Teórico (09/02)

Nota mínima: 8,0 → T

10,0 → P

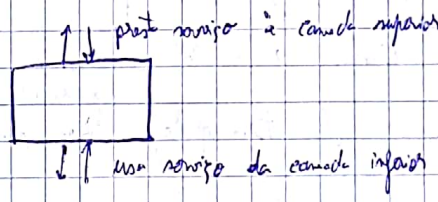
Teste - 05/06

Exame - 26/06

Teórica (16/02)

(3) Serviços camada de Transporte necessários?

TCP vs UDP



	TCP	UDP
- Serviço fiável: Garantia de entrega (Controlo de erros)	✓	✗
Garantia ordenação	✓	✗
- Serviço controlo ligação (Inicia, mantém e termina)	✓	✗
- Serviço congestão (Controlo congestão)	✓	✗
- Serviço Controlo de fluxo	✓	✗
- Multiplexar/Desmultiplexar vários apps em simultâneo	✓	✓

3 • Não é possível sobre o IP?

Não, pois não há ordenamento de entrega

Se está congestionado, não enviar mais bytes

O emissor está a enviar bem e depressa, mas o receptor não consegue processar. Logo, haver um mecanismo em que o receptor diz ao emissor para abrandar pois tem pouco espaço para processar. Controlar o débito entre os dois sistemas terminais

Se tivermos vários apps não podemos misturar os pacotes. Cada pacote vai pelo seu pilha correta

- Desmultiplexagem
  - Orientado à conexão

É necessário existir uma conexão, para isso é necessário existir 4 itens

Portas:

0 - 1024 : Reservado

1024 - 65533 Utilizador

Check-sum: Costuma estar sempre a 0, assumimos que a rede já é fiável

(19) Teste

Não exige recursos, um buffer ocupa pouco recurso, p.e.:

Velocidade é alta, criamos o socket e enviamos logo.

Overhead da camada de Transporte, leva no máximo 8 bytes, no TCP leva bastante mais

Não existe garantia de entrega, mas muitas vezes isto nem é necessário.

(31)

Teórica (23/02)

Funções da Camada de Transporte:

→ Multiplexagem / Desmultiplexagem da aplicação ✓ Porta origem/Destino (Sockets)

→ Controlo de conexão ✓ (FIN, ACK, ...)

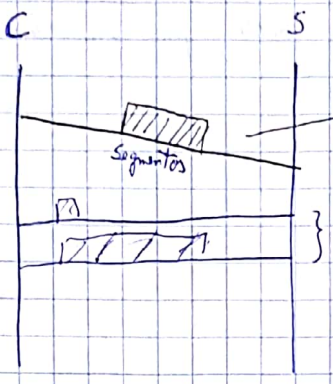
→ Serviço fiável, controlo de erros — 1. Detetar erro; 2. Lidar c/ erro

→ Controlo congestão

→ Controlo fluxo

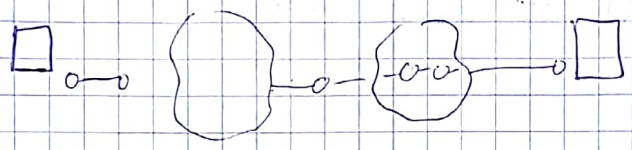
Scanned by CamScanner

Estabelecimento da ligação TCP/IP:



→ MSS (Tamanho máximo de envio)

} Será que devemos enviar pacotes pequenos ou grandes?



Como saber o MSS?

- Calcular o MIN (Percurso)

{ Calcular o número de saltos e o MTU (Maximum Transfer Unit) → Dado pelo routing
{ Assim, já sabemos o MSS

(33) Transporte fiável - controlo de erros

1. Detetar erros:

- Precisamos de um nº sequência
- Checksum
- Timeout

} Resposta às perguntas

→ Para detetar se está íntegro (se chegou na sua totalidade)

→ Para detetar se algum se perdeu

Como determinar o timeout?

Tem de ser muito maior que o tempo de ida e volta (RTT)

↓

Round Trip Time

Ele é variável ao longo do tempo

Depende da rede

Scanned by CamScanner

(34) Como corrigir o erros?

- Usando um ARQ (Pedindo de novo o pacote, pois ele chegou com erros)

↓

Resposta pergunta 3

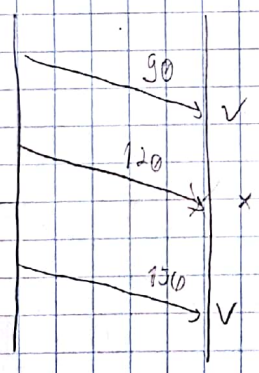
(36)

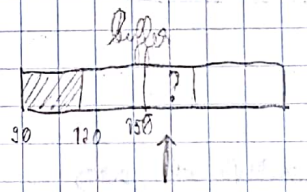
Número de sequência: Número de bytes

3 ACK's repetidos (NACK) → Taxa de erroos

(46)

- 30
- ACK 90



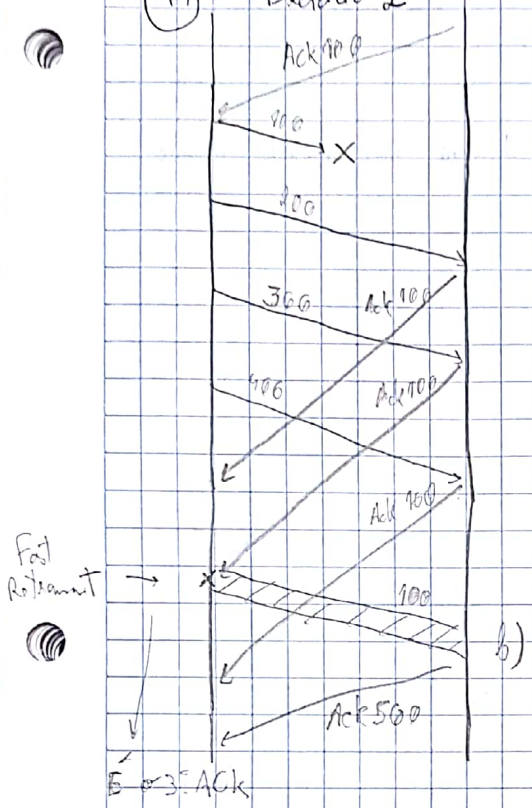


1. Guardar no buffer: Selective repeat
2. Não guardar no buffer: Go-Back-N

! Selective Reject

Scanned by CamScanner

(47) Exercício 2



// Acaba aqui o controlo de erros

(49) Controlo de fluxo

(52) Controlo de congestão

CC tp1

2 ou 3?

Uso da camada transporte:
- fornece uma comunicação lógica entre processos;
- usa e melhora os serviços disponibilizados pela camada de rede;
- TCP (troca de dados fiável e ordenada):
  - Controlo fluxo;
  - Controlo de erros;
  - Controlo de congestão;
- UDP (troca de dados não fiável e desordenada);

Scanned by CamScanner

Teórica (02/03)

Exercício 3 → Teste

$t=5 \rightarrow$ Tamanho da janela de congestão TCP diminui para metade, devido a um ACK duplicado

a)

$t=5 \rightarrow$ [ACKs duplicados] janela = janela / 2

$t=13 \rightarrow$ [timeout] janela = 1

$t=17 \rightarrow$ [Chegou ao patamar de congestão - threshold]

b)

[0-5] → Slow Start

[5-13] → AIMD ⇔ Congestion avoidance
↳ Já foi descoberta a capacidade da rede e estamos numa fase de prevenção da congestão em que se pudesse aproveita e mandar, não vai cair

[13-17] → Slow Start

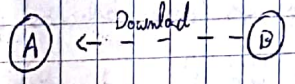
[17 ...] → AIMD ⇔ Congestion Avoidance

→ Quando ocorre congestão, atinge-se o pico e a janela passa para metade → ACKs duplicados

Em $t=5$ e $t=13$, o patamar é ajustado (threshold):

patamar = janela/2 → Apenas é ajustado em $t=5$ (ACKs duplicados)

Exercício 5



a) $f = 12\,000$ bytes $= 8 \times 12\,000$ bit
$R = 4$ Mbps $= 4 \times 10^6$ bps
RTT = 5 ms } normalmente, não têm um valor fixo. Foram fixados estes valores para simplificar
S = 500 bytes }
$t = ?$

$$t_{ótimo} = t_{UDP} = \frac{L}{R} = \frac{12000 \times 8 \text{ bit}}{4 \times 10^6 \text{ bit p/segundo}} = 0.02405 \simeq 24 \text{ ms} \rightarrow \text{ideal}$$

tt → Tempo de transmissão 500 bytes, porque s1 = 500 bytes

$$tt = \frac{S}{R} = \frac{500 \times 8 \text{ bit}}{4 \times 10^6} = 1 \text{ ms} = 1 \times 10^{-3} \text{ s}$$ → Decorre 1 ms desde o instante em que é enviado o primeiro bit, depois sai o último bit

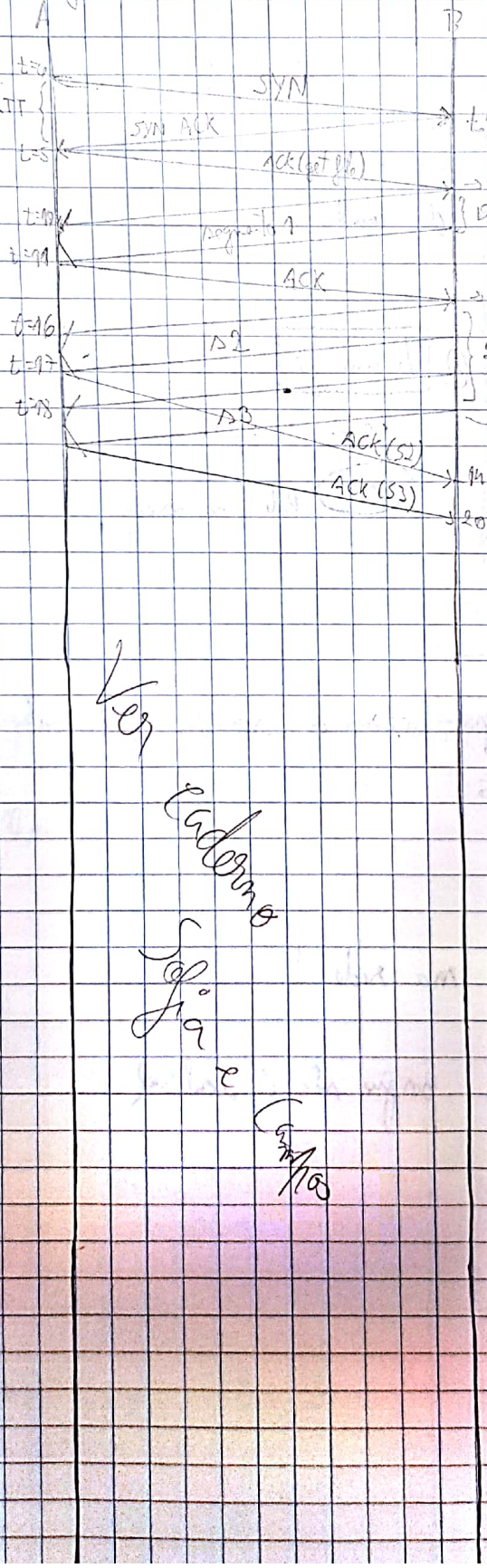
Scanned by CamScanner

S → Tamanho do segmento

$R_{efetivo} = \frac{12000 \times 8}{50 \times 10^{-3}} = 1,92$ Mbps

$N = \frac{L}{S} = \frac{12000 \text{ bytes}}{500 \text{ bytes}} = 24$ segmentos

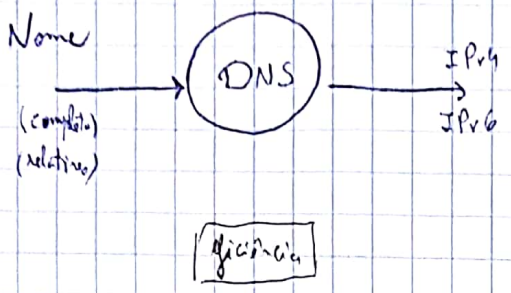
$U = \frac{R_{efetivo}}{R(real)} = \frac{1,92}{4} = 48\%$ → no melhor do caso



Ver caderno fig e cmo

Teórica (09/03)

(1)



a) Ver se os domínios têm servidor de mail
Pode usar DNS

costa @ di.uminho.pt

nome → IPv?

Domínio Pode ir ser servido

b) Também é resolvido por DNS

d) É necessário ir ao DNS para localizar os servidores VoIP onde entregar a ~~[illegible]~~ chamada.

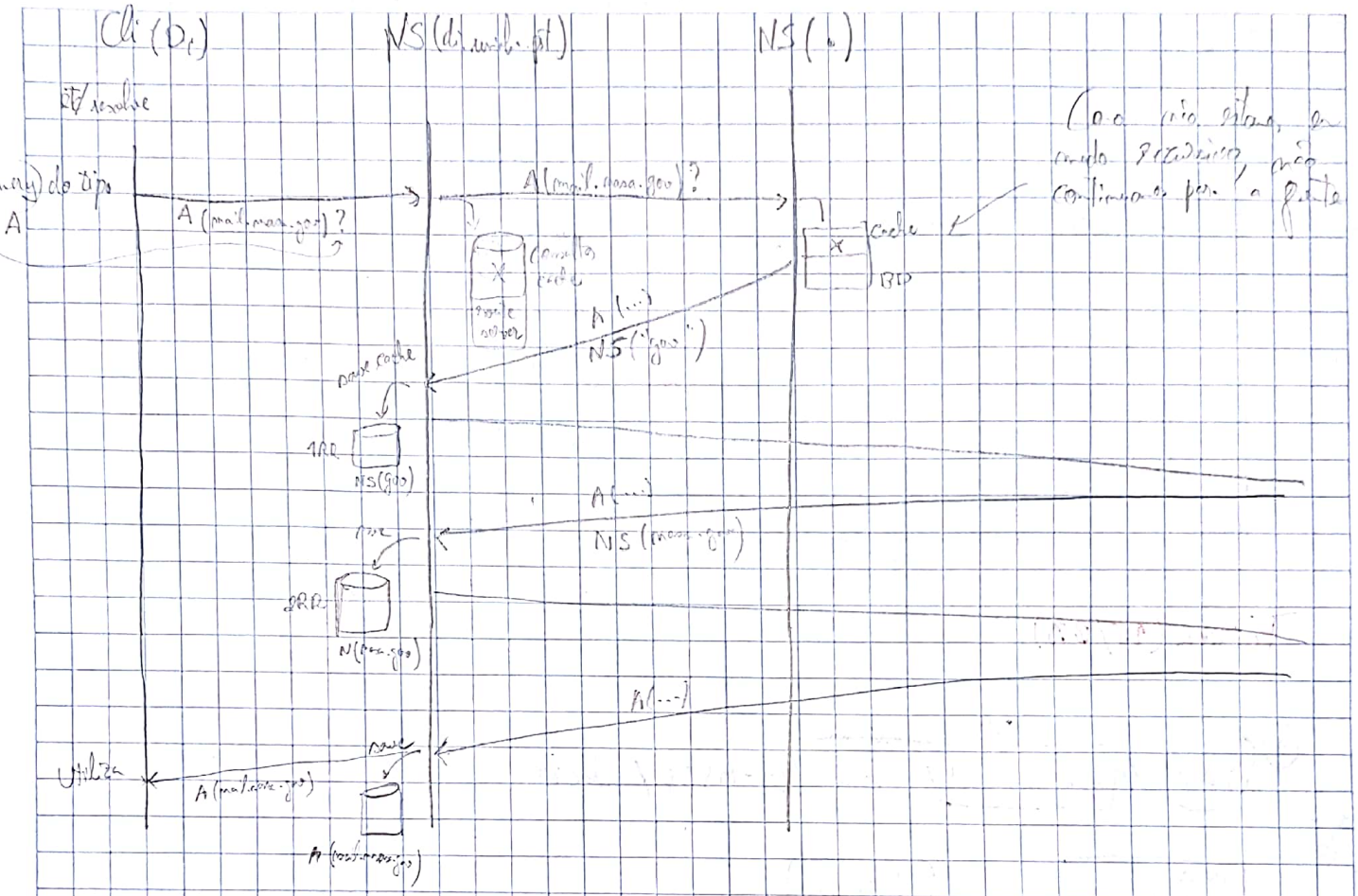
e) Também é preciso ir ao DNS

(3) Não colocar informação do estado na rede!

(4) Não colocar numa Base de Dados porque não é escalável

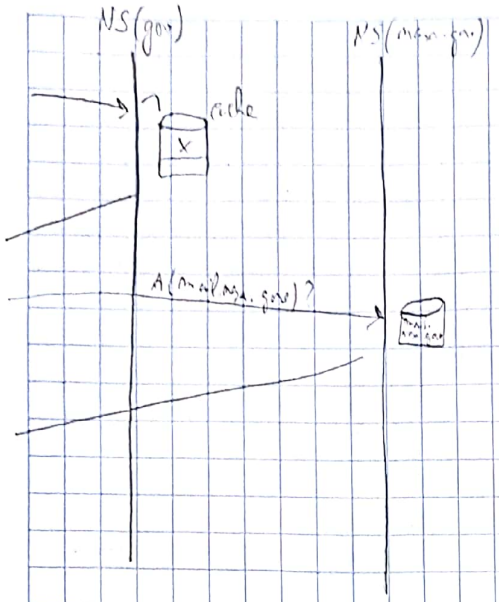
(27) Exercício

Scanned by CamScanner



Tempo (DNS) = 1 RTT local + 3 RTT

Ver melhor → caderno Sofia



Teórica (15/03)

① http://www.uminho.pt/disciplinas/CC-MIEI/index.html

DNS

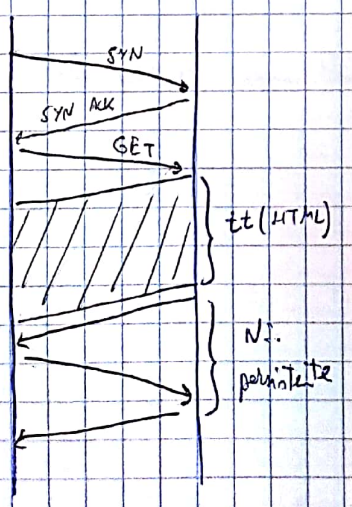
Como default porta 80

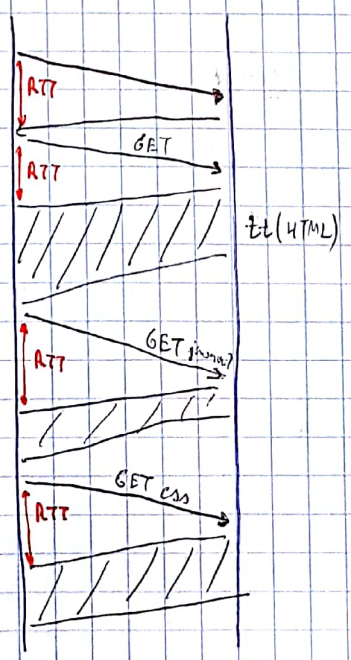
<HTML>
<script..
<img>
<...>
</HTML>

N recursos

1) Descarregar mal aparece
2) Lançar pedidos descargas em paralelo
3) Usar mesma conexão (reutilizar conexão?)

HTTP 1.0





Persistente

Sem Pipeline

Scanned by CamScanner

HTTP 1.1

Persistente

c/ Pipeline

Tempo de processamento

Se for muito elevado temos um bloqueio da conexão, logo resolvem com HTTP 2.0, deixando enviar algum conteúdo para o cliente

HTTP 2.0

1) Server PUSH (empurrar conteúdo não pedido)
2) Servidor escolhe a ordem
3) Multiplexar o objetos (partes dos objetos intercalados)

Web Services

URL identifica recurso				
http://.../livros/...	Obter o livro	Apagar o livro	Alterar o livro	X
http/.../livros	Obter objeto "lista livro"	apagar o recurso	altera a lista do livro	Criar um novo
Métodos	GET	Delete	PUSH	POST
	Read	Delete	Modify	Create

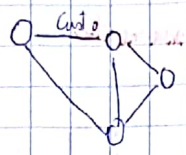
Scanned by CamScanner

Prática (06/04)

- Conceitos, algoritmos e protocolos de encaminhamento

(3) Como resolver o problema de encaminhamento?

- Podíamos criar um grafo, mas iríamos ter problemas de congestionamento, tal com outros



Logo, não podemos fazer isto, pois o grafo não pode ser estático, tem de ir sendo atualizado.

É necessário termos o conhecimento da topologia.

Serviço Rede:

✓ → Segurança — Confidencialidade / Integridade / Autenticação origem
Autenticação destino

✓ → Conectividade — Fim-a-fim

× [Garantia de entrega
× Ordem
× Garantia de processos (Qualidade)] } Serviço de circuito virtual

Estes 3 não foram implementados em serviço de rede, logo foi usado um serviço de "Datagramas"

↓

<u>Best-effort</u>, não há garantias

(4)

Ligar e funcionar logo

plug & play

Hubs ✓

Bridge ✓

Só trabalha a nível MAC ← Switch (L2) ✗, pois tem uma tabela de MAC addresses, que sabem o MAC de cada porta. Mas se for self-learning então plug & play

Trabalha a nível MAC e IP (todos os portos são iguais) ← Switch (L3) ✓, se tiver self-learning

Tem diagnósticos ← Routers ✗, primeiro pois é necessário configurar os IPs; depois, pode não necessário fazer routing

Isolamento de tráfego

Hubs ✗, não sabem nada

Bridge ✓

Switch (L2) ✓, a tabela MAC serve mesmo para isso

Switch (L3) ✓

Routers ✓

Encaminhamento

Hubs ✗ Bridge ✗ Switch (L2) ✗

Switch (L3) ✓ Router ✓

Store-and-Forward

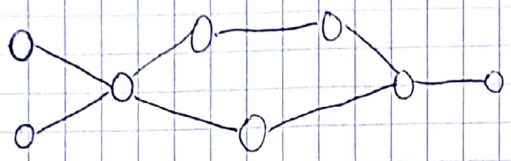
Hubs ✗ Bridge ✗ Switch (L2), tem de analisar pelo menos o MAC address ✗

Switch (L3) ✗ Router ✓, é obrigado a de fato colocar na porta e reencaminhar depois

Teórica (13/04)

(19)

Topologia "paxa"



Problemas de encaminhamento?

- Existem 2 rotas conhecidas, logo teremos de escolher a melhor rota

À partida a de baixo será a melhor rota, mas poderá não ser.

Soluções?

1) O ideal seria fazer um balanceamento de carga (19) entre as 2 rotas:

- Multipath → Igual Custo, é fácil de fazer, como OSTF
  → Custo diferente, é mais complicado

2) Search Routing → A origem define o caminho e coloca-o nos pacotes

É uma solução fácil de encaminhar, mas também fácil de fazer um ataque à rede (o que é mau, logo é perigoso)

3) Routing c/ QoS (15) → Normalmente a métrica é o nº de saltos, mas podemos mudar a métrica, de modo a distribuir o tráfego.

Vamos ter uma tabela de routing por cada métrica.

Ex: O de cima é a métrica de largura de banda, o de baixo já vai ser o caminho mais curto

4) SDN (Software Define Network) → O controlador da rede é que define qual a melhor rota a partir de cada router

Scanned by CamScanner

(25)

Para facilitar o Routing, é necessário um bom endereçamento

(27)

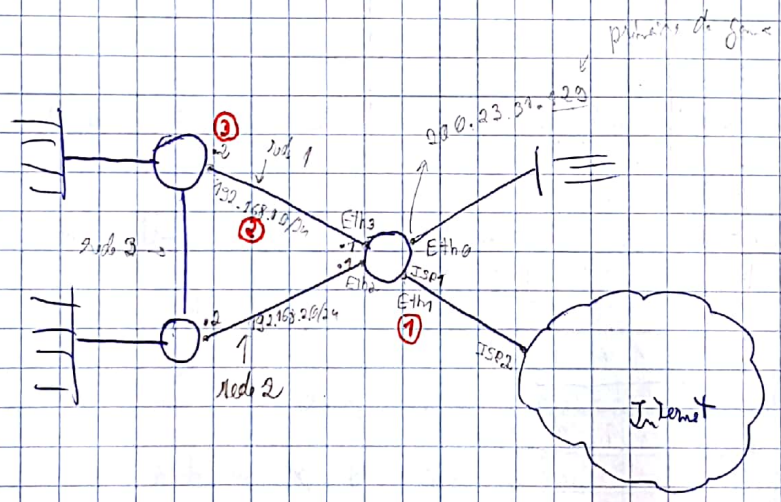
Seja IPv4 ou IPv6 temos a parte da rede e da host.
Mas o endereçamento é só feito na parte da rede.

(30)

Permite facilitar a tabela de encaminhamento (ficam mais curtas e mais eficientes)

(32)

Exercício:



1) O endereço do router vai ser sempre o 1º ou último endereço na gama da rede

Atribuir endereços: 3 redes

rede 1 → 192.168.1.0/24
rede 2 → 192.168.2.0/24
rede 3 → 192.168.3.0/24

Tabela de encaminhamento:

	Destino (Prefixo)	Int. Saída	Próximo Salto	Métrica (custo)
Default	0.0.0.0/0	ISP1	ISP2	1 ... N
Diretamente conectada	200.23.31.128/26	Eth0 200.23.23.129	—	1
agregações	200.23.16.0/22 200.23.20.0/22 200.23.24.0/23 200.23.26.0/23	Eth3 192.168.1.1	192.168.1.2	2
5 rotas agregações	200.24.28.0/24 29.0/24 30.0/24 31.0/26 31.64/26	Eth2 192.168.2.1	192.168.2.2	2

Nota: Agregação permite passar de 11 para 4 rotas

Prefixo comum

	128	64	32	16	8	4	2	1
200.23.(16.0) →	0	0	0	1	0	0	0	0
(20.0) →	0	0	0	1	0	1	0	0
(24.0) →	0	0	0	1	1	0	0	0
(26.0) →	0	0	0	1	1	0	1	0

200.23.16.0/20

Scanned by CamScanner

(33)

- <u>Inserção</u> : Fácil de fazer
- <u>Remoção</u> : Fácil de fazer

Não podemos fazer inserção e remoção ao mesmo tempo, por isso é necessário clonar a estrutura e fazer separadamente

- <u>Pesquisa</u> : Pior caso → Match com 32 bits
- <u>Memória</u> : Depende da implementação, se for com apontadores é caro, mas se for com uma estrutura implícita vai ocupar pouca memória.

Esta estrutura dá para implementar em hardware, e pode ser muito mais rápido do que com software.

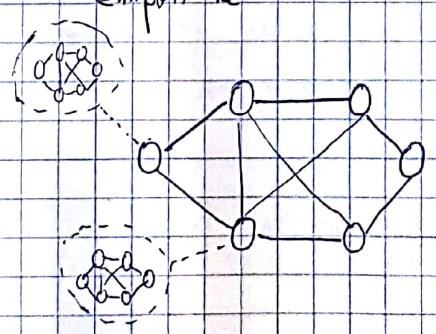
(38)

Assumo que é uma métrica <u>aditiva</u> (Os custos somam-se)

- Nº saltos? ✓ $\qquad C(x,z) = C(x,y) + C(y,z)$
- Custo monetário? ✓
- Delay ✓ $\qquad delay(x,z) = delay(x,y) + delay(y,z)$?
- Largura de banda ✗ (É côncava, utiliza outro tipo de função, como máximo ou mínimo)

(43)

Importante



Todos os nós conhecem a rede, depois de se conhecerem calculam Dijkstra

1) Conhecer a topologia
2) Calcular
3) Tabela

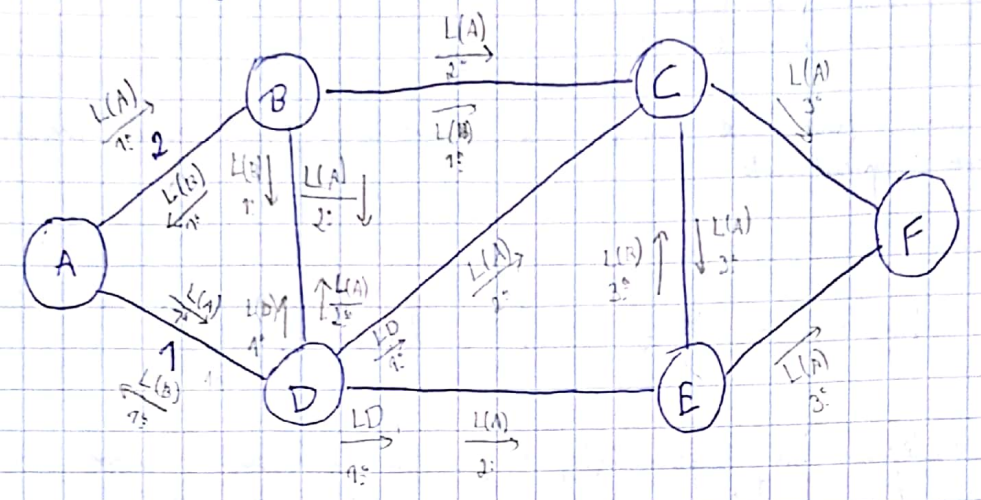
Vai criar uma LSA (Link State Advertisement) → É o perfil do router

Scanned by CamScanner

Teórica (20/04) → Teste

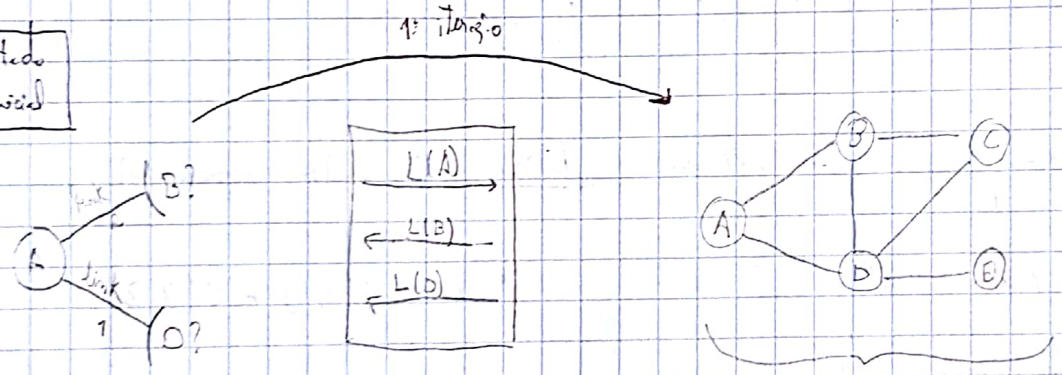
(44)

1. Inundação física



a) Nó A

Estado Inicial



Base de dados topológica no final da 1ª iteração

LSA_A (Link State Advertisement) = $\{(A \to B, 2), (A \to D, 1)\}$

Quantas iterações é necessário para se conhecer a topologia toda?

// Feito no desenho a lápis

São necessárias 3 iterações, é o caminho mais longo entre 2 nós.

Reliable Flooding (Algoritmo)

ESCOLHA

CASO (estado dos link mudam)

→ Cria $LSA_0()$

→ Envia LSA() a todos os vizinhos v

→ Para todos as ligações fazer vizinho_ACK(LSA(v)) = True;

CASO (Receba LSA(v)):

Se!(LSA(v) ∈ Topologia())

→ Envia LSA(v) a todos os vizinhos excepto v

→ Para todos os vizinhos excepto v

espere_ACK(LSA(v),) = True

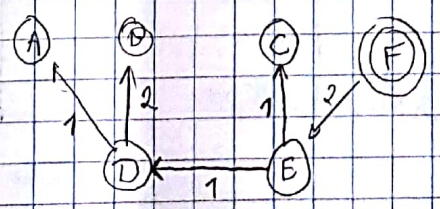
CASO (recebe_ACK(LSA(v)))

espere_ACK(LSA(v)) = false

Se (já recebi todos)

{ envia_ACK(ao, in)

(52) Arvore de caminhos mais curtos de (F)



Custo total arvore = 7//

Scanned by CamScanner

Visitados	D(A) P(A)	D(B) P(B)	D(C) P(C)	D(D) P(D)	D(E) P(E)
F	∞	∞	5, F	∞	2, F ✓
F, E	∞	∞	3, E	3, E	
F, E, D	4, D	5, D	3, E		
F, E, D, C	8, C (não atualizou) 4, D	6, C 5, D			
F, E, D, C, A		5, D			
$\boxed{F, E, D, C, A, B}$					

Fim do algoritmo

Scanned by CamScanner

Tabela Encaminhamento

Destino	Próx. salto	Custo	Interface
A	E	4	F → E
B	E	5	F → E
C	E	3	F → E
D	E	3	F → E
E	E	2	F → E

Teórica (27/04)

- Falta

Teórica (04/05)

(pág. 75)

	LS (Estado de ligação)	DV (Vector Distância)
Recursos: CPU, RAM, Largura Banda	↑↑ Muito elevado $O(N^2)$ ↑ Toda a Tipologia (N, M) ↑↑ Uma alteração link (N×M mensagens) ⇒ flooding toda a tipologia (sempre)	↓↓ Bastante reduzido ↓ Vetores de distância só dos vizinhos ↓↓ Só envia rotas aos vizinhos se houver alteração na tabela de melhores rotas
Congestionamento tráfego normal	Melhor controlo	Pode haver ciclos
Convergência	Muito elevado (Reliable flooding garante convergência rápida)	Pode ser lenta (contagem até infinito) (assíncrona)
Robustez	↑↑	↓↓

Scanned by CamScanner

// Fim deste módulo

→ Redes Definidas por Software (SDN)

Teórica (09/05)

→ Segurança em Redes

(2)

Ameaças

- Interceptar mensagens (recolher indevidamente informação)
  espionagem (passiva)
- Fazer-se passar por um dado (disfarce, roubo de identidade)
- Impossibilitar a comunicação: Mandar excesso de mensagens, spam, ...
  Negação de serviços
- Manipulação da comunicação: Inserção / remoção / alteração de dados
- Repúdio

Como combater estas ameaças?

Página 3

Resolve:

(3)

- Confidencialidade: → mecânismo → Interceção / Manipulação
- Integridade: → manipulação
- Escalável: → Negação do serviço
- Autenticação (origem, destino): → Roubo de Identidade / Disfarce
- Não repúdio (origem, destino) → Repúdio

Scanned by CamScanner

Teórica (25/05)

Algoritmos mais usados

- MD5 - Message Digest

Teste → Estudar basicamente os exercícios dos slides

Sobre esta matéria de segurança:

→ Apenas uma pergunta teórica (não muito importante)

↓

Está nos slides também

Scanned by CamScanner