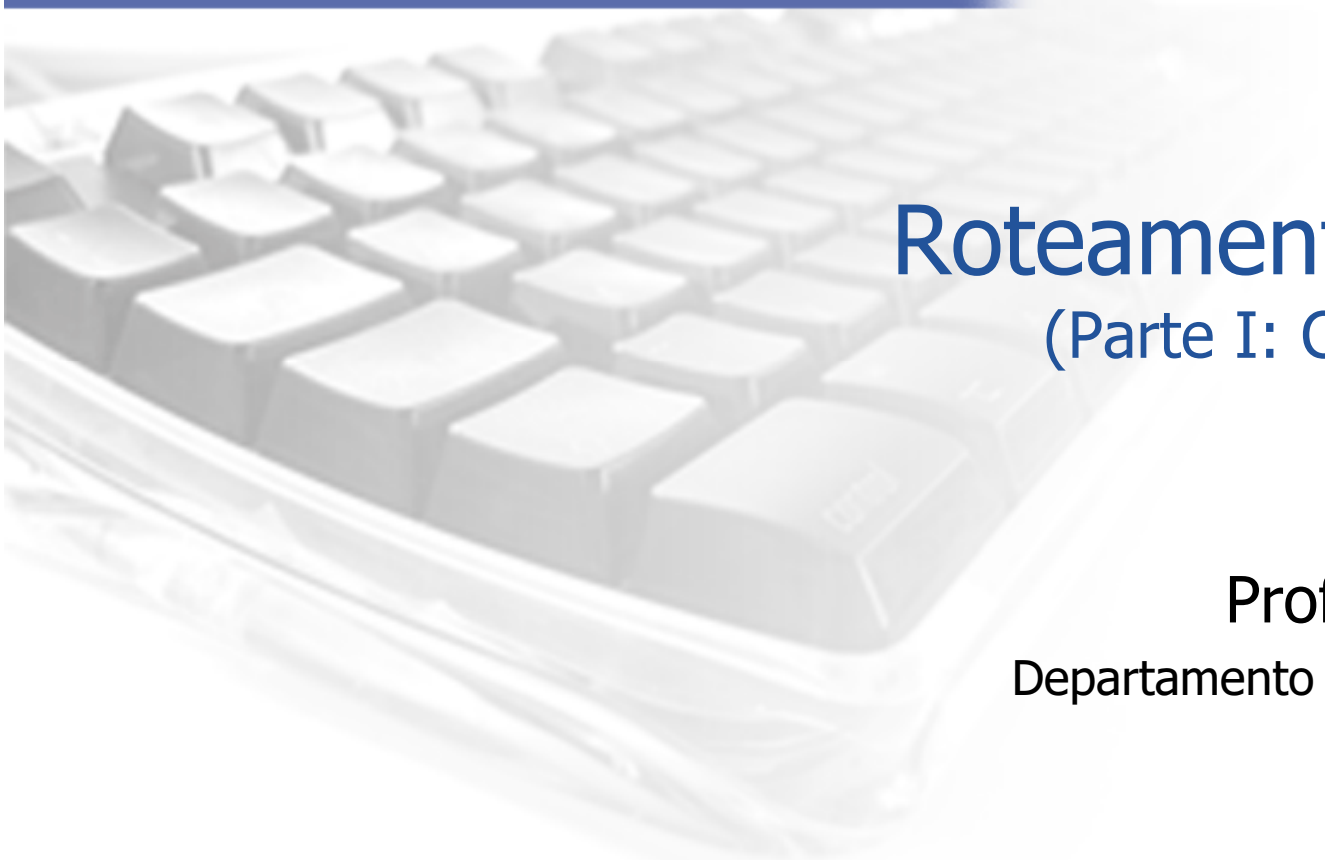
Laboratório de Pesquisa em Redes e Multimídia



Roteamento Dinâmico

(Parte I: Conceitos Básicos)

Prof. José Gonçalves

Departamento de Informática – UFES

zegonc@inf.ufes.br

Universidade Federal do Espírito Santo

Departamento de Informática

Roteamento Dinâmico

- Como visto, no roteamento estático as informações que um roteador precisa saber para poder encaminhar pacotes corretamente aos seus destinos são colocadas manualmente na tabela de rotas.
- Diferentemente, no roteamento dinâmico, os roteadores podem descobrir estas informações automaticamente e compartilhá-la com outros roteadores via protocolos de roteamento dinâmicos.
- Um protocolo de roteamento dinâmico é uma linguagem que um roteador fala com outros roteadores a fim de compartilhar informações sobre alcançabilidade e estado das redes.
- Protocolos de roteamento dinâmico permitem determinar o próximo melhor caminho para um destino se o atual torna-se inacessível devido à queda de um link ou se uma região fica inacessível em virtude do congestionamento.

Roteamento Dinâmico (cont.)

- Esta capacidade de se adaptar (compensar) às mudanças de topologia é a vantagem mais importante que o roteamento dinâmico oferece quando comparado ao roteamento estático.
- Apesar das suas vantagens:
  - Protocolos de roteamento dinâmico criam tráfego extra na rede.
  - Podem ocorrer loops de pacotes enquanto a informação de roteamento está sendo trocada entre os roteadores
  - Enquanto os roteadores tentam entender o que está acontecendo pacotes para um mesmo destino podem ser enviados por rotas diferentes e links bidirecionais podem ser tratados de forma distinta, confundindo o gerenciamento da rede e as aplicações de rede em execução.

Roteamento Dinâmico (cont.)

- Todo protocolo de roteamento dinâmico é construído sobre um algoritmo.
- O algoritmo de roteamento deve, no mínimo, especificar o seguinte:
  - Um procedimento para passar informação de alcançabilidade de redes a outros roteadores;
  - Um procedimento para receber informação de alcançabilidade de outros roteadores;
  - Um procedimento para determinar rotas ótimas baseado na informação de alcançabilidade disponível e para armazenar essa informação na tabela de rotas;
  - Um procedimento para reagir a, compensar e divulgar mudanças de topologia em uma internet.

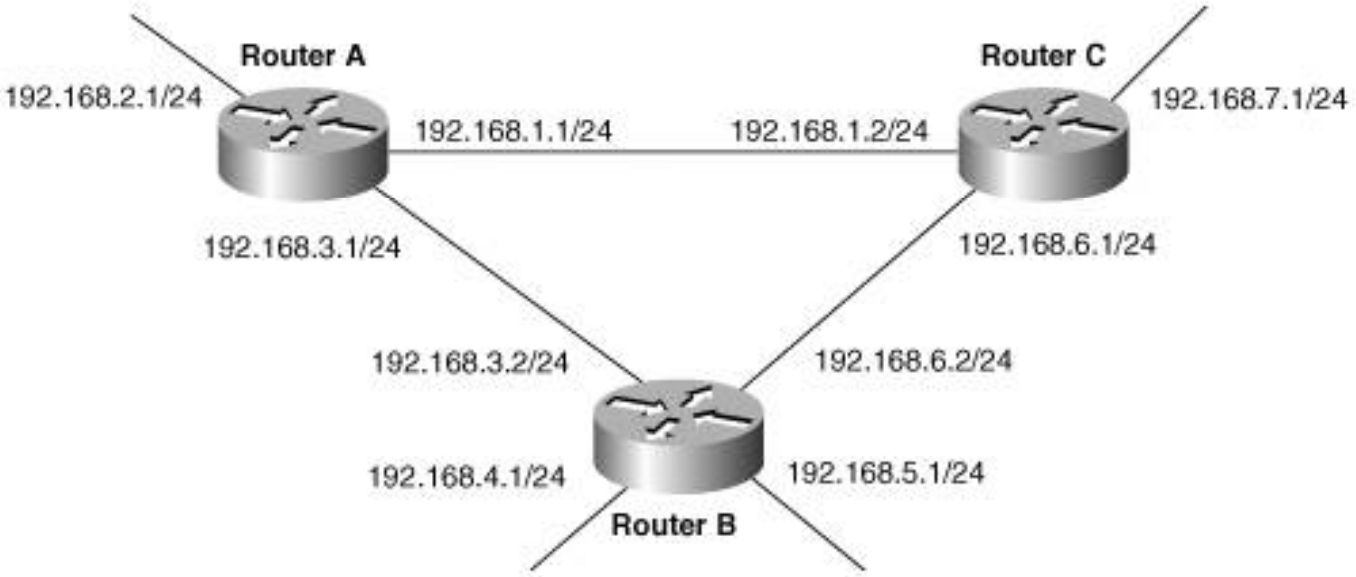
Roteamento Dinâmico (cont.)

- A maioria dos protocolos de roteamento dinâmico caem em uma das duas categorias: *distance vector* ou *link state*.
- Questões comuns a qualquer algoritmo de roteamento:
  - Determinação de caminho (*path determination*)
  - Métrica (*metrics*)
  - Número de saltos (*hop count*)
  - Largura de banda (*bandwidth*)
  - Carga (*load*)
  - Retardo (*delay*)
  - Alcançabilidade (*reachability*)
  - Custo (*cost*)
  - Convergência (*convergence*)

Determinação de Caminho

- Os endereços das interfaces de rede nos roteadores são os pontos de origem de informações de alcançabilidade.
- Cada roteador tem conhecimento das redes a ele diretamente conectadas pelos endereços e máscaras a elas assinaladas.
- Cada interface implementa os protocolos das camadas de enlace e física da rede à qual está conectada; logo o roteador também sabe o estado da rede (*up* ou *down*).

Exemplo 1


Router A
Router C
Router B
192.168.2.1/24
192.168.1.1/24
192.168.1.2/24
192.168.7.1/24
192.168.3.1/24
192.168.6.1/24
192.168.3.2/24
192.168.6.2/24
192.168.4.1/24
192.168.5.1/24

Exemplo 1 (cont.)

- Roteador A examina os seus endereços IP e máscaras associadas e deduz que ele está conectado às redes 192.168.1.0, 192.168.2.0 e 192.168.3.0.
- Roteador A registra essas redes na sua tabela de rotas, juntamente com algum flag indicando que as redes são diretamente conectadas.
- Roteador A coloca essa informação em um pacote:
  - "Minhas redes diretamente conectadas são x, y e z"
- Roteador A transmite cópias deste pacote (*routing updates*) para os roteadores B e C.

Exemplo 1 (cont.)

- Roteadores B e C seguem o mesmo procedimento, enviando pacotes de *update* com informação de suas redes diretamente conectadas para o roteador A.
- Roteador A registra as informações recebidas na sua tabela de rotas, juntamente com o endereço (da interface) do roteador que enviou o pacote de *update*.
- Roteador A sabe agora sobre todas as redes e os endereços dos roteadores às quais elas estão conectadas.

Questões a Considerar...

- O que o roteador A deve fazer com os *updates* recebidos de B e C após registrá-los na sua tabela de rotas? Deve ele passar as informações de roteamento recebidas de B para o roteador C e de C para o roteador B?
- Se A não encaminha os *updates* o compartilhamento não é completo. Por exemplo, se o link entre B e C não existe então cada um desses roteadores não saberá sobre as redes do outro. Logo, A deve encaminhar os *updates.*
  - Mas isso abre um novo leque de problemas...
- Se A ouve sobre a rede 192.168.4.0 de ambos os roteadores B e C, qual deles deve ser usado para atingir essa rede? São ambos válidos? Qual é o melhor caminho?

Questões a Considerar...(cont.)

- Que mecanismo será usado para garantir que todos os roteadores receberão todas as informações de roteamento, ao mesmo tempo que se prevenirá que pacotes de *update* fiquem circulando eternamente na internet?
- Roteadores compartilham algumas redes diretamente conectadas (192.168.1.0, 192.168.3.0, 192.168.6.0).
  - Devem os roteadores anunciar essas redes também?
- Conclusão: protocolos de roteamento dinâmicos são complexos!

Métricas

- Quando existem múltiplas rotas para o mesmo destino o roteador necessita de um mecanismo para determinar o melhor caminho.
- A *métrica* é uma variável assinalada às rotas como um meio de qualificá-las de melhor a pior enlace ou de rota mais preferida à menos preferida.
- Assim, a métrica é uma forma de qualificar as alternativas de rotas.
- Diferentes protocolos de roteamento usam diferentes, e às vezes múltiplas, métricas.
  - RIP: menor número de router hops
  - IGRP: combinação de menor largura de banda e delay total da rota.

Hop Count

- Métrica simples que contabiliza quantos roteadores existem entre a origem e o destino.
- No exemplo 1, do roteador A para a rede 192.168.5.0 a métrica é 1 hop se os pacotes são transmitidos pela interface 192.168.3.1 (roteador B) e 2 hops se enviados pela interface 192.168.1.1 (roteadores C e B).
- O link A-B é mesmo o melhor? E se esse for um link DS-0 (64 Kbps) e A-C e C-B são links T-1 (1544 Mbps)? Nesse caso, a largura de banda faria a diferença e a rota de 2 hops seria mais eficiente.

Bandwith

- Essa métrica leva em consideração a capacidade do *link* em transmitir dados. Quanto maior a largura de banda melhor.
- Entretanto, a largura de banda por si só pode não ser uma boa métrica:
  - O que acontece se os links T1 estiverem muito carregados?
  - Ou se o link de maior banda também apresentar maior delay?

Carga

- Essa métrica reflete a quantidade de tráfego passando por um link. Quanto menor a carga melhor.
- Diferentemente das duas métricas anteriores, a carga nas rotas varia e, portanto, a métrica também variará.
  - Se a métrica varia muito freqüentemente então uma mudança freqüente de rotas preferidas pode ocorrer (*route flapping*)
  - *Route flapping* provoca efeitos adversos no consumo de CPU do roteador, na largura de banda dos links e na estabilidade geral da rede.

Delay

- Medida que reflete o tempo gasto por um pacote para atravessar um caminho.
- Essa métrica pode considerar não apenas o tempo de retardo do enlace como também outros tempos envolvidos na transmissão dos pacotes (latência do roteador, tempo de enfileiramento, etc.).
- O delay de uma rota pode não ser efetivamente uma medida mas sim a soma de quantias estáticas definidas para cada interface ao longo do caminho.
  - Cada medida individual poderia ser uma estimativa baseada no tipo do link no qual a interface está conectada.

Confiabilidade

- Métrica que reflete a probabilidade de falha do link. Pode ser fixa ou variável.
  - Variável: número de vezes que o link falhou ou o número de erros recebidos em um certo intervalo de tempo.
  - Fixo: baseado na qualidade do link, determinada pelo administrador da rede.

Custo

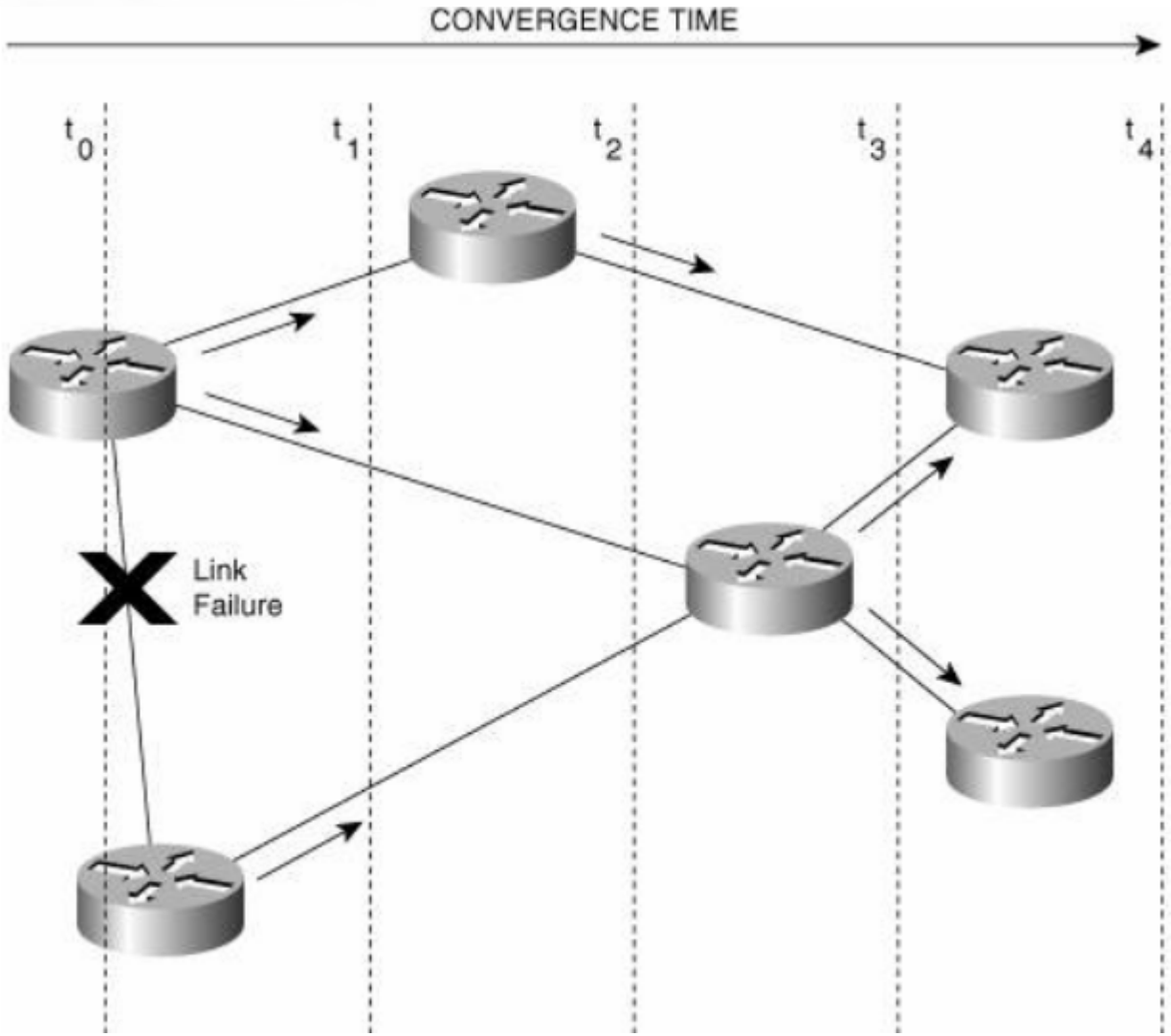
- Métrica utilizada pelo administrador da rede para forçar a escolha de determinada rota.
  - Reflete a decisão do administrador sobre rotas com maior ou menor preferência.
- O termo "custo" é freqüentemente usado como um termo genérico quando se trata de rotas.
  - "RIP chooses the lower-cost path based on hop count"
- Idem para "shortest" ou "longest":
  - "RIP chooses the shortest path based on hop count"

Convergência

- Informações de alcançabilidade nas tabelas de rotas de todos os roteadores da internet devem estar consistentes.
- É necessário um procedimento dentro do algoritmo de roteamento que garante a formação de tabelas de rotas consistentes e corretas de acordo com a topologia atual da rede.
  - Inconsistência: no exemplo 1, roteador A determina que o melhor caminho para a rede 192.168.5.0 é via roteador C e o roteador C determina que o melhor caminho para a mesma rede é pelo roteador A (*routing loop*).
- O processo de se trazer todas as tabelas de rotas a um estado de consistência é chamado de "convergência".
- O tempo que se gasta para atingir esta estabilidade é chamado de "tempo de convergência".

Convergência

No instante t_2 os três roteadores mais à esquerda já reconhecem a mudança da topologia enquanto que os três mais à direita ainda não receberam esta informação. Os roteadores com informação desatualizada continuarão a rotear os pacotes normalmente, sem conhecimento do problema. É durante esse estado intermediário que erros de roteamento podem ocorrer. O *tempo de convergência* é, portanto, um fator importante em qualquer protocolo de roteamento dinâmico.



Balanceamento de carga

- Refere-se à prática de distribuir carga entre os múltiplos caminhos para um mesmo destino, com o objetivo de se usar a largura de banda de forma eficiente.
- Na Figura 4.1, se um host na rede 192.168.2.0 envia conjunto de pacotes para um host na rede 192.168.6.0, o roteador A pode enviá-lo tanto via roteador B quanto pelo roteador C. Em ambos os casos, a rede esta a 1 hop de distância.
- Entretanto, enviar todos os pacotes por uma única rota não é a escolha mais eficiente do ponto de vista de uso da largura de banda disponível.
- Assim, balanceamento de carga deve ser implementado para alternar o tráfego entre os dois caminhos.
  - Pode-se usar custo igual ou custo desigual;
  - Pode-se usar balanceamento por destino ou por pacote.

Distance Vector

- Baseado nos trabalhos de E. Bellman, L. R. Ford e D. R. Fulkerson.
  - R. E. Bellman. Dynamic Programming. Princeton, New Jersey: Princeton University Press; 1957.
  - L. R. Ford Jr. and D. R. Fulkerson. Flows in Networks. Princeton, New Jersey: Princeton University Press; 1962.
- O nome vem do fato das rotas serem divulgadas como vetores de (distância, direção).
  - Distância = métrica Direção = next-hop
- Principais representantes: RIP v1/v2 e IGRP.
- Compartilha tudo que sabe mas apenas com os vizinhos (aprende dos vizinhos e divulga para os vizinhos).

Características Principais

- Um típico protocolo de roteamento *distance vector* usa um algoritmo de roteamento em que os roteadores periodicamente enviam "*routing updates*" para todos os vizinhos através do broadcast de toda a sua tabela de rotas.
  - Uma exceção é o EIGRP, da Cisco, cujos updates não são periódicos, não há broadcast e nem toda a tabela é transmitida.

Características Principais (cont.)

- No início, a tabela de roteamento de um gateway apresenta apenas os endereços das redes diretamente conectadas a ele, sendo a distância igual a zero.
- Periodicamente os gateways enviam cópias das suas tabelas de roteamento para todos os gateways alcançados diretamente (os vizinhos), sendo assim feita a atualização das tabelas.

Exemplo

Tabela de Rotas de G1

Destino	Distância	Rota
Rede 1	0	Direto
Rede 2	0	Direto
Rede 4	8	G3
Rede 17	5	G4
Rede 24	6	G2
Rede 30	2	G6
Rede 42	2	G2

Informação de G2

Destino	Distância
Rede 1	2
Rede 4	3
Rede 17	6
Rede 21	4
Rede 24	5
Rede 30	10
Rede 42	3

Destino	**Distância**	**Rota**
Rede 1	0	Direto
Rede 2	0	Direto
Rede 4	4	G2
Rede 17	5	G4
Rede 24	6	G2
Rede 30	2	G6
Rede 42	2	G2
Rede 21	5	G2

Periodic Updates

- Atualizações periódicas são enviadas ao final de um certo período de tempo (ex: AppleTalk's RTMP = 10s, Cisco IGRP = 90s).
- Se muito frequentes, congestionamento e overloading da CPU do roteador podem ocorrer; se pouco frequentes, o tempo de convergência pode ser inaceitavelmente alto.

Vizinhança

- No contexto dos roteadores, vizinhos referem-se a roteadores que compartilham um enlace de dados ou algum outro nível mais alto de adjacência lógica.
- Um protocolo de roteamento *distance vector* envia os seus *updates* para a os roteadores vizinhos e depende deles para passar informações de *update* para a vizinhança. Por esta razão, roteamento *distance vector* é dito usar updates *hop-by-hop*.

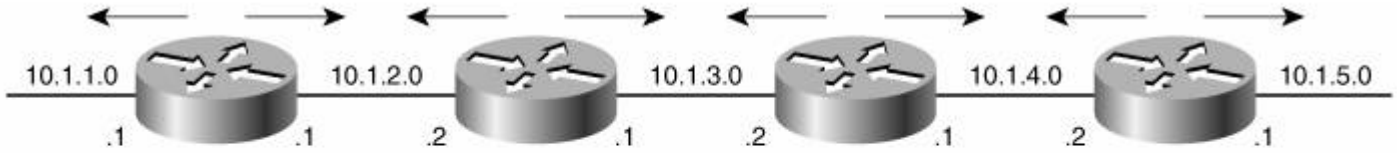
Broadcast Updates

- Quando um roteador se torna ativo pela primeira vez na rede ele envia *updates* para o endereço de broadcast da rede (no caso do IP, 255.255.255.255).
- Roteadores vizinhos falando o mesmo protocolo ouvirão o broadcast e tomarão as ações devidas. Hosts e outros equipamentos não interessados nesses updates simplesmente descartam os pacotes.

Full Routing Table Updates

- A maioria dos protocolos de roteamento distance vector simplesmente falam para os seus vizinhos tudo o que eles sabem, dando um broadcast na sua tabela de roteamento completa, com algumas exceçcões.
- Os vizinhos ao receberem esses updates examinam a tabela, pegando as informações que eles precisam e descartando o restante.

Routing by Rumor



	Router A			Router B			Router C			Router D		
t_0	NET	VIA	HOPS	NET	VIA	HOPS	NET	VIA	HOPS	NET	VIA	HOPS
	10.1.1.0	--	0	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0
	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0	10.1.5.0	--	0
t_1	NET	VIA	HOPS	NET	VIA	HOPS	NET	VIA	HOPS	NET	VIA	HOPS
	10.1.1.0	--	0	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0
	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0	10.1.5.0	--	0
	10.1.3.0	10.1.2.2	1	10.1.1.0	10.1.2.1	1	10.1.2.0	10.1.3.1	1	10.1.3.0	10.1.4.1	1
				10.1.4.0	10.1.3.2	1	10.1.5.0	10.1.4.2	1			
t_2	NET	VIA	HOPS	NET	VIA	HOPS	NET	VIA	HOPS	NET	VIA	HOPS
	10.1.1.0	--	0	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0
	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0	10.1.5.0	--	0
	10.1.3.0	10.1.2.2	1	10.1.1.0	10.1.2.1	1	10.1.2.0	10.1.3.1	1	10.1.3.0	10.1.4.1	1
	10.1.4.0	10.1.2.2	2	10.1.4.0	10.1.3.2	1	10.1.5.0	10.1.4.2	1	10.1.2.0	10.1.4.1	2
				10.1.5.0	10.1.3.2	2	10.1.1.0	10.1.3.1	2			
t_3	NET	VIA	HOPS	NET	VIA	HOPS	NET	VIA	HOPS	NET	VIA	HOPS
	10.1.1.0	--	0	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0
	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0	10.1.5.0	--	0
	10.1.3.0	10.1.2.2	1	10.1.1.0	10.1.2.1	1	10.1.2.0	10.1.3.1	1	10.1.3.0	10.1.4.1	1
	10.1.4.0	10.1.2.2	2	10.1.4.0	10.1.3.2	1	10.1.5.0	10.1.4.2	1	10.1.2.0	10.1.4.1	2
	10.1.5.0	10.1.2.2	3	10.1.5.0	10.1.3.2	2	10.1.1.0	10.1.3.1	2	10.1.1.0	10.1.4.1	3

Routing by Rumor (cont.)

At time t1, the first updates have been received and processed by the routers. Look at Router A's table at t1. Router B's update to Router A said that Router B can reach networks 10.1.2.0 and 10.1.3.0, both zero hops away. If the networks are zero hops from B, they must be one hop from A. Router A incremented the hop count by one and then examined its route table. It already recognized 10.1.2.0, and the hop count (zero) was less than the hop count B advertised, (one), so A disregarded that information.

Network 10.1.3.0 was new information, however, so A entered this in the route table. The source address of the update packet was Router B's interface (10.1.2.2) so that information is entered along with the calculated hop count.

Notice that the other routers performed similar operations at the same time t1. Router C, for instance, disregarded the information about 10.1.3.0 from B and 10.1.4.0 from C but entered information about 10.1.2.0, reachable via B's interface address 10.1.3.1, and 10.1.5.0, reachable via C's interface 10.1.4.2. Both networks were calculated as one hop away.

At time t2, the update period has again expired and another set of updates has been broadcast. Router B sent its latest table; Router A again incremented B's advertised hop counts by one and compared. The information about 10.1.2.0 is again discarded for the same reason as before. 10.1.3.0 is already known, and the hop count hasn't changed, so that information is also discarded. 10.1.4.0 is new information and is entered into the route table.

The network is converged at time t3. Every router recognizes every network, the address of the next-hop router for every network, and the distance in hops to every network.

Distance vector algorithms provide road signs to networks. They provide the direction and the distance, but no details about what lies along the route. And like the sign at the fork in the trail, they are vulnerable to accidental or intentional misdirection.

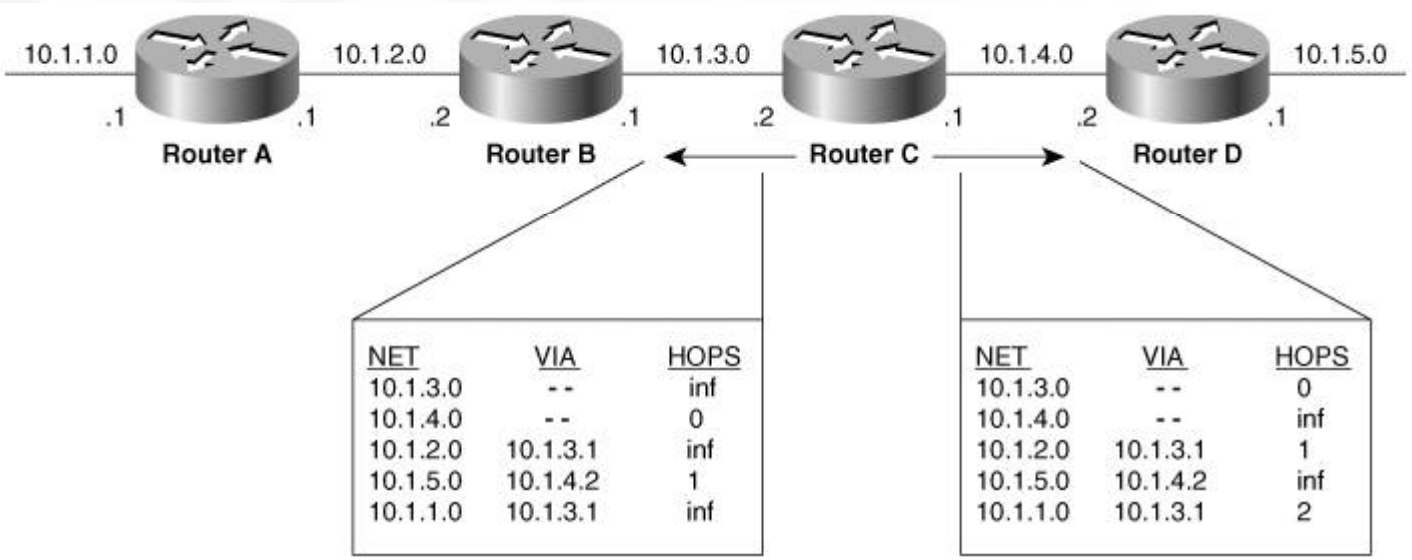
Route Invalidation Timers

- Now that the network in Figure 4-3 is fully converged, how will it handle reconvergence when some part of the topology changes? If network 10.1.5.0 goes down, the answer is simple enoughRouter D, in its next scheduled update, flags the network as unreachable and passes the information along.
- But what if, instead of 10.1.5.0 going down, Router D fails? Routers A, B, and C still have entries in their route tables about 10.1.5.0; the information is no longer valid, but there's no router to inform them of this fact. They will unknowingly forward packets to an unreachable destinationa black hole has opened in the network.
- This problem is handled by setting a route invalidation timer for each entry in the route table. For example, when Router C first hears about 10.1.5.0 and enters the information into its route table, C sets a timer for that route. At every regularly scheduled update from Router D, C discards the update's already-known information about 10.1.5.0 as described in "Routing by Rumor." But as C does so, it resets the timer on that route.
- If Router D goes down, C will no longer hear updates about 10.1.5.0. The timer will expire, C will flag the route as unreachable and will pass the information along in the next update.
- Typical periods for route timeouts range from three to six update periods. A router would not want to invalidate a route after a single update has been missed, because this event might be the result of a corrupted or lost packet or some sort of network delay. At the same time, if the period is too long, reconvergence will be excessively slow.

Split Horizon

- Every network known by Router A in Figure 4-3, with a hop count higher than zero, has been learned from Router B. Common sense suggests that for Router A to broadcast the networks it has learned from Router B back to Router B is a waste of resources. Obviously, B already "knows" about those networks A route pointing back to the router from which packets were received is called a reverse route. Split horizon is a technique for preventing reverse routes between two routers.
- Besides not wasting resources, there is a more important reason for not sending reachability information back to the router from which the information was learned. The most important function of a dynamic routing protocol is to detect and compensate for topology changes if the best path to a network becomes unreachable, the protocol must look for a next-best path.
- Look yet again at the converged network of Figure 4-3 and suppose that network 10.1.5.0 goes down. Router D will detect the failure, flag the network as unreachable, and pass the information along to Router C at the next update interval. However, before D's update timer triggers an update, something unexpected happens. C's update arrives, claiming that it can reach 10.1.5.0, one hop away! Remember the road sign analogy? Router D has no way of recognizing that C is not advertising a legitimate next-best path. It will increment the hop count and make an entry into its route table indicating that 10.1.5.0 is reachable via Router C's interface 10.1.4.1, just two hops away.
- Now a packet with a destination address of 10.1.5.3 arrives at Router C, which consults its route table and forwards the packet to D. Router D consults its route table and forwards the packet to C, C forwards it back to D, ad infinitum. A routing loop has occurred. Implementing split horizon prevents the possibility of such a routing loop. There are two categories of split horizon: simple split horizon and split horizon with poisoned reverse.
- The rule for simple split horizon is, when sending updates out a particular interface, do not include networks that were learned from updates received on that interface.
- The routers in Figure 4-4 implement simple split horizon. Router C sends an update to Router D for networks 10.1.1.0, 10.1.2.0, and 10.1.3.0. Networks 10.1.4.0 and 10.1.5.0 are not included because they were learned from Router D. Likewise, updates to Router B include 10.1.4.0 and 10.1.5.0 with no mention of 10.1.1.0, 10.1.2.0, and 10.1.3.0..

Split Horizon (cont.)

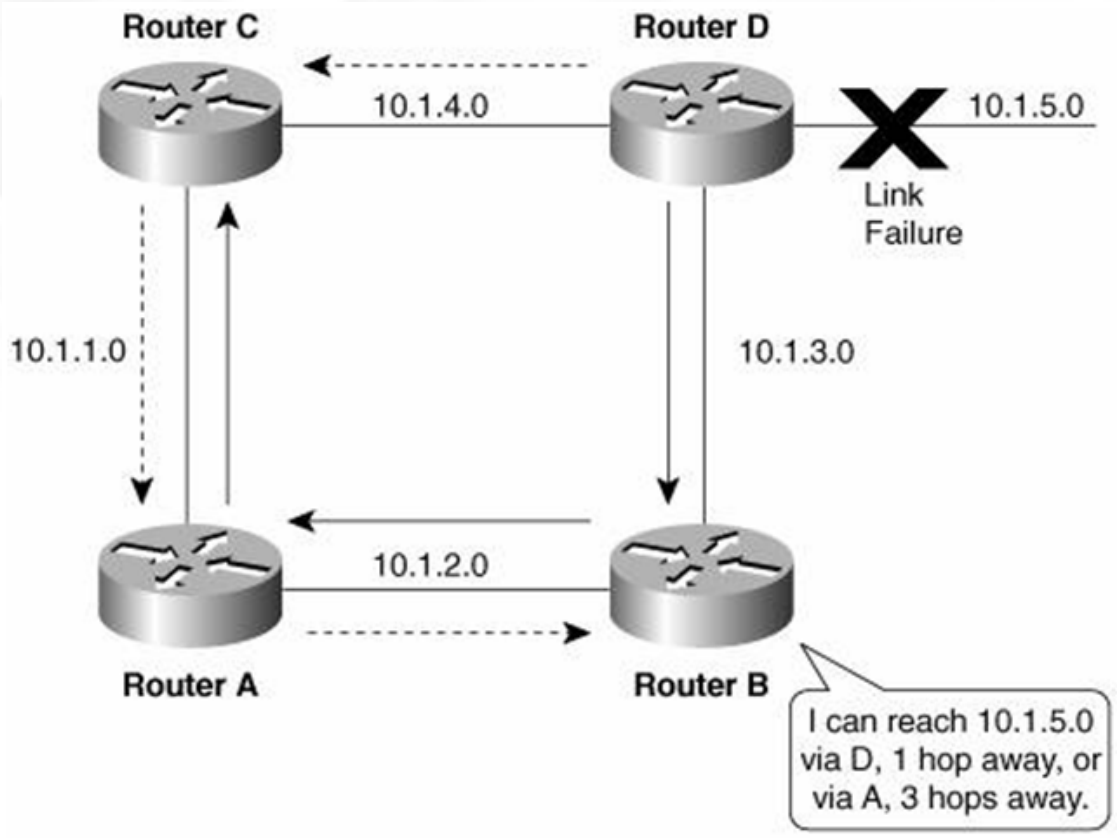


The routers in Figure 4-4 implement simple split horizon. Router C sends an update to Router D for networks 10.1.1.0, 10.1.2.0, and 10.1.3.0. Networks 10.1.4.0 and 10.1.5.0 are not included because they were learned from Router D. Likewise, updates to Router B include 10.1.4.0 and 10.1.5.0 with no mention of 10.1.1.0, 10.1.2.0, and 10.1.3.0.

Counting to Infinity

- Split horizon will break loops between neighbors, but it will not stop loops in a network such as the one in Figure 4-6. Again, 10.1.5.0 has failed. Router D sends the appropriate updates to its neighbors, Router C (the dashed arrows), and Router B (the solid arrows). Router B marks the route via D as unreachable, but Router A is advertising a next-best path to 10.1.5.0, which is three hops away. B posts that route in its route table.
- B now informs D that it has an alternative route to 10.1.5.0. D posts this information and updates C, saying that it has a four-hop route to the network. C tells A that 10.1.5.0 is five hops away. A tells B that the network is now six hops away.
- "Ah," Router B thinks, "Router A's path to 10.1.5.0 has increased in length. Nonetheless, it's the only route I've got, so I'll use it!"
- B changes the hop count to seven, updates D, and around it goes again. This situation is the counting-to-infinity problem because the hop count to 10.1.5.0 will continue to increase to infinity. All routers are implementing split horizon, but it doesn't help.
- The way to alleviate the effects of counting to infinity is to define infinity. Most distance vector protocols define infinity to be 16 hops. As the updates continue to loop among the routers in Figure 4-6, the hop count to 10.1.5.0 in all routers will eventually increment to 16. At that time, the network will be considered unreachable.
- This method is also how routers advertise a network as unreachable. Whether it is a poisoned reverse route, a network that has failed, or a network beyond the maximum network diameter of 15 hops, a router will recognize any 16-hop route as unreachable.
- Setting a maximum hop count of 15 helps solve the counting-to-infinity problem, but convergence will still be very slow. Given an update period of 30 seconds, a network could take up to 7.5 minutes to reconverge and is susceptible to routing errors during this time. Triggered updates can be used to reduce this convergence time.

Counting to Infinity (cont.)



Triggered Updates

- Triggered updates, also known as flash updates, are very simple: If a metric changes for better or for worse, a router will immediately send out an update without waiting for its update timer to expire. Reconvergence will occur far more quickly than if every router had to wait for regularly scheduled updates, and the problem of counting to infinity is greatly reduced, although not completely eliminated.
- Regular updates might still occur along with triggered updates. Thus a router might receive bad information about a route from a not-yet-reconverged router after having received correct information from a triggered update. Such a situation shows that confusion and routing errors might still occur while a network is reconverging, but triggered updates will help to iron things out more quickly.
- A further refinement is to include in the update only the networks that actually triggered it, rather than the entire route table. This technique reduces the processing time and the impact on network bandwidth.

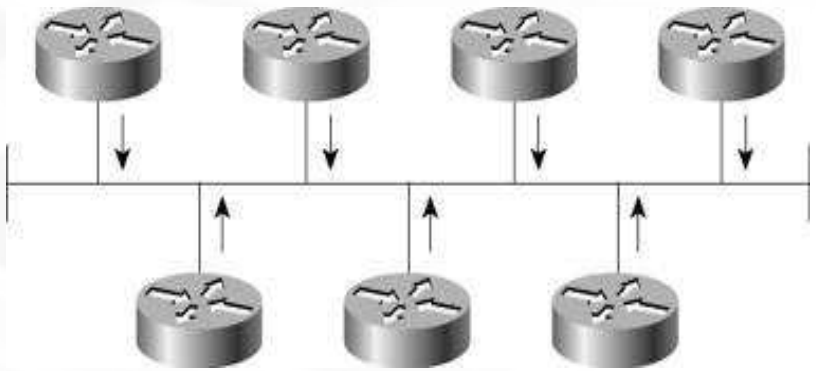
Holddown Timers

- Triggered updates add responsiveness to a reconverging network. Holddown timers introduce a certain amount of skepticism to reduce the acceptance of bad routing information.
- If the distance to a destination increases (for example, the hop count increases from two to four), the router sets a holddown timer for that route. Until the timer expires, the router will not accept any new updates for the route.
- Obviously, a trade-off is involved here. The likelihood of bad routing information getting into a table is reduced but at the expense of the reconvergence time. Like other timers, holddown timers must be set with care. If the holddown period is too short, it will be ineffective, and if it is too long, normal routing will be adversely affected.

Asynchronous Updates

- Figure 4-7 shows a group of routers connected to an Ethernet backbone. The routers should not broadcast their updates at the same time; if they do, the update packets will collide. Yet this situation is exactly what can happen when several routers share a broadcast network. System delays related to the processing of updates in the routers tend to cause the update timers to become synchronized. As a few routers become synchronized, collisions will begin to occur, further contributing to system delays, and eventually all routers sharing the broadcast network might become synchronized
- Asynchronous updates might be maintained by one of two methods:
  - Each router's update timer is independent of the routing process and is, therefore, not affected by processing loads on the router.
  - A small random time, or timing jitter, is added to each update period as an offset.
- If routers implement the method of rigid, system-independent timers, all routers sharing a broadcast network must be brought online in a random fashion. Rebooting the entire group of routers simultaneously such as might happen during a widespread power outage, for example, could result in all the timers attempting to update at the same time.
- Adding randomness to the update period is effective if the variable is large enough in proportion to the number of routers sharing the broadcast network. Floyd and Jacobson have calculated that a too-small randomization will be overcome by a large enough network of routers, and that to be effective the update timer should range up to 50 percent of the median update period.

Asynchronous Updates (cont.)



If update timers become synchronized, collisions might occur.