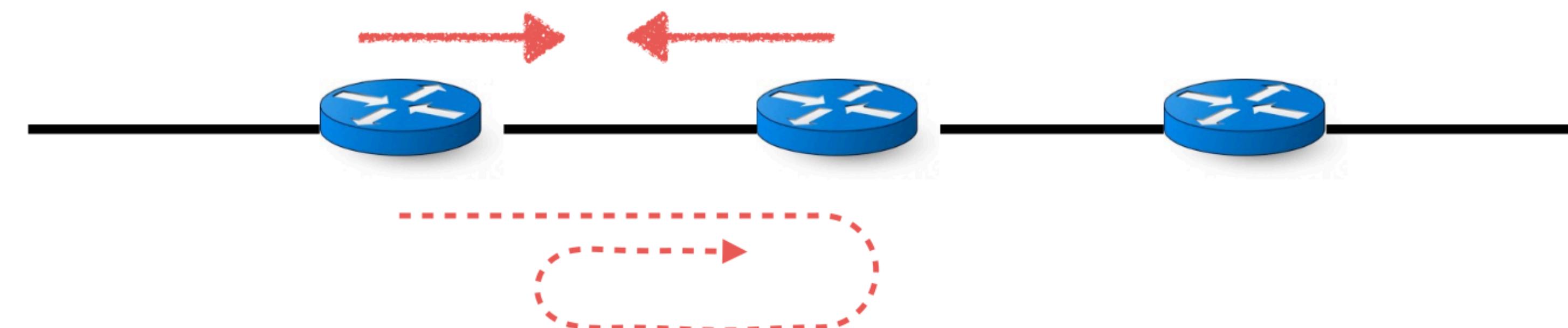
## 6. CAMPOS PARA ESPECIFICAR QUALQUER TRATAMENTO ESPECIAL DO PACOTE PELA REDE

VERSÃO 4 bits	TAMANHO DO CABEÇALHO 4 bits	TIPO DE SERVIÇO / DIFFSERV 8 bits	COMPRIMENTO (PACOTE) 16 bits						
IDENTIFICADOR 16 bits		FLAGS 3 bits	OFFSET 15 bits						
TEMPO DE VIDA (TTL) 8 bits	PROTOCOLO 8 bits	CHECKSUM 16 bits							
ENDEREÇO IP DE ORIGEM 32 bits									
ENDEREÇO IP DE DESTINO 32 bits									
OPÇÕES									
DADOS (PAYLOAD)									

## 7. LIDAR COM PROBLEMAS QUE APARECEM DURANTE O CAMINHO/ROTA

### TTL (8 bits)

- ▶ Prevenção de *loops*
- ▶ O valor é decrementado a cada "salto" (*hop*), e o pacote é descartado quando o TTL=0
- ▶ e uma mensagem de *time exceeded* é enviada de volta para a origem (utilizando ICMP)

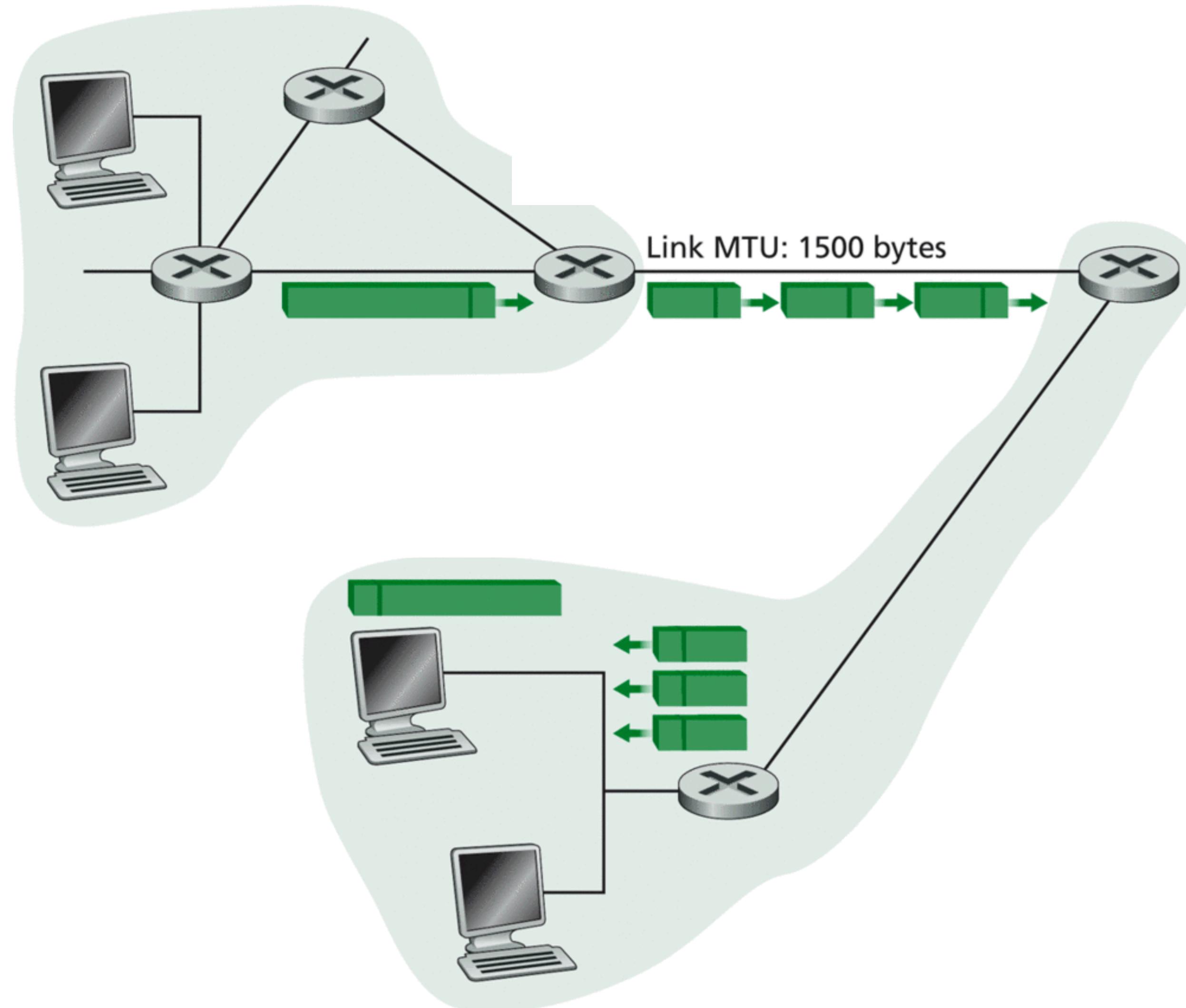


## 7. LIDAR COM PROBLEMAS QUE APARECEM DURANTE O CAMINHO/ROTA

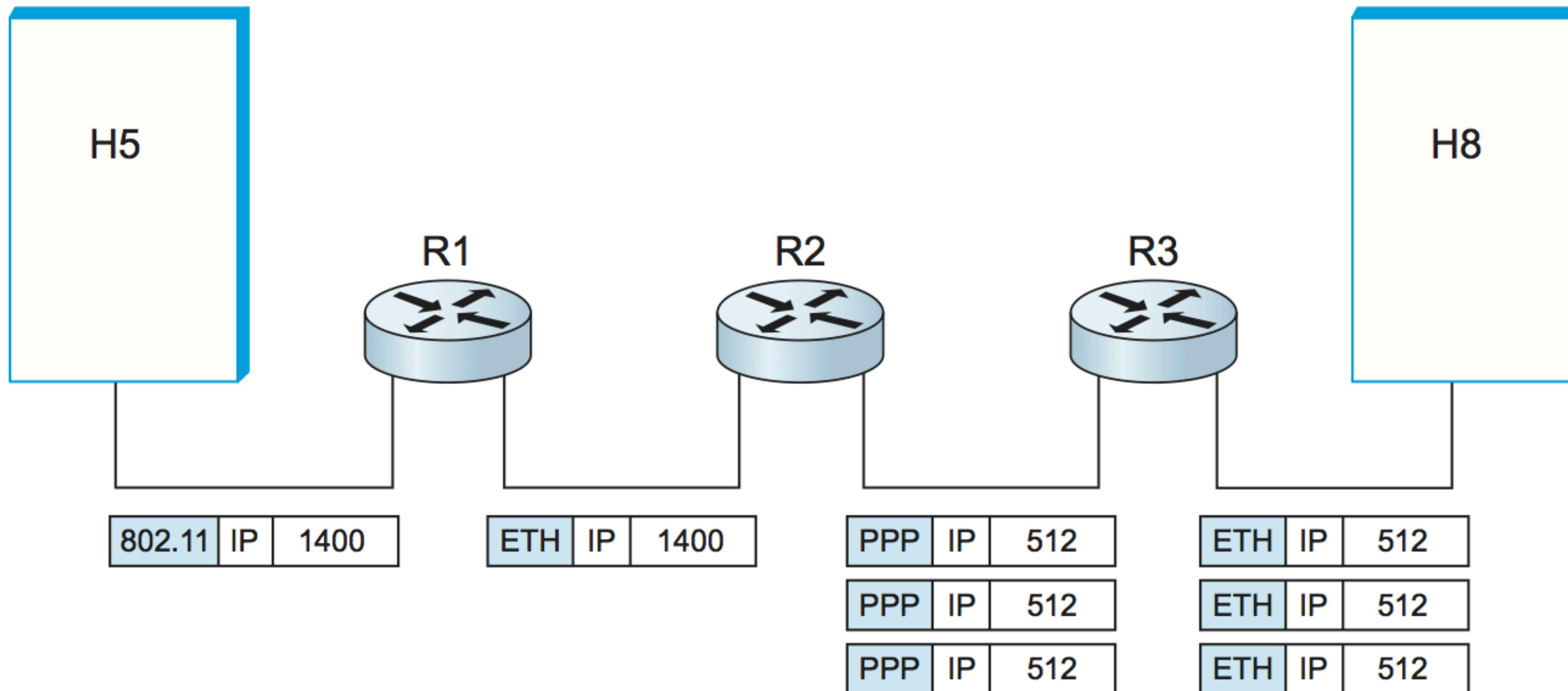
### Checksum (16 bits)

- ▶ Checksum em cima do cabeçalho do pacote
- ▶ Se não bater o checksum, descarta o pacote
- ▶ Checksum é recalculado em todo roteador
- ▶ Por que?
- ▶ Por que não calcular o checksum em todo o pacote (incluindo dados)?

# FRAGMENTAÇÃO IPV4



# FRAGMENTAÇÃO IPV4



*todo fragmento é um pacote IP!*

## 7. LIDAR COM PROBLEMAS QUE APARECEM DURANTE O CAMINHO/ROTA

**Identificador** (*16 bits*)

- ▶ Usado para identificar fragmentos do datagrama IP original

**Flags** (*3 bits*)

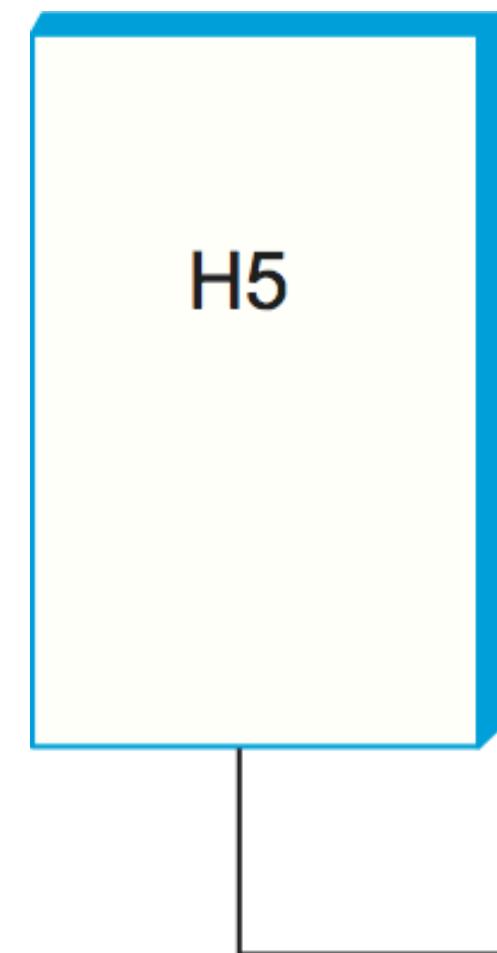
- ▶ Usado para controlar ou identificar fragmentos

**Offset** (*13 bits*)

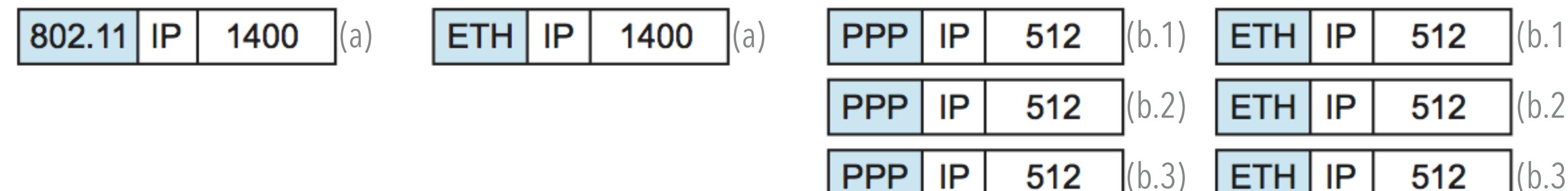
- ▶ Permite que um receptor determine o local de um fragmento em particular no datagrama IP original.

# FRAGMENTAÇÃO IPV4

(a)	Start of header
	Ident=x    0    Offset=0
	Rest of header
	1400 data bytes



(b)	Start of header
	Ident=x    1    Offset=0
	Rest of header
	512 data bytes



(b)	Start of header
	Ident=x    1    Offset=64
	Rest of header
	512 data bytes

(b)	Start of header
	Ident=x    0    Offset=128
	Rest of header
	376 data bytes

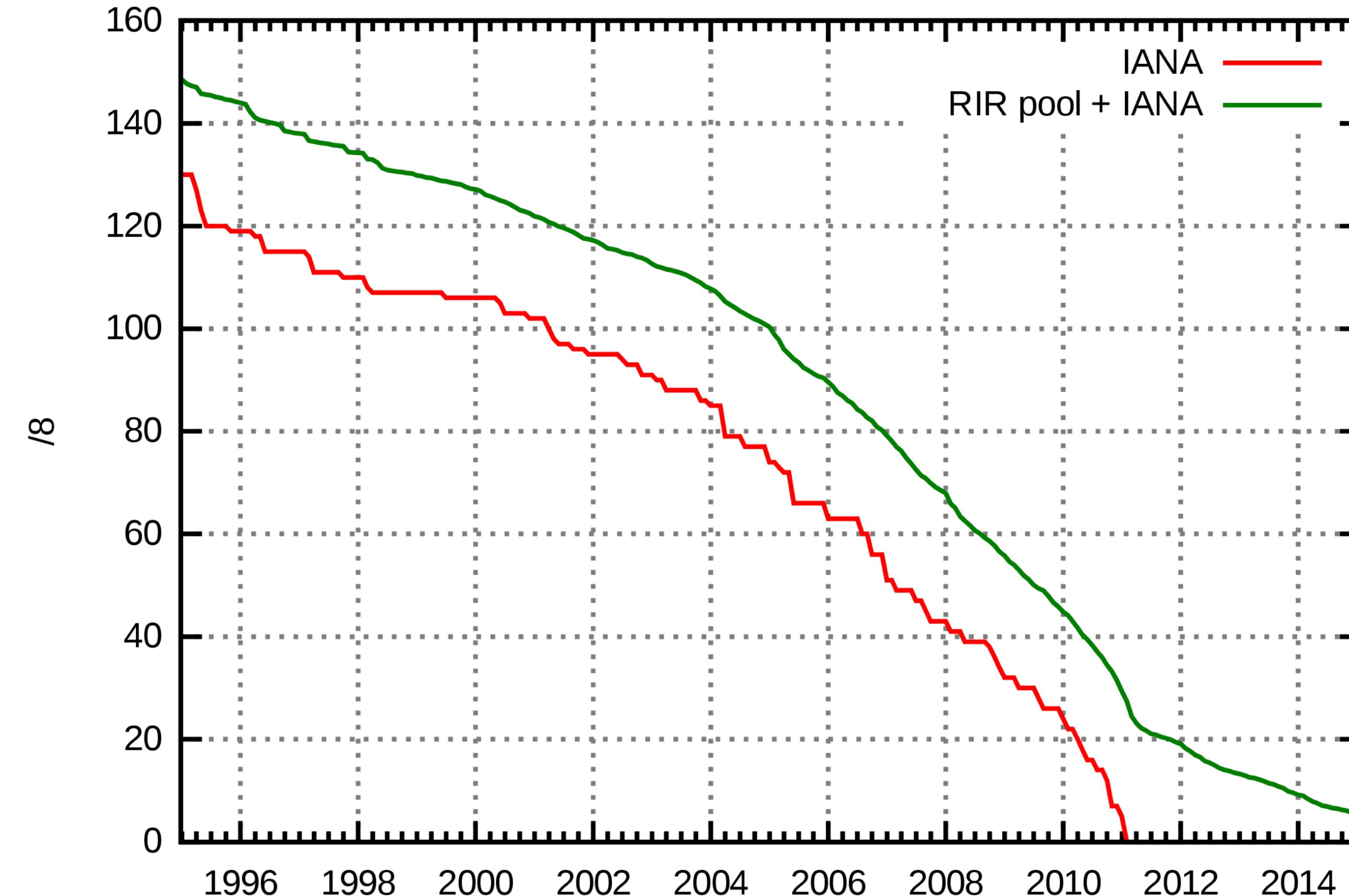
## 7. CAMPOS PARA LIDAR COM PROBLEMAS QUE APARECEM DURANTE O CAMINHO/ROTA

VERSÃO 4 bits	TAMANHO DO CABEÇALHO 4 bits	TIPO DE SERVIÇO / DIFFSERV 8 bits	COMPRIMENTO (PACOTE) 16 bits	
IDENTIFICADOR 16 bits		FLAGS 3 bits	OFFSET 13 bits	
TEMPO DE VIDA (TTL) 8 bits	PROTOCOLO 8 bits	CHECKSUM 16 bits		
ENDEREÇO IP DE ORIGEM 32 bits				
ENDEREÇO IP DE DESTINO 32 bits				
OPÇÕES				
DADOS (PAYLOAD)				

# CABEÇALHO IP COMO UMA INTERFACE

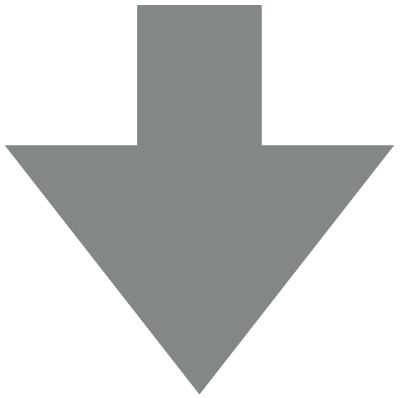
- ▶ É inútil decorar o cabeçalho IP
  - ▶ Se você lembrar das tarefas que tem que ser realizadas
  - ▶ Entender por que o formato do cabeçalho é dessa forma
  - ▶ Em geral: se você entende o problema, a solução é fácil
- ▶ Transição do IPv4 para o IPv6
  - ▶ Acontecendo gradualmente...

# EXAUSTÃO DO IPV4



# SOLUÇÃO (NÃO PROVISÓRIA)?

32 bits



128 bits

# SOLUÇÃO (NÃO PROVISÓRIA)?

32 bits

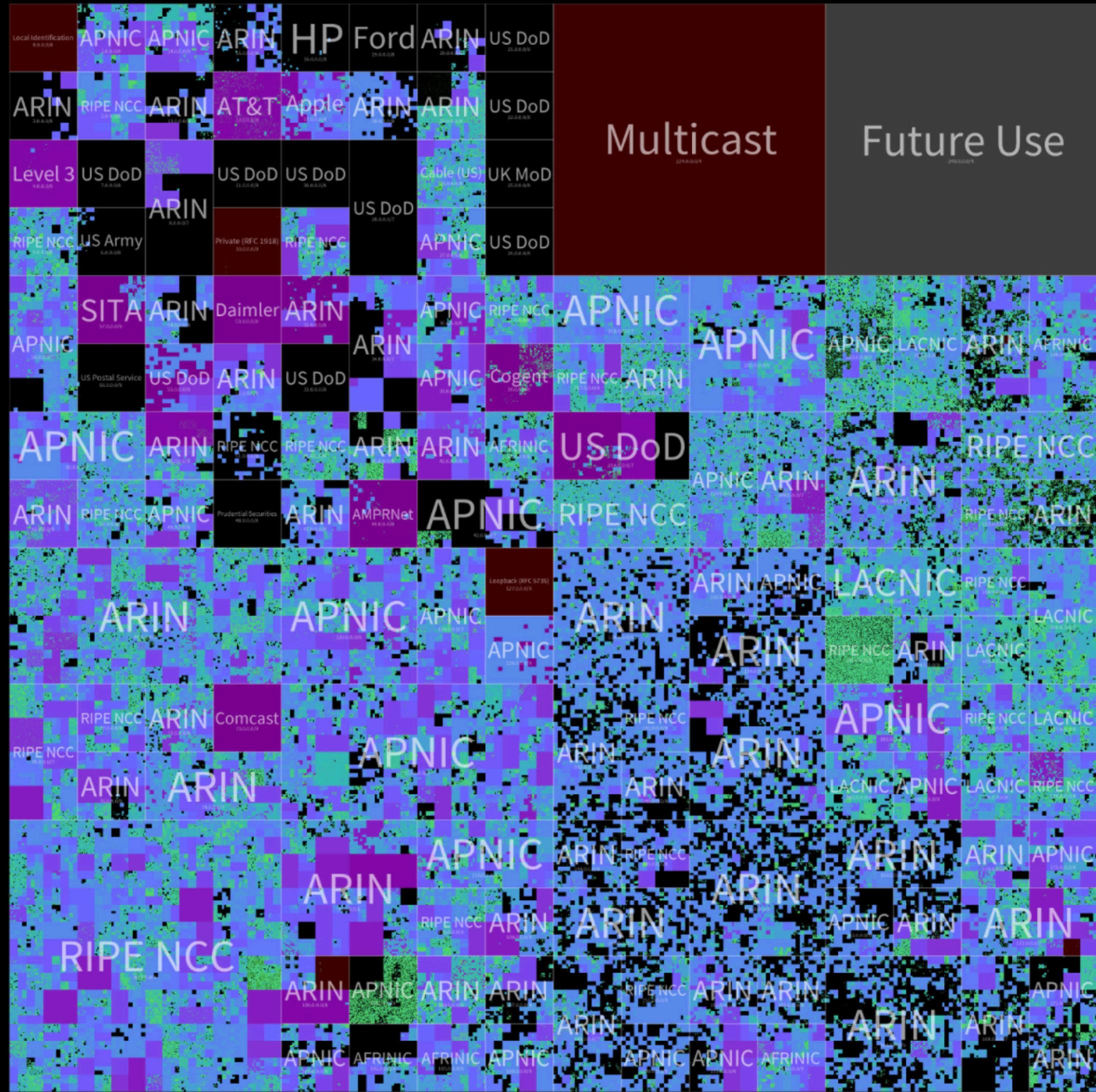
$$2^{32} = 4.294.967.296 \text{ endereços}$$

128 bits

$$2^{128} = 340.282.366.920.938.463.463.374.607.431.768.211.456 \text{ endereços}$$

# The IPv4 Internet

January 2019



# The IPv6 Internet

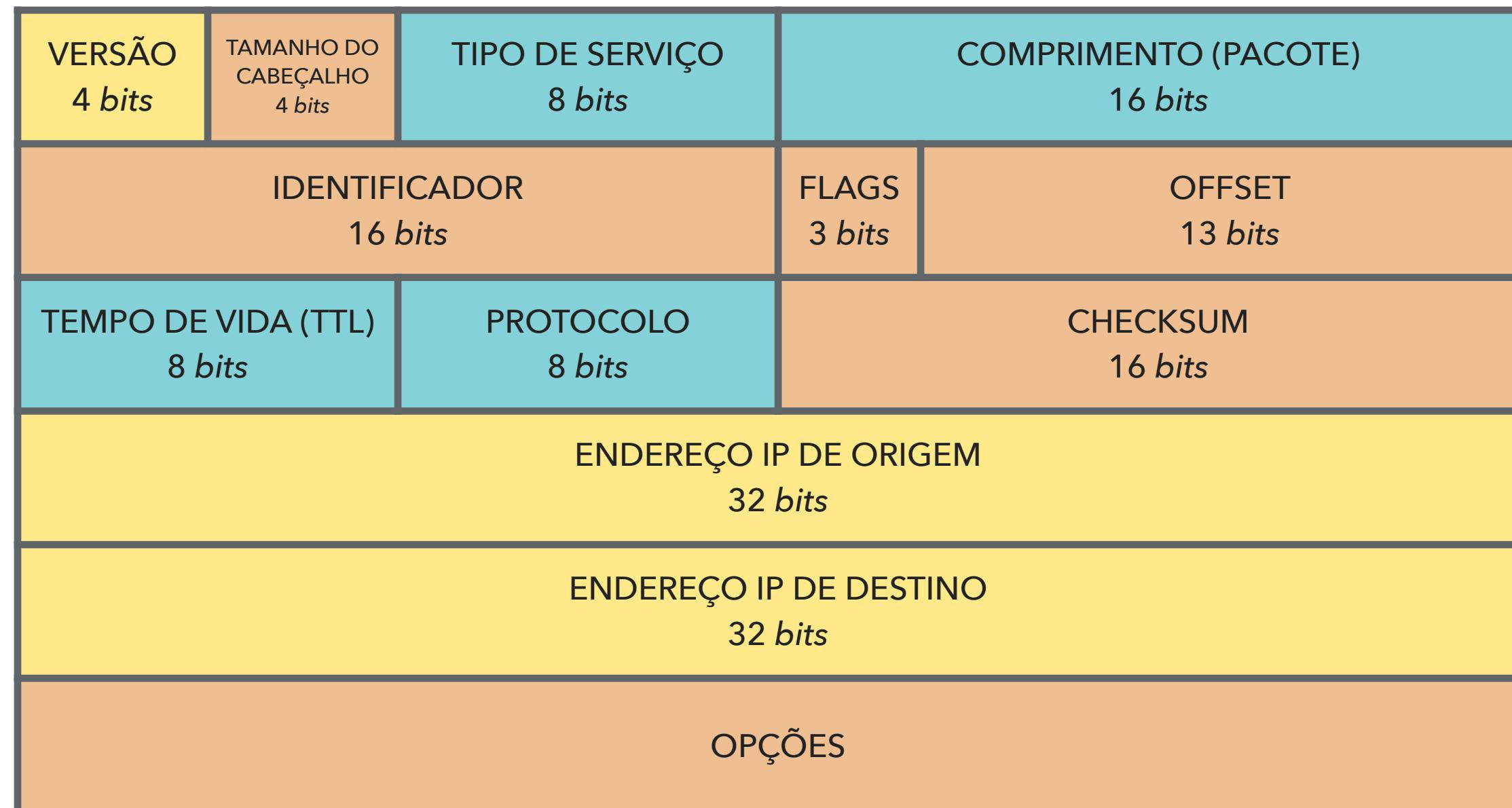
January 2019



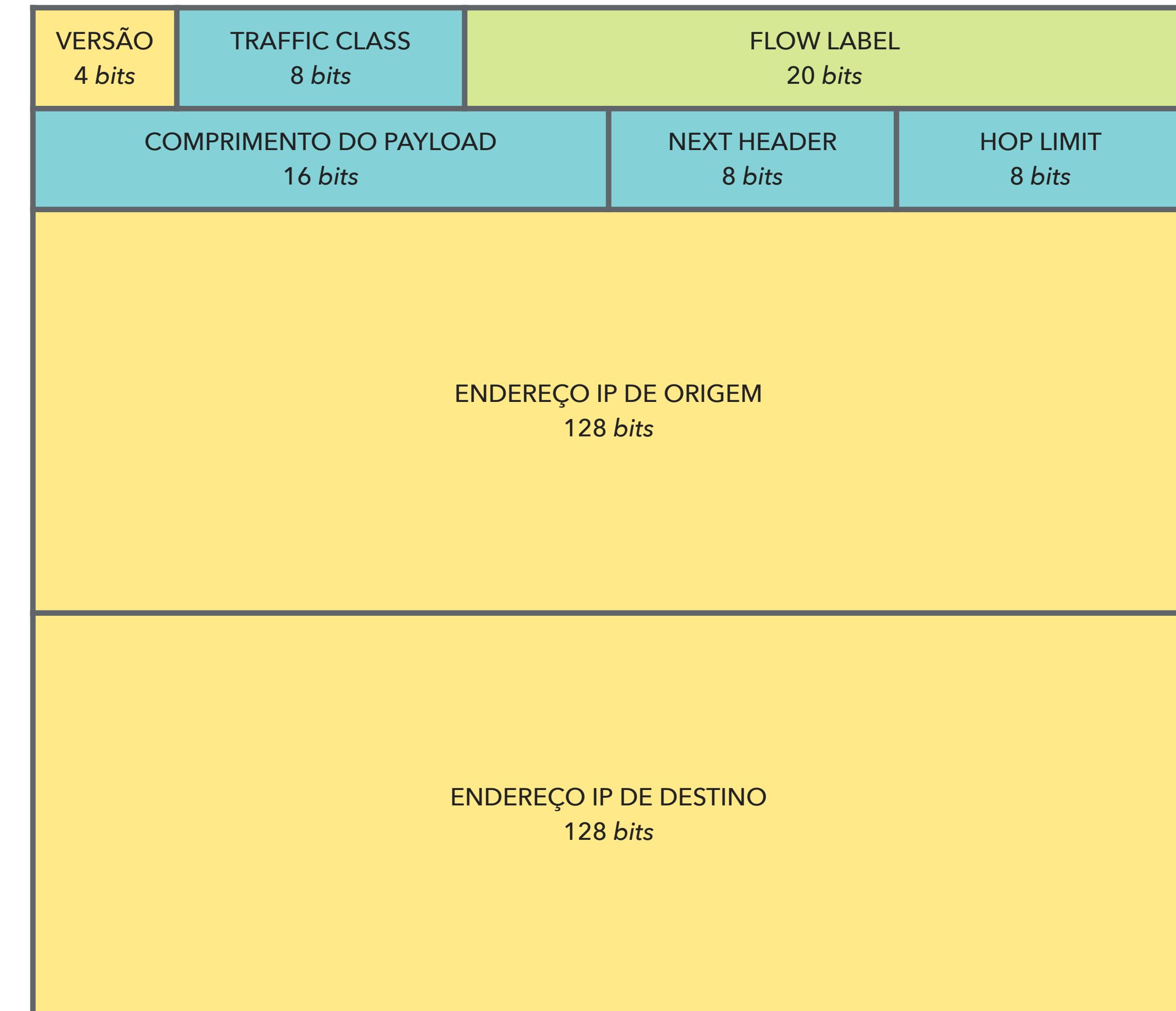
# IPv6

- ▶ Simplificar o IP
- ▶ Remover todos os campos que não são absolutamente necessários

# COMPARAÇÃO DOS CABEÇALHOS IPV4 E IPV6



	Campos que permaneceram do IPv4 para o IPv6
	Campos removidos no IPv6
	Campo mudou de nome e posição no IPv6
	Campo novo no IPv6



# RESUMO DAS MUDANÇAS

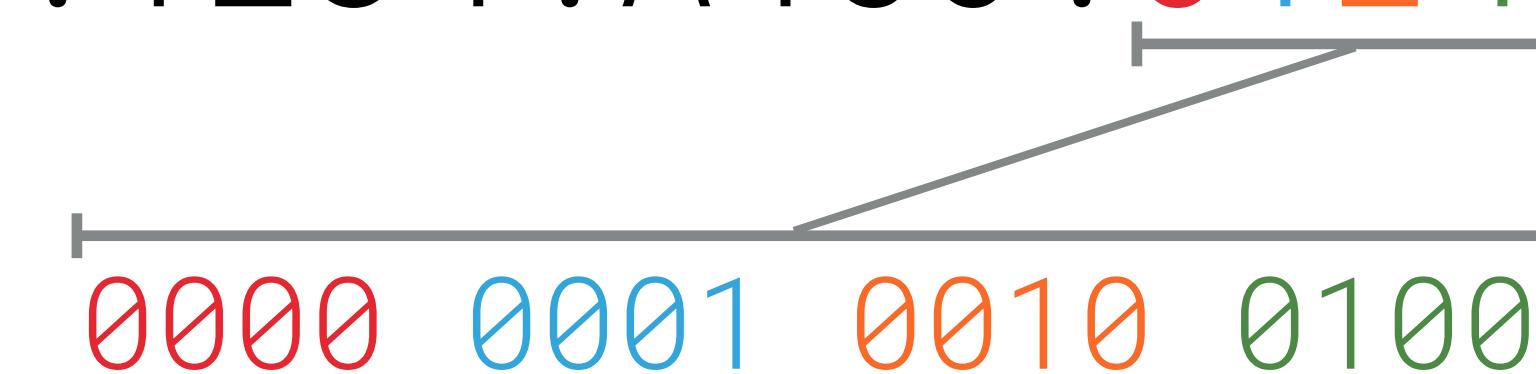
- ▶ Eliminou a fragmentação
- ▶ Eliminou o comprimento do cabeçalho
- ▶ Eliminou checksum
- ▶ Novo mecanismo de opções (cabeçalhos de extensão)
- ▶ Espaço de endereçamento foi expandido
- ▶ Adicionado o *Flow Label*

# NOTAÇÃO DE ENDEREÇOS IPV6

47CD:1234:4422:AC02:0022:1234:A456:0124

# NOTAÇÃO DE ENDEREÇOS IPV6

47CD : 1234 : 4422 : AC02 : 0022 : 1234 : A456 : 0124



# NOTAÇÃO DE ENDEREÇOS IPV6

47CD:0000:0000:0000:0000:0000:A456:0124

47CD::A456:0124

47CD::A456:124

# NOTAÇÃO DE ENDEREÇOS IPV6

2001 :0db8 :85a3 :03fa :0000 :0000 :0000 :7344

2001 :0db8 :85a3 :03fa ::7344

2001 :db8 :85a3 :3fa ::7344