Esse é o primeiro de uma série de artigos sobre a teoria e o funcionamento das redes de computadores.

Nestes artigos iremos abordar vários assuntos, desde o básico teórico ao intermediário prático.

Introdução

O que são redes de computadores?

subentende-se por redes de computadores um conjunto de computadores autônomos interconectados, que podem trocar informações por meio de uma conexão por fio de cobre, fibras ópticas, microondas, ondas de infravermelho ou satélites de comunicação. Existem redes em muitos tamanhos, modelos e formas.

Origem das redes de computadores

As redes de dados foram um resultado dos aplicativos empresariais que foram escritos para microcomputadores. Naquela época os microcomputadores não eram conectados portanto não havia uma maneira eficiente de compartilhar dados entre vários microcomputadores. Tornou-se óbvio que o compartilhamento de dados através da utilização de disquetes não era uma maneira eficiente e econômica de se administrar empresas. Os "Sneakernets", como este compartilhamento era chamado, criavam várias cópias dos dados. Cada vez que um arquivo era modificado ele teria que ser compartilhado novamente com todas as outras pessoas que precisavam daquele arquivo. Se duas pessoas modificavam o arquivo e depois tentavam compartilhá-lo, um dos conjuntos de modificações era perdido. As empresas precisavam de uma solução que respondesse satisfatoriamente às três questões abaixo:

- Como evitar a duplicação de equipamentos e recursos
- Como se comunicar eficazmente
- Como configurar e gerenciar uma rede

As empresas perceberam que a tecnologia de rede aumentaria a produtividade enquanto lhes economizaria dinheiro. Novas redes foram sendo criadas ou expandidas tão rapidamente quanto surgiam novos produtos e tecnologias de rede. No início dos anos 80, houve uma grande expansão no uso de redes, apesar da desorganização na primeira fase de desenvolvimento.

As tecnologias de rede que surgiram tinham sido criadas usando diferentes implementações de hardware e software. Cada empresa que criava hardware e software para redes usava seus próprios padrões. Estes padrões individuais eram desenvolvidos devido à competição com outras companhias. Conseqüentemente, muitas das novas tecnologiasde rede eram incompatíveis umas com as outras. Tornou-se cada vez mais difícil para as redes que usavam especificações diferentes se comunicarem entre si. Freqüentemente era necessário que o equipamento antigo de rede fosse removido para que fosse implementado o novo equipamento.

Dispositivos de Rede

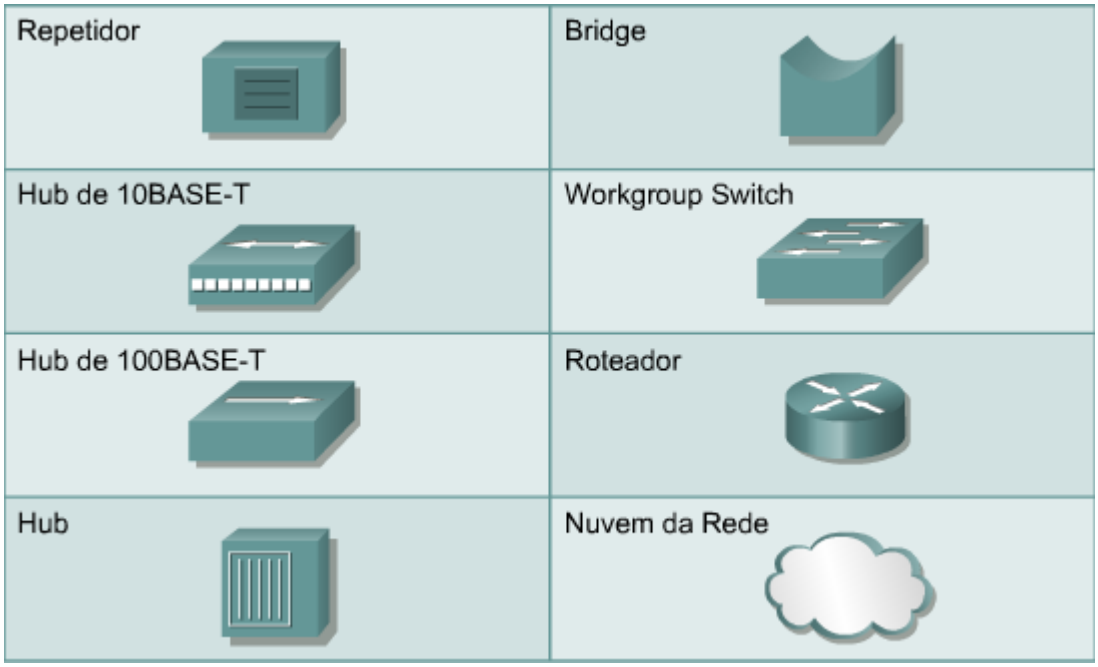
Os equipamentos que se conectam diretamente a um segmento de rede são chamados de dispositivos. A rede é composta por diversos dispositivos. Estes dispositivos são divididos em duas classificações:

- Dispositivos de usuário final - Incluem computadores, impressoras, scanners e outros dispositivos que fornecem serviços diretamente ao usuário.
- Dispositivos de rede - Incluem todos os dispositivos que fazem a interconexão de todos os dispositivos do usuário final permitindo que se comuniquem.

Dispositivos de usuário final:

Fornecem aos usuários uma conexão à rede são e também conhecidos como hosts. Estes dispositivos permitem que os usuários compartilhem, criem e obtenham informações. Os hosts são fisicamente conectados aos meios de rede usando uma placa de rede (NIC - Network Interface Card). Cada placa de rede individual transporta um indentificador exclusivo, denominado endereço de Controle de Acesso ao Meio (MAC - Media Access Control). Este endereço é usado para controlar as comunicações de dados do host na rede.

Os dispositivos de rede possuem uma representação padrão mostrada na imagem abaixo:



Repetidor
É um dispositivo de rede usado para regenerar um sinal. Os repetidores regeneram os sinais analógicos e digitais que foram distorcidos por perdas na transmissão devido à atenuação. Um repetidor não realiza decisões inteligentes sobre o encaminhamento de pacotes como um roteador ou bridge. Um repetidor possui apenas duas portas.

Hubs
Concentram conexões. Em outras palavras, juntam um grupo de hosts e permitem que a rede os veja como uma única unidade. Isto é feito passivamente, sem qualquer outro efeito na transmissão dos dados.

Hubs Ativos
Possuem a mesma função de um Hub comum mas além disso eles também regeneram sinais. Pode se dizer que um Hub ativo é um conjunto de repetidores interligados, ou seja, é um repetidor multi-porta.

Bridges
Também conhecidas como pontes, convertem os formatos de dados transmitidos na rede assim como realizam gerenciamento básico de transmissão de dados. As bridges, como o próprio nome indica, proporcionam conexões entre redes locais. As bridges não só fazem conexões entre redes locais, como também verificam os dados para determinar se devem ou não cruzar a bridge. Isto faz com que cada parte da rede seja mais eficiente. As bridges possuem apenas duas portas.

Switches
São responsáveis por adicionar mais inteligência ao gerenciamento da transferência de dados. Eles não só podem determinar se os dados devem ou não permanecer em uma rede local, mas como também podem transferir os dados somente para a conexão que necessita daqueles dados. Pode-se dizer que um switch é uma bridge multi-porta.

Roteadores
O mais "inteligente" de todos. Possuem todas as capacidades listadas acima. Os roteadores podem regenerar sinais, concentrar conexões múltiplas, converter formatos dos dados transmitidos, e gerenciar as transferências de dados. Eles também podem ser conectados a uma WAN, que lhes permite conectar redes locais que estão separadas por longas distâncias. Nenhum outro dispositivo pode prover este tipo de conexão.

Classificação das Redes

As redes podem ser classificadas de acordo com sua topologia e/ou extensão e área de cobertura.

Classificação por extensão e área de cobertura
Uma das primeiras redes a serem criadas foi a rede locai (LAN - Local Área Network) ao se conectar vários computadores de uma sala.

À medida que o uso do computador cresceu, logo se percebeu que as LANs não eram o suficiente. Era necessário um modo de mover informações de maneira rápida e eficiente, não só dentro de uma sala/empresa, mas também de uma empresa para outra. A solução, então, foi a criação de redes de áreas metropolitanas (MANs) e de redes de longa distância (WANs). Como as WANs podiam conectar as redes usuárias dentro de grandes áreas geográficas, elas tornaram possível a comunicação entre empresas ao longo de grandes distâncias.

As redes podem ser classificadas por sua extensão de acordo com a tabela abaixo:

Distância entre processadores	Exemplo	Classificação
1 m	Metro quadrado	Rede pessoal (PAN)
10 m	Sala	Rede local (LAN)
100 m	Edifício	
1 km	Campus	
10 km	Cidade	Rede metropolitana (MAN)
100 km	País	Rede geograficamente distribuida (WAN)
> 1.000 km	Continente/Planeta	

Exemplos de cada rede:

- PAN (Personal Área Network) - Seu celular e o fone de ouvido bluetooth, é um exemplo de rede PAN;
- LAN (LAN Local Área Network) - Vários computadores de uma sala ou prédio;
- MAN (Metropolitan Área Network) - Duas filiais de uma empresa, em cidades diferentes, se comunicando;
- WAN (Wide Area Network) - Comunicação entre Países ou planetas, como por exemplo a comunicação entre a NASA e um fogete e/ou sonda.

Classificação por Topologia

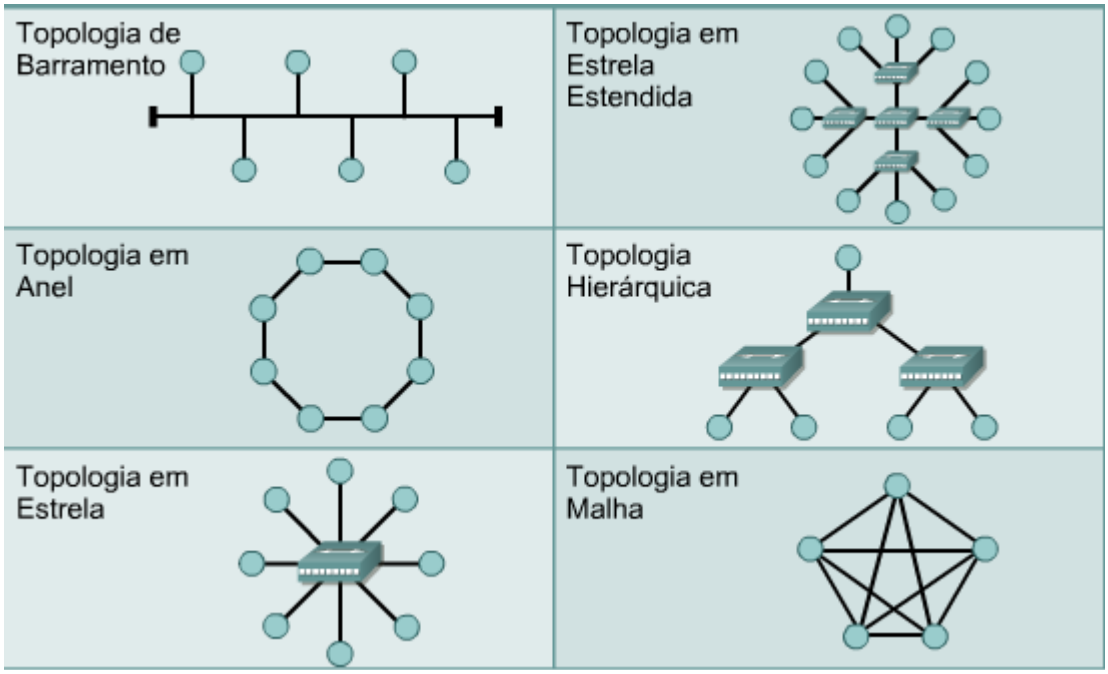
Topologias de rede definem a estrutura da rede. A definição de topologia pode ser dividida em duas:

- Topologia física - é a física da rede (fios e componentes como roteadores, concentradores, switches e clientes).
- Topologia lógica - define como os meios físicos são acessados pelos hosts para o envio de dados.

As topologias físicas que são comumente usadas são as seguintes:

- Uma topologia em barramento (bus) - usa um único cabo backbone que é terminado em ambas as extremidades. Todos os hosts são diretamente conectados a este barramento.
- Uma topologia em anel (ring) - conecta um host ao próximo e o último host ao primeiro. Isto cria um anel físico utilizando o cabo.
- Uma topologia em estrela (star) - conecta todos os cabos a um ponto central de concentração.
- Uma topologia em estrela estendida (extended star) - une estrelas individuais ao conectar os hubs ou switches.
- Uma topologia hierárquica - semelhante a uma estrela estendida porém, ao invés de unir os hubs ou switches, o sistema é vinculado a um computador que controla o tráfego na topologia.
- Uma topologia em malha (mesh) e/ou malha completa (full-mesh) - implementada para prover a maior proteção possível contra interrupções de serviço. Nessa topologia cada host tem suas próprias conexões com todos os outros hosts. Apesar da Internet ter vários caminhos para qualquer local, ela não adota a topologia em malha completa.

Abaixo um esquema das topologias físicas citadas acima



A topologia lógica de uma rede é a forma como os hosts se comunicam através dos meios. Os dois tipos mais comuns de topologias lógicas são broadcast (difusão) e passagem de token.

- Topologia de broadcast - Significa que cada host envia seus dados a todos os outros hosts conectados ao meio físico da rede. Não existe uma ordem que deve ser seguida pelas estações para usar a rede. A ordem é: primeiro a chegar, primeiro a usar.
- Topologia lógica - Utiliza a passagem de token. A passagem de token controla o acesso à rede passando um token eletrônico seqüencialmente para cada host. Quando um host recebe o token, significa que esse host pode enviar dados na rede. Se o host não tiver dados a serem enviados, ele passar o token para o próximo host e o processo será repetido.

Fontes:

- Cisco NetAcad
- Redes de Computadores, de Andrew S. Tanenbaum, 4.ed. Campus - 2003;

O que são protocolos

Protocolo é um acordo entre as partes que se comunicam, estabelecendo como se dará a comunicação. O protocolo é uma descrição formal de um conjunto de regras e convenções que governam a maneira de comunicação entre os dispositivos em uma rede. Os protocolos determinam o formato, temporização, seqüência, e controle de erros na comunicação de dados. Sem os protocolos, o computador não pode criar ou reconstruir o fluxo de bits recebido de outro computador no seu formato original. Estas regras para redes são criadas e mantidas por diferentes organizações e comitês. Incluídos nestes grupos estão:

- Institute of Electrical and Electronic Engineers (IEEE);
- American National Standards Institute (ANSI);
- Telecommunications Industry Association (TIA);
- Electronic Industries Alliance (EIA);
- International Telecommunications Union (ITU), anteriormente conhecida como Comité Consultatif International Téléphonique et Télégraphique (CCITT)
- FIPS – Federal Information Processing Standards
- MILSTD – Military Standards
- FCC – Federal Communications Comission
- IANA — Internet Assigned Number Authority
- IETF — Internet Engineering Task Force
- IRTF — Internet Research Task Force

- Internet Engineering Steering Group

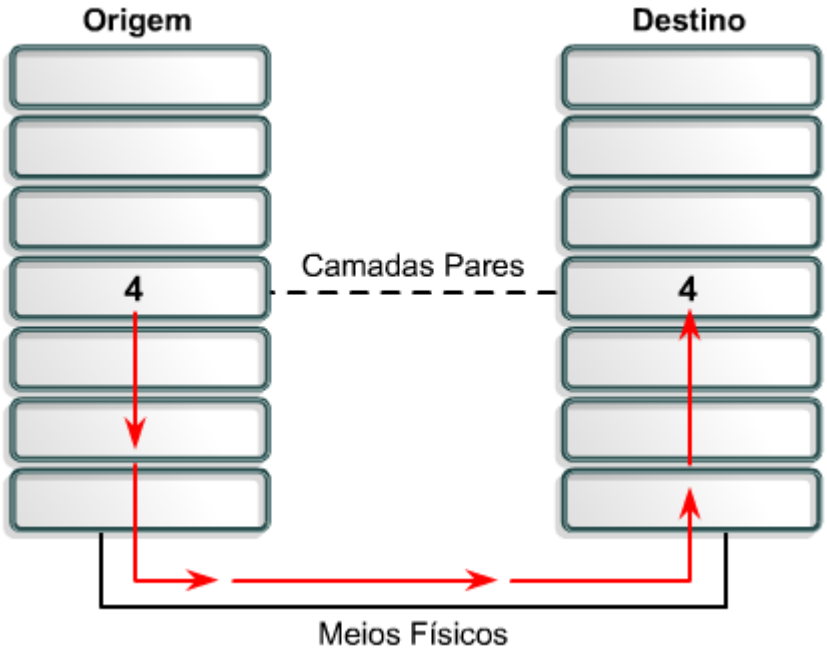
Os sistemas de comunicação de dados não usam apenas um único protocolo para tratar todas as tarefas de transmissão. Esse processo requer uma pilha de protocolos cooperativos, denominados algumas vezes de família de protocolos ou pilha de protocolos. Alguns exemplos de funcionalidades dos protocolos:

- **Indisponibilidade:** Causada por falha de um host ou roteador, seja por falha de hardware ou crise no sistema operacional, ou um enlace de transmissão pode falhar ou ser desconectado acidentalmente. Alguns protocolos detectam tais falhas e recuperar-se delas.
- **Congestionamento de redes:** Mesmo quando todo o hardware e software operam corretamente, as redes têm capacidade finita que pode ser ultrapassada. Os protocolos precisam encontrar formas para que uma máquina em congestionamento possa suprir o excesso de tráfego.
- **Demora ou perda de pacotes:** Algumas vezes, os pacotes demoram muito ou são perdidos. Os protocolos precisam aprender sobre as falhas ou adaptar-se a longas demoras.
- **Danificação de dados:** Interferência elétro-magnética ou falhas de hardware pode causar erros de transmissão que danificam o conteúdo dos dados transmitidos. Os protocolos necessitam detectar e recuperar tais erros.
- **Duplicação de dados ou erros seqüenciais:** Redes que oferecem rotas múltiplas podem transmitir dados fora de seqüência ou podem transmitir pacotes duplicados. Os protocolos necessitam reorganizar os pacotes e detectar/descartar pacotes duplicados.

Considerados em conjunto, esses problemas parecem enormes. É difícil entender como é possível preparar um único protocolo que poderá tratar todos eles. Aplica-se então a velha máxima de guerra: "Dividir para conquistar", onde cada camada assume a responsabilidade de tratar uma parte do problema.

As Camadas conceituais dos protocolos

Para que os pacotes de dados trafeguem de uma origem até um destino, através de uma rede, é importante que todos os dispositivos da rede usem a mesma linguagem, ou protocolo. Podemos imaginar todos os protocolos empilhados verticalmente em camadas como na figura abaixo:



No exemplo acima a Camada 4 da origem se comunica com a Camada 4 no computador de destino. É muito importante frisar que a comunicação é realizada entre camadas pares, isto é, a camada 4 de origem se comunica com a camada 4 de destino, a camada 3 de origem se comunica com a camada 3 de destino, a camada... Nunca ocorrerá comunicação

entre camadas de níveis diferentes. Isto se deve ao fato das regras e convenções usadas para uma camada serem desconhecidas para camadas diferentes. Seria algo similar como colocar um brasileiro (que só sabe falar português) pra falar com um japonês (que só sabe falar japonês).

Quando um dado é enviado da origem para o destino, o dado é tratado por todas as camadas, de cima para baixo, até chegar à camada mais baixa e ser transmitido, por isso costumamos dizer que a camada N prove serviço para a camada N+1. Quando o pacote chega ao destino os protocolos desfazem a construção do pacote que foi feito no lado da fonte. Isto é feito na ordem inversa, de baixo para cima.

Na prática, o protocolo é muito mais complexo do que o modelo mostrado. Cada camada toma decisões em relação à correção da mensagem e escolhe uma ação apropriada baseada no tipo de mensagem ou no endereço de destino.

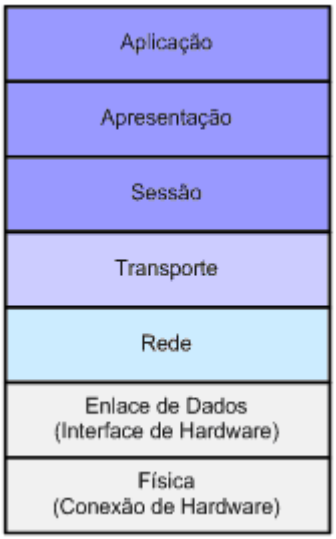
Modelo ISO/OSI

Voltando para o lado histórico das redes. Nos meados de 1980, as empresas começaram a sentir os problemas causados pela rápida expansão. Assim como pessoas que não falam o mesmo idioma têm dificuldade na comunicação entre si, era difícil para as redes que usavam diferentes especificações e implementações trocarem informações. O mesmo problema ocorreu com as empresas que desenvolveram tecnologias de rede proprietária ou particular. Astecnologias de rede que seguiam estritamente as regras proprietárias não podiam comunicar-se com tecnologias que seguiam diferentes regras proprietárias.

Para tratar dos problemas de incompatibilidade entre as redes, a ISO realizou uma pesquisa nos modelos de redes como Digital Equipment Corporation net (DECnet), Systems Network Architecture (SNA) e TCP/IP a fim de encontrar um conjunto de regras aplicáveis a todas as redes. Com o resultado desta pesquisa, a ISO criou um modelo de rede que ajuda os fabricantes na criação de redes que são compatíveis com outras redes.

O modelo de referência da Open System Interconnection (OSI) lançado em 1984 foi o modelo descritivo de rede que foi criado pela ISSO, por isso é chamado de ISSO/OSI. Ele proporcionou aos fabricantes um conjunto de padrões que garantiam uma maior compatibilidade e interoperabilidade entre as várias tecnologias de rede produzidas pelas companhias ao redor do mundo.

O modelo ISO contém sete camadas conceituais organizadas como mostra a figura abaixo:



Abaixo um resumo básico sobre as 7 camadas do modelo ISSO/OSI:

Camada Física. Especifica a interconexão física, incluindo as características elétricas de voltagem e corrente. Dizemos que na camada física é responsável pelos fios, conectores, voltagens, taxa de dados, hubs e tranceivers.
Camada de enlace de dados. O protocolo de nível dois define o formato dos quadros e especifica como as duas máquinas reconhecem os limites do quadro e implementa um primeiro nível de detecção de erro. Dizemos que na

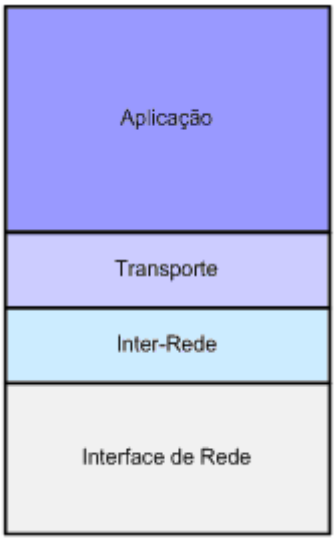
camada de enlace é responsável pelo controle de acesso ao meio, placas de redes, bridges, switches e endereçamento MAC.
Camada de rede. Contém a funcionalidade que completa a definição da interação entre o host e a rede. Esse nível define: a unidade básica de transferência na rede; endereçamento lógico; escolha do melhor caminho; entrega por melhor esforço; transferência de dados confiável através do meio. Dessa forma a camada de rede é responsável por Endereços IPs, endereçamentos IPXs, roteamento, protocolos de roteamento e pelos roteadores.
Camada de transporte. Oferece confiabilidade ao fazer com que o host de destino se comunique com o host central. É a camada que provê comunicação fim-a-fim com controle de erro e retransmissão. Também é responsabilidade dessa camada o controle de fluxo e de congestionamento.
Camada de sessão. Permite que os usuários de diferentes máquinas estabeleçam sessões entre eles, oferecendo serviços de controle de diálogo, entre outros.
Camada de apresentação. É voltada a incluir funções de que muitos programas aplicativos precisam ao usar a rede. Torna possível a comunicação entre computadores com diferentes representações de dados pois ela define a formatação, compressão e construção das estruturas de dados.
Camada de aplicação. São os próprios programas/aplicativos que usam a rede, comumente necessários para os usuários. Exemplos incluem o correio eletrônico e o programa de transferência de arquivos e navegadores.

Modelo TCP/IP

O padrão histórico e técnico da Internet é o modelo TCP/IP. O Departamento de Defesa dos Estados Unidos (DoD) desenvolveu o modelo de referência TCP/IP porque queria uma rede que pudesse sobreviver a qualquer condição, mesmo a uma guerra nuclear. Em um mundo conectado por diferentes tipos de meios de comunicação como fios de cobre, microondas, fibras ópticas e links de satélite, o DoD queria a transmissão de pacotes a qualquer hora e em qualquer condição. Este problema de projeto extremamente difícil originou a criação do modelo TCP/IP.

Ao contrário das tecnologias de rede proprietárias mencionadas anteriormente, o TCP/IP foi projetado como um padrão aberto. Isto queria dizer que qualquer pessoa tinha a liberdade de usar o TCP/IP. Isto ajudou muito no rápido desenvolvimento do TCP/IP como padrão.

O modelo TCP/IP tem as seguintes quatro camadas:

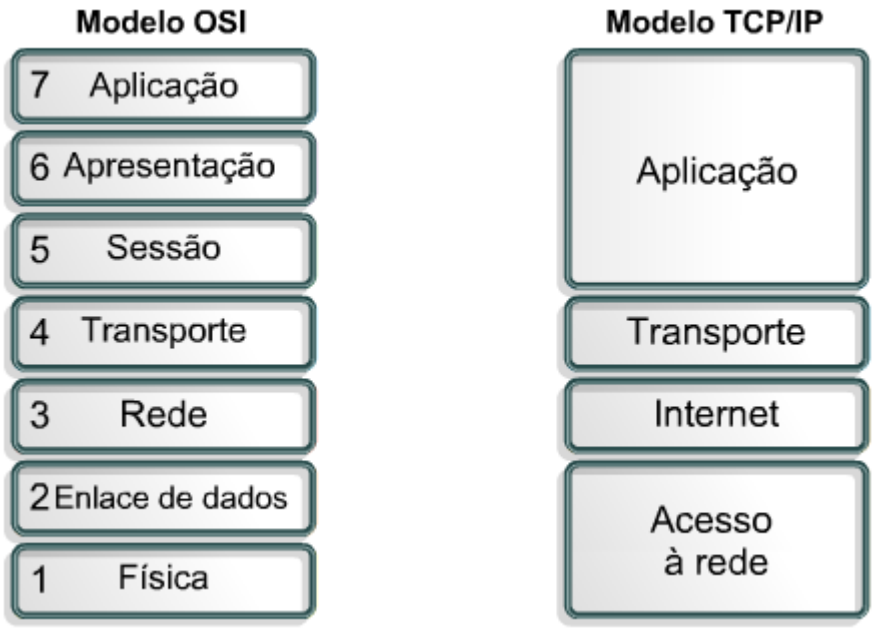


Os projetistas do TCP/IP decidiram que os protocolos de mais alto nível deviam incluir os detalhes da camada de sessão e de apresentação do OSI. Eles simplesmente criaram uma camada de aplicação que trata de questões de representação, codificação e controle de diálogo.

Como disse, nos próximos posts vou abordar cada camada tratando seus protocolos, PDUs, fomatos de encapsulamento e detalhar seu funcionamento! Então se preparem, pois o difícil está por vir.

Chegando a 3 parte desse curso veremos a camada física do modelo OSI. Tentarei cobrir uma camada a cada post, não sei se vou conseguir Mas antes vamos rever um pouco as camadas OSI e TCP/IP.

Eu gosto de dizer que o modelo de 7 camadas OSI foi enxuto e resumido em 5 camadas no TCP/IP. Cada camada do OSI tem seu "relativo" no TCI/IP. Digo relativo pois não são idênticos, cada um utiliza uma gama de protocolos diferentes, porém todos tem a mesma finalidade. O modelo OSI pode ser comparada ao modelo TCP/IP da seguinte forma:



Eu gosto muito da "resumida" do TCP/IP. Só não sou 100% a favor pois não consigo tratar a camada Física e de enlaçe como sendo uma única coisa. Agora as camadas Sessão, apresentação e aplicação foram agrupadas perfeitamente na camada de aplicação. Realmente, controle de sessão, formatação de dados e apresentação dos dados para o usuário é função da aplicação e não da rede! Até mesmo porque nunca vingou um protocolo de "sessão"! Essa é a camada mais absurda do OSI...

Deixando de conversa vamos lá...

Camada Física

A camada física se refere aos meios de transmissão. Atualmente podemos dividir os meios de transmissão em três tipos:

- Meios de Cobre;
- Meios Óptiocos;
- Meios sem Fio.

Os meios de cobre são usados em quase todas as redes locais. Estão disponíveis vários diferentes tipos de cabos de cobre, cada tipo tem suas vantagens e desvantagens. Uma seleção cuidadosa de cabeamento é a chave para uma operação eficiente de redes.
Os meios ópticos são freqüentemente usado para as transmissões ponto-a-ponto a grandes distâncias e com alta largura de banda necessárias para backbones das redes locais e em WANs.
A tecnologia sem fio oferece uma portabilidade verdadeira ao mundo da computação.

Especificações do cabeamento

Cada cabo possui uma especificações diferente e de acordo com a especificação, possui um desempenho diferente.

As especificações levam em conta as seguintes características:
- Velocidades para transmissão de dados
- Tipo de transmissão: A transmissão digital, ou de banda base (BASE), e a transmissão analógica ou de banda larga (BROAD).
- Distância que um sinal percorre até ser atenuado de forma a não ser reconhecido

Alguns exemplos de especificações Ehternet:
- 100BASE-TX
- 10BASE5
- 10BASE2

Vamos tomar como exemplo a notação 100BASE-TX:
- O primeiro número, 100, indica a velocidade de transmissão, 100Mbps.
- O BASE indica que ele deve ser utilizado para transmissão em banda base (sinal digital). Para sinais analógicos seria utilizado a notação BROAD, banda larga.
- A última letra/número indica o tipo de cabo e a distância. TX indica um cabo par trançado que transmite em até 100 metros.

Outros valores que podemos encontrar serão detalhados abaixo.
- 10BASE-T - Velocidade de transmissão a 10 Mbps, tipo de transmissão é banda base e utiliza par trançado.
- 10BASE5 - Velocidade de transmissão a 10 Mbps, tipo de transmissão é banda base e o 5 representa o limite de transmissão de aproximadamente 500 metros. O 10BASE5 é geralmente conhecida como Thicknet. O cabeamento Thicknet era utilizado antigamente para backbone e ligação entre prédios.
- 10BASE2 - Velocidade de transmissão a 10 Mbps, tipo de transmissão é banda base e o 2 representa o limite de transmissão de aproximadamente 200 metros. A 10BASE2 é geralmente conhecida como Thinnet.

Cabo Coaxial
O cabo coaxial consiste em um condutor de cobre envolto por uma camada isolante flexível. O condutor central também pode ser feito de um fino cabo de alumínio laminado, permitindo que o cabo seja industrializado a baixo custo. Sobre o material isolante, há uma trança de lã de cobre ou uma folha metálica, que age como um segundo fio no circuito e como blindagem para o fio interior. Esta segunda camada, ou blindagem, também reduz a quantidade de interferência eletromagnética externa. A capa reveste esta blindagem. Geralmente classificado em thinnet e thicknet devido a sua espessura.

Par trançado
O cabo par trançado é composto por 4 pares (8 fios). Os cabos Par trançados são categorizados de 1 a 7. Essa categoria vem escrita na capa do cabo utilizando abreviações de "Category 5" (Cat. 5). Essas categorias são relacionadas à capacidade do meio. Baseado na atenuação, ruído e perda os cabos são direcionados para uma certa aplicação conforme abaixo:
- Cat 1 - Serviços telefônicos e dados de baixa velocidade
- Cat 2 - RDSI e circuitos T1/E1 - 1,536 Mbps/2,048 Mbps
- Cat 3 - Dados até 16 MHz, incluindo 10Base-T e 100Base-T
- Cat 4 - Dados até 20 MHz, incluindo Token-Ring e 100B-T (extinto)
- Cat 5 - Dados até 100 MHz, incluindo 100Base-T4 e 100Base-TX (extinto)
- Cat 5e - Dados até 100 MHz, incluindo 1000Base-T e 1000Base-TX
- Cat 6 - Dados até 200/250 MHz, incluindo 1000Base-T e 1000Base-TX
- Cat 7 - Dados até 500/600 MHz

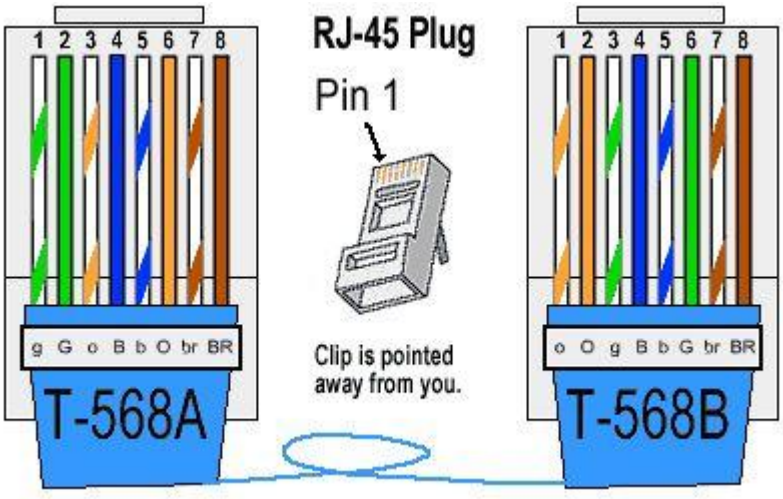
Tipos de Cabos
As redes gigabit ethernet devem utilizar cabos de categoria maior ou igual a 5e, para que a rede tenha um bom desempenho.
Aqui vamos nos dedicar mais aos cabos Cat. 5 devido a padronização e produção dos cabos acima do Cat. 6 não estarem completamente definidos. Eles geralmente são separados entre UTP (Unshilded Twisted Pair), STP (Shielded Twisted Pair) e ScTP (Screened Twisted Pair).

UTP - É o mais usado atualmente tanto em redes domésticas quanto em grandes redes industriais devido ao fácil manuseio, instalação, permitindo taxas de transmissão de até 100 Mbps. É o mais barato para distâncias de até 100 metros. Sua estrutura é de quatro pares de fios entrelaçados e revestidos por uma capa de PVC. Pela falta de blindagem este tipo de cabo não pode ser instalado próximo a equipamentos que possam gerar campos magnéticos (fios de rede elétrica, motores) e também não podem ficar em ambientes com umidade.

ScTP - Também referenciado como FTP (Foil Twisted Pair), os cabos são cobertos pelo mesmo composto do UTP, no entanto todos os pares são revestidos por uma capa metálica (Foil) enrolada sobre todos os pares trançados, o que contribui para um maior controle de EMI (Interferência Eletromagnética), embora exija maiores cuidados quanto ao aterramento.

STP - Ou Par Trançado Blindado (cabo com blindagem). É semelhante ao ScTP. A diferença é que além da blindagem externa de todos os pares ele possui uma malha de blindagem sobre cada par. É usado em ambientes com interferência eletromagnética. Por causa de sua blindagem possui um custo mais elevado.

Os cabos pares trançados utilizam conectores RJ-45 para sua terminação. Pra cabos com blindagem é necessária a utilização de conectores RJ-45 especiais que possuem uma parte metálica. Isso se deve à necessidade de manter um "terra único" para todo o cabeamento estruturado. Dessa forma não só o RJ-45 deve ser diferenciado mas também o jack, os keystones e patch pannels que venham a se conectar a esse cabeamento blindado.
A pinagem do cabo par trançado também é muito importante. Existem 2 padrões para a pinagem TIA/EIA-568-A e TIA/EIA-568-B. Esses padrões podem ser vistos ns imagem a seguir:



Quando utilizamos uma cabo com o mesmo padrão em ambas as extremidade dizemos que ele é um cabo **pino-a-pino**, ou **direto**. Se utilizarmos uma extremidade com um padrão e a outra extremidade com outro padrão teremos um cabo **cross-over**. Existe uma "regrinha" para saber quando utilizar os cabos diretos e os cabos cross:

- Os cabos diretos geralmente são utilizados para realizar conexões entre dispositivos de categorias diferentes como: PC-swich, PC-Modem e Hub-switch.
- Os cabos cross-over geralmente são utilizados para interligar equipamentos da mesma categoria como: switch-switch, roteador-roteador, Modem-modem e PC-PC.

Existem alguns casos em que essa "regra" não se aplica. Por exemplo, ao interligarmos um PC e um roteador é utilizado um cabo cross. O mesmo ocorre ao interligar um HUB e um Switch. A regra correta é seguir a classificação dos dispositivos (DTEs e DCEs) ou pensar no "cruzamento" de pares internos em cada equipamento.

Essa classificação é devido a categorização dos equipamentos como DTE (data termination equipment) e DCE (data communication equipment) que seriam equipamentos que "geram/recebem a comunicação" (DTE) e equipamentos que provêem essa comunicação (DCE). Atualmente muitos switches e roteadores possuem a capacidade de detectar o tipo de cabeamento utilizado e se auto-ajustar.

Existe também o cabo **roll-over** utilizado para realizar a conexão entre a porta serial do PC e a porta de console de roteadores. Essa conexão pode ser feita através de um conversor (tranceiver) RJ-45 para DB-9.

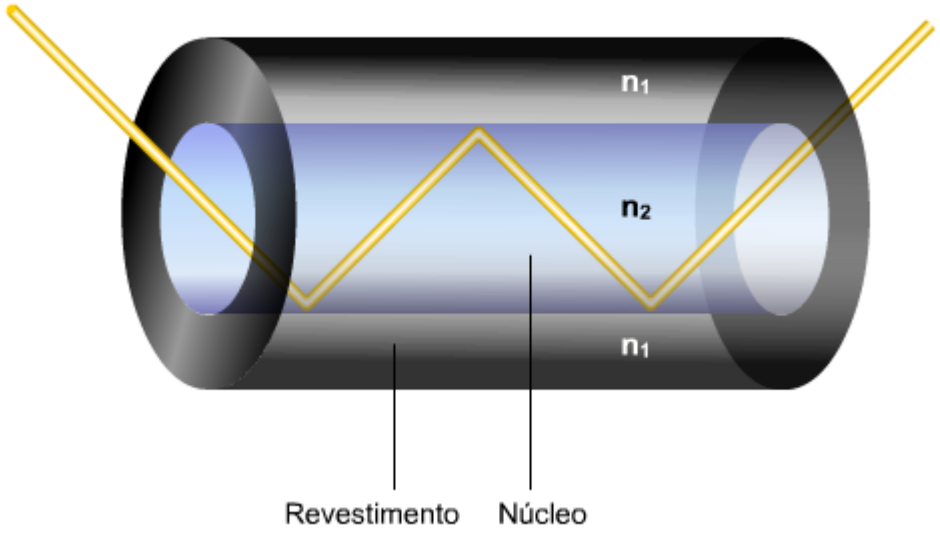
Se observarmos um cabo cross-over simplesmente possui os pinos 1-3 e 2-6. Com isso muitas pessoas dizem que basta apenas realizar esse cruzamento. E pode desprezar o restante dos cabos. Essa afirmação é apenas para redes que chegam a apenas 100Mbps. Redes de 1GBps utilizam os 4 pares para realizar a transmissão, dessa forma todos os cabos importam. Outra afirmação que costumamos ouvir é que não importa a a ordem porque "a cor não vai influir". Realmente a cor não influi, mas os cabos pares trançados forma projetados com uma certa trançagem que provê cancelamento entre os pares. Então ao mudar a ordem dos pinos você perde "a qualidade" do cabo.

Abaixo uma lista da maioria dos padrões ethernet sobre cobre:
Xerox Ethernet – Primeira implementação ethernet;

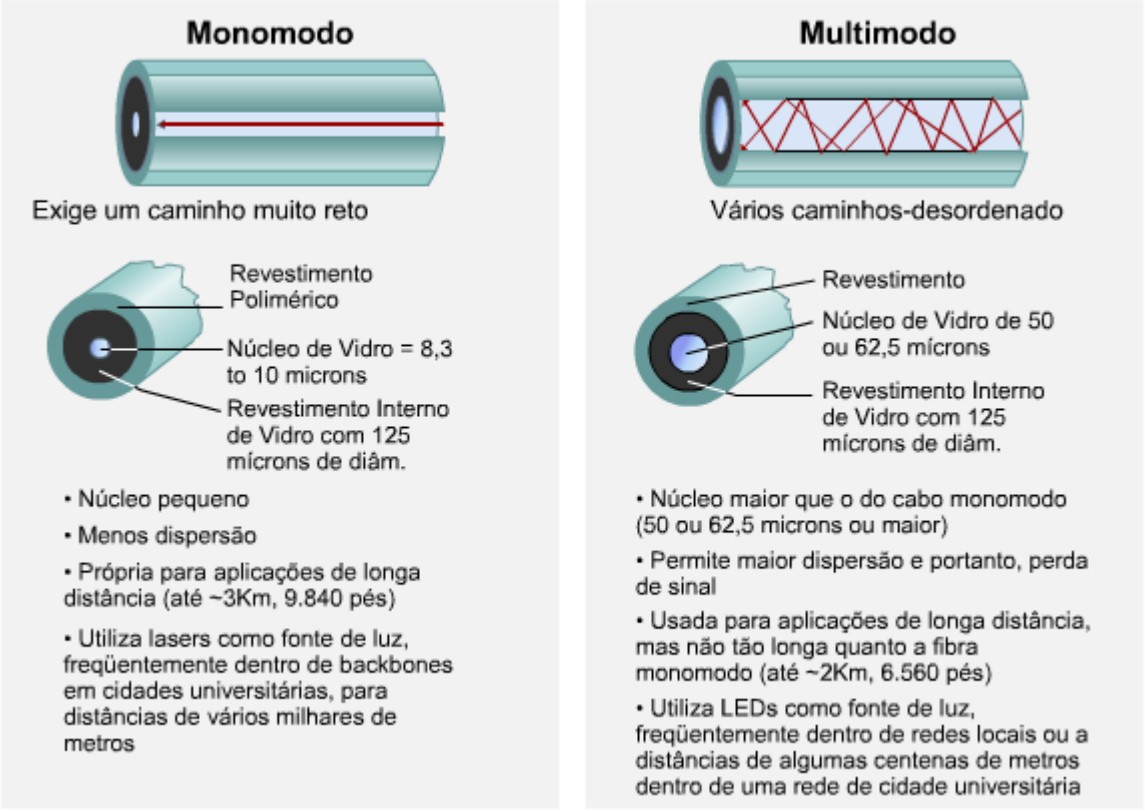
- 10BASE5 – Padrão Thinnet;
- 10BROAD36 – Antigamente utilizado para comunicação entre distâncias mais longas entre modem;
- 1BASE5 – Primeira tentativa de padronizar uma rede local de baixo custo;
- StartLan 1 – Primeira implementação de ethernet sobre par trançado;
- 10BASE2 – Padrão Thinnet ou Cheapnet;
- StarLan 10 – Primeira implementação de 10Mbps sobre par trançado. Precursor do padrão 10BASTE-T;
- 10BASE-T – Evolução do StarLan 10;
- 100BASE-TX – Padrão mais utilizado atualmente, utiliza cabos par trançado cat. 5;
- 100BASE-T4 – Implementação da comunicação a 100Mbps sobre cabos Cat. 3 utilizando todos os 4 pares;
- 100BASE-T2 – Implementação da comunicação a 100Mbps sobre cabos Cat. 3 utilizando todos os 2 pares;
- 1000BASE-T – Gigabit Ethernet sobre par trançado Cat. 5e ou 6;
- 1000Base-CX – Anterior ao 1000BASE-T que possibilitou a comunicação Gigabit sobre cobre (anteriormente só implementada sobre fibra ótica) e utilizava cabos twiaxiais porém cobria pequenas distâncias (25m). Cabos twiaxiais tem a aparência ed dois cabos coaxiais "colados".

Fibras Óticas

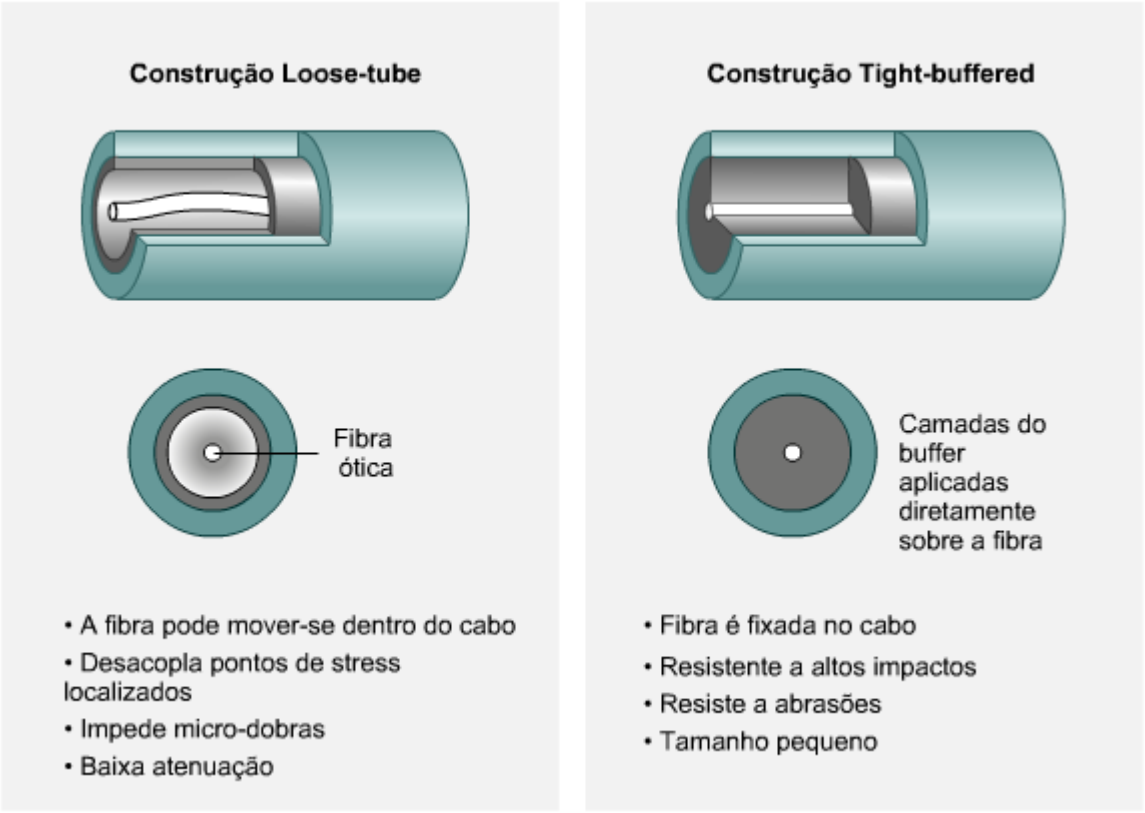
As fibras óticas utilizam a luz e o princípio da reflexão para realizar a transmissão. Como a fibra ótica não utiliza impulsos elétricos para realizar a transmissão ela não é susceptível a interferências eletromagnéticas porém a luz ainda sofre atenuação devido ao princípio da refração (não confundir com reflexão).
A fibra ótica é composta por um núcleo de vidro revestido por um encapamento. A luz é inserida (por lasers ou LEDs) no vidro e "viaja" através do núcleo de vidro através da reflexão nas "paredes" entre o vidro e o revestimento. Porém essa "parede" é perfeita e a luz é refratada perdendo potência.



Quanto menor o diâmetro do núcleo menor é a refração da luz e maior é o alcance da fibra. Com isso criou-se a classificação de fibras monomodo e multimodo. As fibras monomodo suportam apenas um modo devido ao seu pequeno diâmetro, variando de 8 a 10 microns (ou micrômetros), enquanto as fibras multímodo variam de 50 a 62,5 microns.



As fibras são compostas por um núcleo, revestimento interno, buffer (caso exista), Cordão de fibra (aramid) e a capa. O buffer é utilizado para suavizar as curvas e evitar que o núcleo se rompa. Existem dois padrões de construção, o tight-buffered e o loose-tube. No tight-bufeered o buffer é quase inexistente enquanto no loose-tube a parte "interna" da fibra fica "mergulhada" em um gel.



Em fibras monomodo são utilizados Lasers devido a necessidade de precisão enquanto na fibra multimodo pode ser utilizado lasers ou LEDs, sendo que lasers possibilitam que o sinal viaje por distancias maiores. As fibras podem ter também dois tipos de terminação ST e LC. Os conectores ST são geralmente utilizados para fibras monomodo enquanto os conectores SC são mais voltados para fibras multímodo. Com isso vemos que podemos ter muitas variações em fibras. De acordo com o tamanho do núcleo e o tipo de transmissor utilizado o alcance das fibras podem variar.

Abaixo alguns padrões de fibra:

- 100BASE-FX – 100 Mbit/s ethernet sobre fibra óptica. Usando fibra ótica multimodo 62,5 mícrons tem o limite de 400 metros;
- 1000BASE-SX – 1 Gbit/s sobre fibra multímodo e LEDs podendo atingindo até 550 metros;
- 1000BASE-LX – Utilizado fibras monomodo e lasers pode-se atingir 5Km.
- 10GBASE-SR – Projetado para suportar distâncias curtas sobre cabeamento de fibra multi-modo, variando de 26m a 82m dependendo do tipo de cabo. Suporta também operação a 300m numa fibra multi-modo;
- 10GBASE-LX4 – Usa multiplexação por divisão de comprimento de ondas (DWDM - Dense Wavelength Division Multiplexing) para suportar distâncias entre 240m e 300m em cabeamento multi-modo. Também suporta 10km com fibra mono-modo;
- 10GBASE-LR e 10GBASE-ER – Esses padrões suportam 10km e 40km respectivamente sobre fibra mono-modo;
- 10GBASE-SW, 10GBASE-LW e 10GBASE-EW. Essas variedades usam o WAN PHY, projetado para interoperar com equipamentos OC-192 / STM-64 SONET/SDH. Eles correspondem à camada física do 10GBASE-SR, 10GBASE-LR e 10GBASE-ER respectivamente, e daí usam os mesmos tipos de fibra e suportam as mesmas ditâncias.

A camada física especifica também as características utilziadas para transmitir em cada tecnologia. Essas características englobam voltagem, corrente, frequência... Preferi dar mais foco nos meios de transmissão do que em carcterísticas de engenharia e mesmo assim achei que ficou grande e tive que cortar açguns conteúdos como problemas de ruidos em cabos par trançado (FEXT, NEXT e etc), problemas de fibras (microbends, macrobend, acoplamento e etc) e redes sem fio. Ainda não sei, dependendo das fontes que eu achar sobre isso, posso fazer um post complementando esse...

Camada 2 - Enlace de Dados (Parte 2)

Um pequeno review

No último post falamos do quadro (Frame) e seus campos, endereçamento MAC e controle de acesso ao meio (CSMA/CD). Agora nos vamos ver a utilidade desses conceitos.

Primeiramente vamos fazer uma abstração para compreender melhor o que seria um quadro, pra que precisamos desses campos e porque ter um controle de acesso ao meio.

Vamos imaginar que os **computadores são pessoas** e a **rede é uma sala**. Nessa sala todas as pessoas podem **se comunicar apenas falando**. Mas cada fala tem uma forma, uma língua **(protocolo)**. Essa língua é definida por palavras **(padrões)**. Dessa forma, se colocarmos um japonês nessa sala, só com brasileiros (considerando que nenhum assiste anime ou fez intercâmbio no Japão) o japonês não conseguirá se comunicar com ninguém, pois ele**não "fala o mesmo protocolo"**. Inclusive esse "fala" é um jargão muito comum em redes: "esse roteador fala OSPF??". Voltando ao escopo. Todos nessa sala podem se comunicar através da fala, e como sabemos que a fala são **ondas sonoras que se propagam pelo ar**. Se comparado a redes, o ar é o meio de transmissão (cabo ethernet) e as ondas sonoras são os pulsos eletro-magnéticos gerados pelas interfaces ethernet. As ondas sonoras chegam até nossos ouvidos **(interface)** que, através da pressão sofrida pelas ondas gera uma oscilação dos tímpanos, "decodifica" essas ondas. Nossos ouvidos funcionam como interfaces ethernet que decodificam os impulsos eletromagnéticos do meio em sinais binários (uns e zeros).

Agora, o que acontece se todos falarem ao mesmo tempo? Ninguém se entende certo?! Por isso temos o **controle de acesso ao meio**. Quando alguém fala, todos os outros se calam. Como isso é feito? Antes de falar você tem que**ouvir** pra saber se alguém está falando (conforme definido pelo algoritmo CSMA/CD) e em caso positivo **espera a pessoa terminar**. Vamos considerar que a pessoa que quer falar é o ***Irado*** (para quem não conhece aqui tá o perfil dele: Irado). Ele primeiro ouve o meio e se **estiver em silêncio**, ele poderá falar. Mas **ao mesmo** tempo em que ele**fala,** ele **ouve,** pois alguém pode começar a falar ao mesmo tempo que ele, ocorrendo assim uma **colisão**. O que acontece se alguém fala ao mesmo tempo que o Irado? Ele age como um cavalheiro e avisa seus companheiros com um sonoro: "Eu to falando p***a!!!". Esse sinal de aviso seria o **sinal de JAM** que tem como objetivo evitar que as pessoas continuem a falar. Enquanto isso, a pessoa que falou ao mesmo tempo que o Irado estará se recuperando do susto (**tempo aleatório** na ponta A) enquanto o Irado conta até 10 (**tempo aleatório** na ponta B) pra se acalmar e não partir a cara desse usuário que teima em não usar o google. Depois de contar até 10 o Irado tenta falar novamente, dessa vez (com razão) ninguém tenta falar ao mesmo tempo que ele!

Obs: Irado isso é só uma brincadeira OK?!

Outro **padrão** na nossa fala é sempre se dirigir a pessoa com quem estamos falando: "Fulano, você já atualizou seu kernel hoje?". Dessa forma **endereçamos** à pessoa (fulano=Endereço MAC) a mensagem desejada. Geralmente, depois de falar o nome da pessoa esperamos **alguns segundos** antes de falar o restante da frase. Isso é para dar tempo da pessoa direcionar sua **atenção** à nossa pergunta. Se não esperarmos, muito provavelmente ouviremos um**"hum?!"** ou **"oq?!"**. Isso é porque não esperamos a pessoa se "**sincronizar**" à nossa conversa, função dos campos**Preâmbulo** e **SFD** do quadro. Por sorte quando conversamos com alguém não precisamos informar o tamanho da frase que falamos e nem que somos nós que estamos falando. Mas essas funções são substituídas pela visão da pessoa, reconhecendo a **origem** e vendo que não estamos mais falando, indicando que terminamos de transmitir (o campo comprimento tem o objetivo de prever o fim do quadro). Os **dados** são nossa pergunta, "você já atualizou seu kernel hoje". O **FCS** tem como objetivo garantir que o que a pessoa ouviu foi o que enviamos. Esse função é exercida por aquela perguntinha chata que muitas vezes ouvimos como resposta de nossas perguntas: "você me perguntou se eu já atualizei meu kernel hoje?". Como em redes podemos conversar com uma única pessoa, com todas as pessoas em um **segmento de rede** ou com um grupo definido. "Segmento de rede?" calma, veremos isso hoje!

Ao falar com uma única pessoa, temos uma comunicação **unicast**: "fulano, você já atualizou seu kernel hoje?". Quando falamos com um grupo de pessoas temos um **multicast**: "fulano e sicrano, vocês já atualizaram o **kernel** hoje?". Quando falamos com todos ao mesmo tempo temos um broadcast: "Galera, vocês já atualizaram o kernel hoje". Sempre que falamos "galera" todo mundo sabe que todos devem ouvir. Esse é o endereços de **broadcast**.

Outra abstração muito utilziada para ensinar é umaginar as redes como os correios. Você antes de mandar uma carta, escreve o conteúdo em um papel põe no envelope e envia a carta. O envelope seria o cabeçalho/trailer.

Com isso revisamos tudo o que vimos anteriormente, só que de uma forma subjetiva. Agora vamos para conceitos novos!

Segmentos de Rede

Segmentos de rede são trechos de uma rede. Podemos ter Segmentos em camada 2 ou segmentos em camada 3. Isso vai depender de com que equipamento é feita a segmentação.
Vamos ver a seguinte imagem:

Essa imagem tem 3 partes. Primeiro temos duas redes totalmentes segmentadas, costumamos dizer que são redes**segregadas**. Chamamos a rede da esquerda de Segmento A e a de direita de Segmento B. Como podemos ligar esses dois segmentos de rede? De diversas formas. Nas imagens apresentei 2 formas: com um HUB ou Repetidor e com uma Bridge ou Switche.

Com um HUB: Ao utilziar o HUB nós **estendemos** os segmentos de rede tornando ambos os segmentos (A e B) em um mesmo segmento. Isso é devido ao fato que um HUB é BURRO! Não, eu não to brincando não! Costumamos dizer que um hub é burro porque ele não tem "inteligência", ou seja, ele não toma decisões. Por isso ele une os dois segmentos. Um HUB funciona como um barramento, qualquer dado que você enviar nele todos (que esteja conectados nele) podem "ouvir" o tráfego e também ele permite a ocorrência de colisões.

Com um Switch: Ao utilizar o switch nós não unimos a rede, nos a interligamos. Teremos dois segmentos de reme na camada 2. Isso será melhor compreendido quando vermos o funcionamento de uma bridge/switch. O que interessa é que, em condições ótimas, os hosts do segmento A não escutarão as conversas do segmento B e não haverá

colisões**entre os segmentos**. Muita atenção aqui. A rede não está livre de colisões no geral porque temos os HUS em cada segmento, mas **ENTRE** os segmentos pode-se dizer que não haverá colisões.

Sempre que expandimos uma rede podemos aumentar, manter ou diminuir o tamanho do segmento, tudo depende de que ativos utilizamos. Conforme vão sendo adicionados nós a um segmento físico Ethernet, vai aumentando a competição pelos meios. E quantos mais nós adicionamos mais perdemos em termos de **throughput**. Throughput é a velocidade real de uma rede. Pode-se dizer que uma rede de 100Mbps com 4 hosts tem um throughput pouco inferior a 25Mbps, considerando que todos falem ao mesmo tempo. Isso é porque os 100Mbps serão divididos para os 4 hosts. "E porque um pouco inferior?!". Porque, como vimos, quando enviamos dados de um host para outro não enviamos apenas os dados enviamos cabeçalhos e dados de controle, isso se chama **overhead**. O overhead varia de acordo com a pilha de protocolos utilizados. Agora vamos às bridges...

Bridge

A Ethernet compartilhada funciona extremamente bem sob condições ideais. Quando o número de dispositivos que tentam acessar a rede é baixo, o número de colisões permanece bem dentro dos limites aceitáveis. No entanto, quando aumenta o número de usuários na rede, o aumento do número de colisões pode causar um desempenho inaceitavelmente baixo. O uso de bridges foi elaborado para ajudar a amenizar os problemas de desempenho que surgiram devido ao aumento das colisões. Uma das funções da bridge é a **segmentação do domínio de colisão**. Pode-se dizer que o domínio de colisão é a mesma coisa que segmento de camada 2.

A bridge é um dispositivo de camada dois que possui duas interfaces suas principais funções são a segmentação de tráfego em camada 2, isto é, a bridge encaminha ou descarta os quadros baseados nas entradas da tabela MAC.

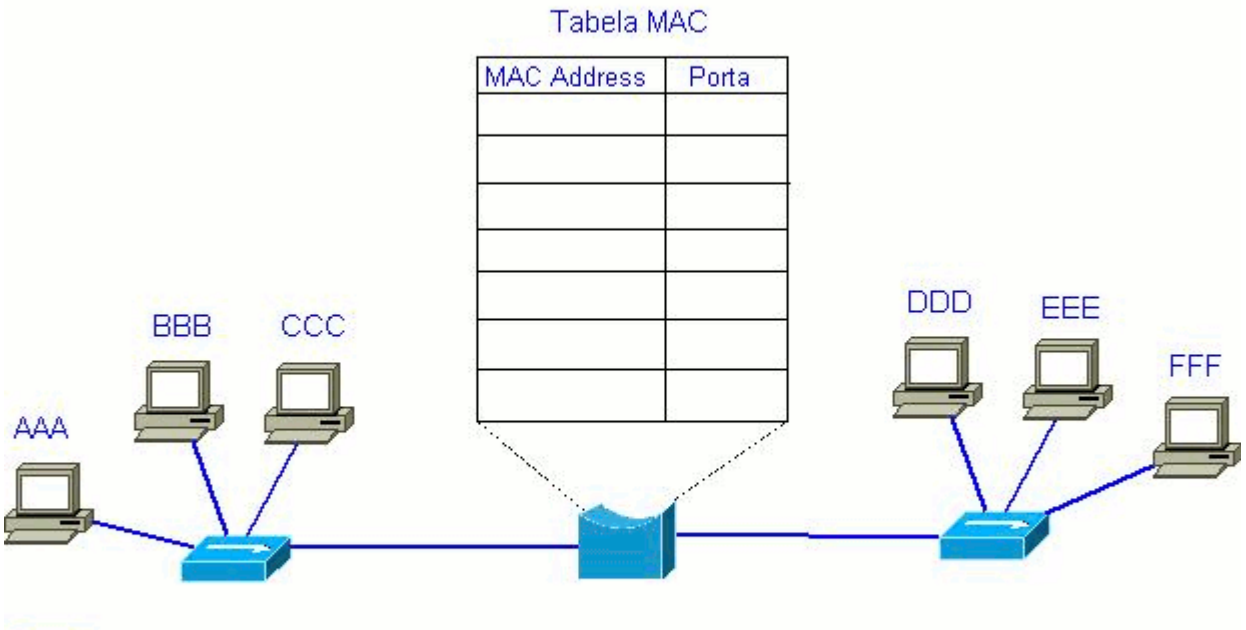
O que é segmentar. Segmentar é separar. Então ao segmentar uma rede, nos a separamos. Separamos com base no que?! Como disse segmentação de camada 2. Logo separamos com base na camada 2. Ou seja, com base no endereço MAC. Mas se você se recorda eu havia dito que o endereço MAC é único para todas as interfaces e ela possui um início baseado no fabricante. Com o endereçamento MAC não conseguimos criar grupos, sub-grupos ou redes, por isso dizemos que o MAC é um **endereçamento linear**. Então o único jeito é criar uma tabela de endereços MAC dinamicamente na memória da bridge. Todas as decisões feitas por uma bridge são baseadas no na tabela MAC e no endereço MAC de destino do pacote e não afetam o endereçamento lógico ou da Camada rede (apenas guarde isso, será útil futuramente). Assim, uma bridge divide um domínio de colisão, mas não tem efeito nenhum no **domínio de broadcast**. "Opa, domínio de colisão e domínio de broadcast?!?!". Ok... Definição formal:

Domínio de colisão: É um segmento de rede (cabos e ativos) em que um pacote pode trafegar e colidir.
Domínio de broadcast: É um segmento de rede (cabos e ativos) em que um host pode se comunicar com outro sem a necessidade de um dispositivo de camada 3.

Bridging

Para entendemos melhor esses conceitos de domínios temos que entender como a bridge funciona. Em suma, quando uma bridge recebe um pacote ela consulta a tabela MAC procurando uma ocorrência do MAC de origem. Ao encontrar, ele sabe se deve ou não encaminhar esse pacote para a outra interface. Caso ele não encontra ele permite a passagem. Pode-se dizer que a bridge é um "firewall de camada 2" que cria suas regras dinamicamente. Abaixo uma animação de como a bridge funciona:

Switching



O cenário é simples:

- Dois segmentos de rede interligados por uma bridge;
- Em cada segmento tem um Hub que, como sabemos, funciona como um barramento, repetindo o pacote recebido por todas as suas portas;
- Temos 6 hosts com os endereços MACs escritos acima deles;
- A bridge possui duas interfaces eth1 e eth2: A eth1 se conecta ao segmento da esquerda enquanto a eth2 se conecta ao segmento da direita.

Algumas "legendas":

- Quando os "cabos de rede" ficam vermelhos eles indicam que o pacote passa por todos esses cabos.
- O envelope que aparece em baixo é o quadro de camada dois com os campos de MAC de origem (SRC) e MAC de destino (DST).

Passo 1

O host de MAC AAA quer se comunicar com o host BBB (não é BigBrother!!!). Ele monta um quadro com os dados de cabeçalho/trailer e envia o quadro pelo meio físico. Como ele está conectado a um HUB, todos os outros hosts conectados a esse HUB "recebem" esse pacote. Porque "recebem" entre aspas? Porque todos os hosts (com excessão do BBB e da Bridge) recebem mas não precessam, a pilha TCP/IP descarta o quadro porque o MAC de destino não é o deles. É aquela analogia da sala com várias pessoas feita anteriormente apesar de todos te ouvirem ninguém dá atenção pra sua conversa porque você não os chamou pelo nome.

Como o quadro é repetido para todos ele chega até o destino (BBB), mas antes vamos ver o que está acontecendo com a Bridge...

Ainda no passo 1, a Bridge recebe o quadro na interface eth1, mas como ela não é um host ela não tem o mesmo comportamente que eles (descartar o quadro). A Bridge faz mais ou menos assim:

Opa, chegou um quadro!! De onde veio? Do AAA pela eth1. Hum... isso pode ser útil, vou anotar! - Ela pega uma prancheta com uma tabela que mais parece uma lista de convidados a adiciona o MAC AAA na primeira coluna e a porta eth1 na segunda coluna - Pra onde esse quadro vai? Hum, é pro BBB... - Ela olha de novo para a tabela e comenta - Hum... Esse cara não tá na minha lsita.. mas como vovó dizia, antes pecar pelo excesso que pela falta! Vou deixar esse quadro passar!

Nisso o pacote passsa o quadro para o segmento da direita e o burro do HUB repassa pra todo mundo. Porém todos nesse segmento descartam o pacote...

-Pausa-

"Porque que a Bridge só anotou o MAC AAA??" Essa é a pergunta mais importante que você pode ser fazer quando estuda switching/bridging! Pelo simples fato dela só ter certeza de onde vem o pacote e não poder afirmar com certeza pra onde ele vai!

-Continuando-

De volta ao BBB: "Chegou um quadro pra mim! Vou responder...". E o quadro de retorno é enviado para o HUB tagarela que repete pra todo mundo, inclusive a bridge. A bridge ao receber o quadro verifica em sua tabela MAC: "Epa, outro quadro. Quam mandou?! BBB?? Esse eu não tenho, vou anotar... Epa mas pera ai!!! esse quadro que entrou pela eth1 é pro host AAA! Mas que p***a, o host AAA ta ai na eth1, quem foi o burro que mandou isso?! Não vai passar...". Assim a bridge impede o envio do quadro para o segmento da esquerda.

Mas mesmo com esse "bloqueio" o quadro chega com sucesso ao AAA, pois ele nunca deveria ter ido pra bridge.

Esse é o processo de aprendizagem da Bridge. Ela se baseia sempre no campo MAC de origem para descartar os próximos quadros. Mas na dúvida ela deixa o pacote passar. Com o passar do tepo ela vai ficando mais eficiente. E é isso que veremos nos próximos passos.
Obs: Acho que deu pra entender ne?! Vou ser mais breve!!

Passo 2
Agora a comunicação é de AAA pra FFF. Ao enviar o quadro o HUB repete e a bridge recebe. A bridge olha a origem e ve que já conhece AAA, então ela analisa o destino e vê que não conhece FFF, então deixa o quadro passar.

Do outro lado o HUB repete o pacote para todos e o FFF recebe o quadro e envia a resposta que é novamente repassada pelo HUB até chegar na Bridge. Ao receber, a bridge já anota o FFF como estando na eth2 e verifica que o destino, AAA, está d outro lado (direito), então ela encaminha o quadro.

Passo 3
Agora, um quadro de DDD pra FFF. O quadro é enviado e repetido até chegar na bridge (e no próprio FFF). A bridge ao receber o quadro anota o MAC DDD vinculado à eth2 e depois descarta o pacote, uma vez que FFF, de a cordo com a tabela está em eth2.

O FFF envia uma resposta, o HUB repete e a Bridge recebe, assim como o DDD. A bridge verifica o destino, DDD, e não permite a passagem para o outro lado pois DDD está na eth2.

Dessa forma, nessa "conversa" do segmento da esquerda não foi "interrompida" por um quadro desnecessário.

Vocês devem estar pensando: "Ah, que besteira, é só um quadro de vez em quando! Não mata ninguém!". Vamos imaginar dois grupos (grupo 1 e grupo 2) de amigos e a conversa rolando solta. Sempre no grupo de amigos tem um chato, logo temos o chato 1 e o chato 2. Então quando alguém quer conversar mandar o chato ir ver se o fulano ta no grupo 2. Pouco depois do chato 1 sair vem o chato 2 perguntar se sicrano ta ai! Atrapalhou a conversa de todo mundo certo?? "Ah, mas nem tanto!!" OK, imagine agora 50 grupos de amigos. Quando alguém do grupo 1 fala, o chato 1 tem que ir nos 50 grupos procurar o fulano. Não só isso, os outros 49 chatos virão incomodar seu grupo de amigos. E ai?! Se preocupou agora?! Pois é...

Esse processo de permitir ou não a passagem de quadros com base no endereço MAC é chamado de**switching/bridging** ou **comutação**.

Olá pessoal!!!

Esse vai ser um post bem simples! Eu estava imaginando em fazer ele mais completo porém não tive tempo de terminar tudo...

Primeiro queria avisar pra galera que coloquei um **CBox** (Chat box) aqui no blog! O intuito desse cbox é ter uma canal de comunicação "informal" com as pessoas que frequentam o blog. Se alguém quiser falar comigo pode utilizá-lo! Também vou utiliza-lo pra evitar postar coisas muito simples direto no blog, como alguma dúvida simples, sugestão desites, cursos, livros ou simplesmente notificar vocês de algum atraso ou problema que venha a ocorrer! Mas se for algo de muito significativo irei criar um post pra isso!

A outra notícia a a utilização de um **novo recurso** para os posts desse curso! Desde a semana passada venho brigando com o Kino e tentando aprender como utilizá-lo. Finalmente ontem consegui fazer algo útil!

Para quem não entendeu direito, ou simplesmente não acreditou, que o **HUB** funciona como um barramento e encaminha uma cópia do quadro para todos os segmentos conectados a ele, ai vai um vídeo de uma simulação feita no Cisco Packet Tracer.

Obs: Utilize o fullscreen para visualizar melhor as palavras...

http://under-linux.org/blogs/magnun/curso-de-redes-algumas-novidades-389/

Para quem não se lembra sobre o HUB pode ler sobre ele nesse post: Camada de Enalaçe de dados - Parte 2

Nesse vídeo coloquei legendas e indiquei alguns processos importantes, se tiverem alguma duvida perguntem agora pois, não vou detalhar tanto os próximos vídeos e irei cortar a montagem da topologia para reduzir o tamanho do vídeo.

O Cisco Packet Tracer é um simulador desenvolvido pela CISCO para fins de ensino. Eu baixo ele do site oficial da CISCO após fazer o logon no **Cisco Academy**. Mas com certeza deve ter em algum **"site não oficial"**. Quando pensei em fazer isso começei a pesquisar em como rodar o Packet Tracer pelo Wine. Depois de achar um tutorial fui baixar o instalador no site da cisco e me deparei com uma versão pra Linux, mais específicamente para **Ubuntu 7.04**. Mas a instalação correu sem problemas e o resultado é esse que vocês viram acima!

Para captura do vídeo utilizei o recordmydesktop, para edição o Kino e para gerar os textos utilizei o GIMP (mas estou estudando outra maneira para os textos). To levando um bom tempo pra fazer isso... Mas a longo prazo acho que vale mais apena que aqueles gifs que eu tava utilizando! e a visualização do processo com o packet tracer é mil vezes melhor...

Fiquem atentos pois novos vídeos estão vindo!

Se alguém tiver alguma crítica ou sugestão por utilizem os comentários (ou o Cbox)!!

Olá pessoal! Com essa nova *feature* do nosso curso de rede gratuito resolvi adiantar um pouco um conteúdo que eu estava segurando... Esse post é um "anexo" da parte 2 postada a alguns dias.

Vai ser bem curto porém com um conteúdo de **extrema** importância!

Protocolo ARP

O ARP (Address Resolution Protocol) é um dos protocolos mais importantes da Camada 3, ele é essencial para o funcionamento de uma rede ethernet.

Pode-se dizer que o ARP é um DNS de camada 2 e 3. Da mesma forma que o DNS vincula um FQDN (Fully Qualified Domain Name) a um IP o ARP vincula um MAC a um IP. Mas diferente do DNS o ARP não necessita de um servidor dedicado.

Uma comparação

Ao escrever uma carta, você põe no envelope o endereço residencial do destinatário e o CEP. Mas de onde você tira o CEP? Vai no site dos correios e consulta o CEP com base na quadra/bairro do endereço de destino. Mas aqueles números não fazem sentido certo?! Não fazem pra gente! Pors correios aquilo é extremamente útil pois informa para qual "agência dos correios" aquela carta deve ser enviada. Não existem CEPs repetidos, assim como não existem MACs repetidos, porém podem existir várias cidades chamadas Belem, São Paulo ou Brasília.

Cenário

Vamos imaginar que o host A (192.168.1.1) quer enviar um ping (ICMP) para o Host B (192.168.1.2) e que ambos estão interligados por um HUB (não há segmentação). A Camada 3 do Host A gera os dados ICMPs e consulta a**Tabela ARP** (não confunda com tabela MAC da bridge/switch). A tabela ARP armazena o par enderço IP:Endereço MAC. Caso o IP de destino não esteja na tabela ARP é realizada uma consula ARP (ARP Request).

Voltando pra nossa comparação. O Host A escreveu uma carta para o Host B e preencheu o endereço de origem: como SEPN 511, Brasília e o endereço de destino como Rua Verbo Divino, São Paulo. Ao levar a carta aos correios (camada 2) o agente do correio informou que faltava o CEP. O Host A, que era um Geek, sacou seu smartphone e acessou a internet para consultar o o CEP da Rua Verbo Divino em São Paulo. Após realizar a consulta ele preencheu os CEPs e enviou a carta.

Infelizmente nosso exemplo não é 100% fiel a realidade pois essa consulta não é tão simples quanto acessar um siteque possui uma base de dados (isso se assemelha ao DNS). O ARP é um pacote que possui como MAC de origem o MAC do Host A, e como MAC de destino o endereço de Broadcast da camada 2 (FF:FF:FF:FF:FF:FF), ou seja, **todomundo**! Dessa forma todos os dispositivos conectados a este segmento de rede irão receber e processar essa requisição. Mas como eles vão saber se é pra eles essa requisição? No capo dados existirá a seguinte string: "Who has 192.168.1.2?" que traduzindo seria: "Quem é 192.168.1.2?". Dessa forma somente o 192.168.1.2 irá responder a requisição ARP.

Ao tentar responder, o Host B já sabe o MAC e o IP do Host A, pois ele possui o pacote que chegou (se você recebe uma carta você pode responder pra quem te mandou simplesmente invertendo a origem com o destino certo?!). A resposta ARP (ARP Reply) é enviada e assim o Host A aprende o MAC do Host B. Após aprender o MAC ele pode finalmente enviar o ICMP que estava até agora pendente!

Simulação

Agora a simulação do processo explicado acima utilizando o Cisco Packet Tracer

http://under-linux.org/blogs/magnun/curso-de-redes-protocolo-arp-385/

Sempre que uma requisição ARP é realizada o par MAC:IP é armazenado na Tabela ARP. As estradas dessa tabela ARP expiram com o tempo, geralmente 2 min, para evitar que hajam dados desatualizados. Sempre que uma transmissão é realizada a tabela ARP é consultada. Se não houver a entrada na tabela ARP é realizada a consulta ARP. Caso haja uma entrada na tabela ARP é utilizado o mapeamento armazenado.

No Windows a tabela arp pode ser listada com o comando arp -a:
Código:

```
C:\>arp -a

Interface: 10.192.51.113 --- 0x10003
```

```
  Endereço IP           Endereço físico       Tipo
  10.192.51.1           00-05-32-14-0d-02     dinâmico
C:\>
```

No linux pode-se usar um comando com a mesma sintaxe:
Código:

```
arp -na
? (192.168.1.254) em 00:14:7f:01:0a:12 [ether] em eth0
? (192.168.1.65) em 00:0f:ea:d8:94:f7 [ether] em eth0
? (192.168.1.66) em 00:0f:ea:d8:94:f7 [ether] em eth0
? (192.168.1.101) em <incompleto> em eth0
? (192.168.1.10) em <incompleto> em eth0
```

Podemos ver que um MAC pode estar vinculado a vários IPs porém um IP não pode possuir vários MACs.

Existe também o RARP (Revers ARP) no qual é consultado um IP para um dado MAC. O protocolo RARP é utilizado em requisições DHCP que veremos mais a frente!

Olá pessoal nessa terceira parte vamos:

1. Firmar de vez o conceito de Domínio de Broadcast e Domínio de Colisão! Para isso vamos rever um pouco o funcionamento das bridge através de um vídeo.
2. Temos mais umas palavras para adicionar no nosso dicionário agora! Entre elas uma se destaca: Latência
3. Vamos também ver os Switches, seus modos de funcionamento e as VLANs!

Dessa vez, ao invés das legendas dos vídeos, eu narrei os vídeos enquanto simulava. Isso foi idéia do Scorpion! Se vocês preferem uma música de fundo e os textos, como antes, avisem que eu passo a fazer como antes. A narração fica até mais fácil pra mim porque torna a edição do vídeo mais simples. Mas fica a preferência de vocês.

Então mãos a obra!

Camada 2 - Enlace de Dados (Parte 3)

Antes de começar a ver sobre switches, vamos rever através de um vídeo o **funcionamento da Bridge** e com isso poderemos fechar o conceito de **Domínio de Colisão** e **Domínio de Broadcast**.

http://under-linux.org/blogs/magnun/curso-de-redes-camada-de-enlace-parte-3-410/

No vídeo acima temos **dois** Domínios de Colisão e **um** Domínio de Broadcast.

Vamos ver isso com calma:
Como podemos ver, sempre que é feita uma **requisição ARP** ela é encaminhada para o outro segmento, isso se deve ao MAC de destino que está no quadro: **FF:FF:FF:FF:FF:FF**. Esse é um endereço de **Broadcast** (para quem não lembra consulte sobre isso aqui). Do ponto de vista da **Bridge**, um endereço de broadcast é uma "ordem" para que encaminhe esse quadro a **todos os segmentos** a ela conectados. Do ponto de vista de **hosts**, um endereço de Broadcast indica que esse quadro está **endereçado** a TODOS os hosts.

Domínio de Broadcast

Podemos deduzir então que o domínio de broadcast são todos os segmentos que um broadcast atinge, nesse caso, toda a rede.

Domínio de Colisão

O Domínio de colisão é melhor visualizado com o seguinte vídeo:

http://under-linux.org/blogs/magnun/curso-de-redes-camada-de-enlace-parte-3-410/

Como visto na primeira parte do vídeo, se dois hosts interligados por um HUB tentarem transmitir ao mesmo tempo há uma colisão. Na segunda parte do vídeo (tivemos que utilizar um switch pois a bridge é limitada a 2 portas) vemos que se eles estiverem interligados por uma Bridge isso **não ocorre** pois a Bridge tem uma "inteligência" superior à do HUB. Cada **porta** da bridge funciona **independentemente**, já no HUB todas as portas são "interligadas" e o que entra por uma imediatamente sai pelas outras. Isso nos leva a conclusão que a Bridge insere uma **latência** (ou atraso, como preferir) na rede, já o HUB não. Mais a frente teremos uma melhor definição de latência.

Como a bridge "separa" o tráfego das portas ela **consegue separar também as colisões**. Se houver uma colisão em um segmento a Bridge **impede** que o outro segmento seja afetado pelo sinal de JAM ou pelos fragmentos da colisão.

Dessa forma vemos que a Bridge segmenta as colisões. Logo ela cria **dois Domínios de Colisão**.

Resumindo até agora:

1. Se temos uma rede apenas com hosts temos um Domínio de Colisão e um Domínio de Broadcast.
2. Se adicionarmos mais hosts à rede com um HUB nós estendemos o Domínio de Colisão e o Domínio de Broadcast.
3. Se adicionarmos mais hosts à rede com uma Bridge estaremos estendendo o Domínio de Broadcast porém reduzindo o Domínio de Colisão.

Mas qual a vantagem disso?? Ao reduzir o tamanho do domínio de colisão reduzimos o "alcance" das colisões e também sua ocorrência. Ao dizer "alcance" me refiro a hosts que irão parar de transmitir devido o sinal de JAM. Agora vocês devem esta se perguntando: "E se eu quiser reduzir o impacto dos Broadcasts??". Ai eu respondo, veremos isso um pouco mais pra frente, mas te adianto que os roteadores são os melhores amigos da sua rede!! 

Latência

A latência é o atraso entre o momento que o quadro começa a sair do dispositivo de origem e o momento que a primeira parte do quadro chega ao seu destino.

Formalmente: É o tempo que o primeiro bit leva para saír da origem e chegar ao destino.

Uma grande variedade de condições pode causar atrasos a medida que o quadro se propaga desde a origem até o destino:

1. Atrasos do meio físico causados pela velocidade finita em que os sinais podem se propagar através do meio físico.
2. Atrasos de circuito causados pelos circuitos eletrônicos que processam o sinal ao longo do caminho.
3. Atrasos de software causados pelas decisões que o software precisa tomar para implementar a comutação e os protocolos.
4. Atrasos causados pelo conteúdo do quadro e onde na comutação do quadro poderão ser feitas as decisões de comutação. Por exemplo, um dispositivo não pode rotear um quadro para um destino até que o endereço MAC de destino tenha sido descoberto

Swich

O switch é um **ativo** de rede muito importante. Ele pode, ao mesmo tempo, trazer para sua rede benefícios e malefícios (se mau administrado).

Vejo muita gente no Fórum fazendo perguntas do tipo: "tenho um HUB, vale a pena trocar por um switch?" Acho que a essa altura do curso vocês sabem responder essa resposta com segurança: SIM!!! Mas vejo dessas perguntas também: "O que eu devo comprar, um switch ou um roteador?!" Ai eu respondo: OS DOIS! Porque?! Por que um complementa o outro. Isso vai ficar claro mais adiante! Vamos começar a estudar o switch.

Um switch é essencialmente uma **bridge multiportas**, que pode conter dezenas de portas. Em vez de criar dois domínios de colisão, cada porta cria seu próprio domínio de colisão. Em uma rede de vinte nós, podem existir vinte domínios de colisão se cada nó for ligado em sua própria porta no switch. Um switch constrói e mantém dinamicamente uma **tabela MAC** contendo todas as informações MAC necessárias para cada porta.

Um switch tem alguns funcionamentos **diferentes de uma Bridge**. Vamos ver um vídeo de demonstração de alguns funcionamentos básicos do switch:

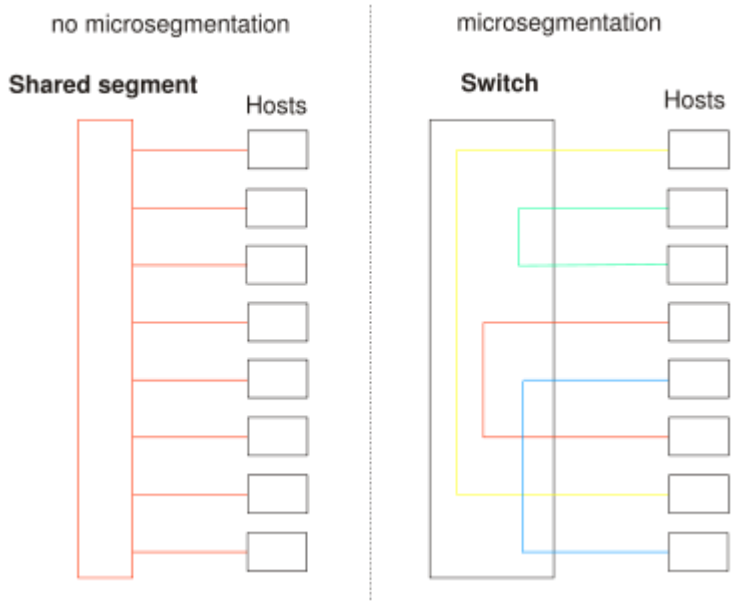
http://under-linux.org/blogs/magnun/curso-de-redes-camada-de-enlace-parte-3-410/

Nesse vídeo abordamos alguns aspectos interessantes:

- **Aprendizado:** O switch aprende da mesma forma que uma bridge, se baseando no MAC de origem dos quadros;
- **Encaminhamento de broadcasts:** Esse processo que o switch executa de replicar um quadro para todas as portas é chamado de Flooding;
- **No caso de dúvida:** Da mesma forma que a bridge, o switch em caso de dúvida permite o encaminhamento do quadro com uma única diferença, ele encaminha para todas as portas (Flooding);
- **Colisões em um switch:** Como se todos nossos hosts estiverem ligados ao switch a probabilidade de ocorrer uma colisão é muito menor.

Microsegmentação

Algo extremamente importante e que pode ser notado no vídeo é a microsegmentação realizada pelo switch. Quando um host tenta falar com outro pode-ser dizer que o switch fecha "um circuto exclusivo" **entre a origem e o destino**. Dessa forma a capacidade da sua rede é realmente aproveitada. É devido a microsegmentação que todos os hosts da sua rede podem se comunicar **ao mesmo tempo**. Abaixo um desenho que explica mais ou menos como isso funciona:



A maneira pela qual um quadro é comutado à sua porta de destino é uma concessão entre latência e confiabilidade.

Modos de Comutação

Um switch pode operar com três modos de comutação: cut-through, store-and-forward e fragment-free

Cut-through
No modo cut-through o switch começar a transferir o quadro assim que o endereço MAC de destino for recebido. A comutação cut-through resulta na redução da latência do switch no entanto, não oferece nenhuma verificação de erros.

Store-and-Forward

No modo store-and-forward o switch recebe o quadro completo antes de enviá-lo à porta de destino. Dando ao switch a oportunidade de verificar o FCS (Frame Check Sequence) para garantir que o quadro foi recebido com integridade antes de enviá-lo ao destino. Se o quadro for identificado como inválido, ele será descartado.

Fragment-free
O modo fragment-free é Uma solução intermediária entre os modos cut-through e store-and-forward. Nesse modo o switch lê os primeiros 64 bytes, que incluem o cabeçalho do quadro, e a comutação se inicia antes que sejam lidos todo o campo de dados e o checksum. Este modo verifica a confiabilidade das informações do endereçamento e do protocolo LLC (Logical Link Control) para garantir que o destino e o tratamento dos dados estejam corretos.

Quando se usa os métodos de comutação cut-through, tanto a porta de origem como a de destino precisam operar à mesma taxa de bits a fim de manter a integridade do quadro. Isto é conhecido como comutação simétrica. Se as taxas de bits não forem iguais, o quadro precisará ser armazenado com uma taxa de bits antes de ser enviado com outra taxa de bits. Isso é conhecido como comutação assimétrica. O modo Store-and-Forward precisa ser usado em comutação assímétrica.

A comutação assimétrica proporciona conexões comutadas entre portas com larguras de banda desiguais, como por exemplo uma combinação de 100 Mbps e 1000 Mbps. A comutação assimétrica é otimizada para os fluxos de tráfego cliente/servidor no qual vários clientes se comunicam simultaneamente com um servidor, exigindo mais largura de banda dedicada à porta do servidor para evitar um gargalo naquela porta.

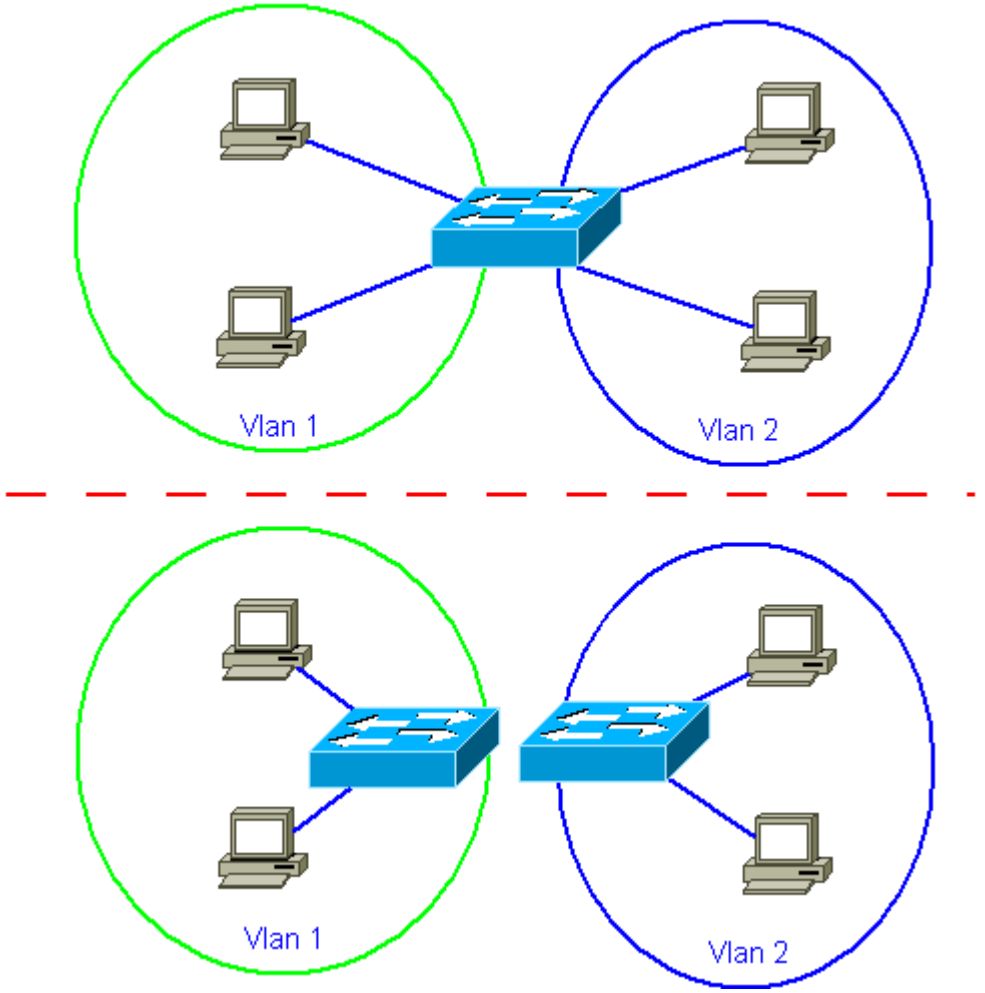
VLANs

VLAN (Virtual LAN) são domínios de Broadcasts "virtuais" compostos por **um ou mais** switches. Como vimos um switch estende o domínio de broadcasts e reduzem os domínios de colisão. Agora vou contar uma coisa: "Eu menti!". Tá eu não menti, é só que os switches tem uma função (diria malandragem) que faz com que ele segmente os domínios de broadcast. Isso é feito com VLANs.

Todo mundo fala de VLANs como sendo algo muito difícil. Mas na verdade é muito simples! O difícil são suas implicações! Vamos a um exemplo:

Na topologia acima temos um switch e 4 hosts. Os 2 hosts da esquerda estão em uma VLAN enquanto os 2 hosts da direita estão em outra VLAN. Como visto no vídeo **não há comunicação** entre os hosts se eles estão em VLANs diferentes, nem mesmo quando há Broadcasts.

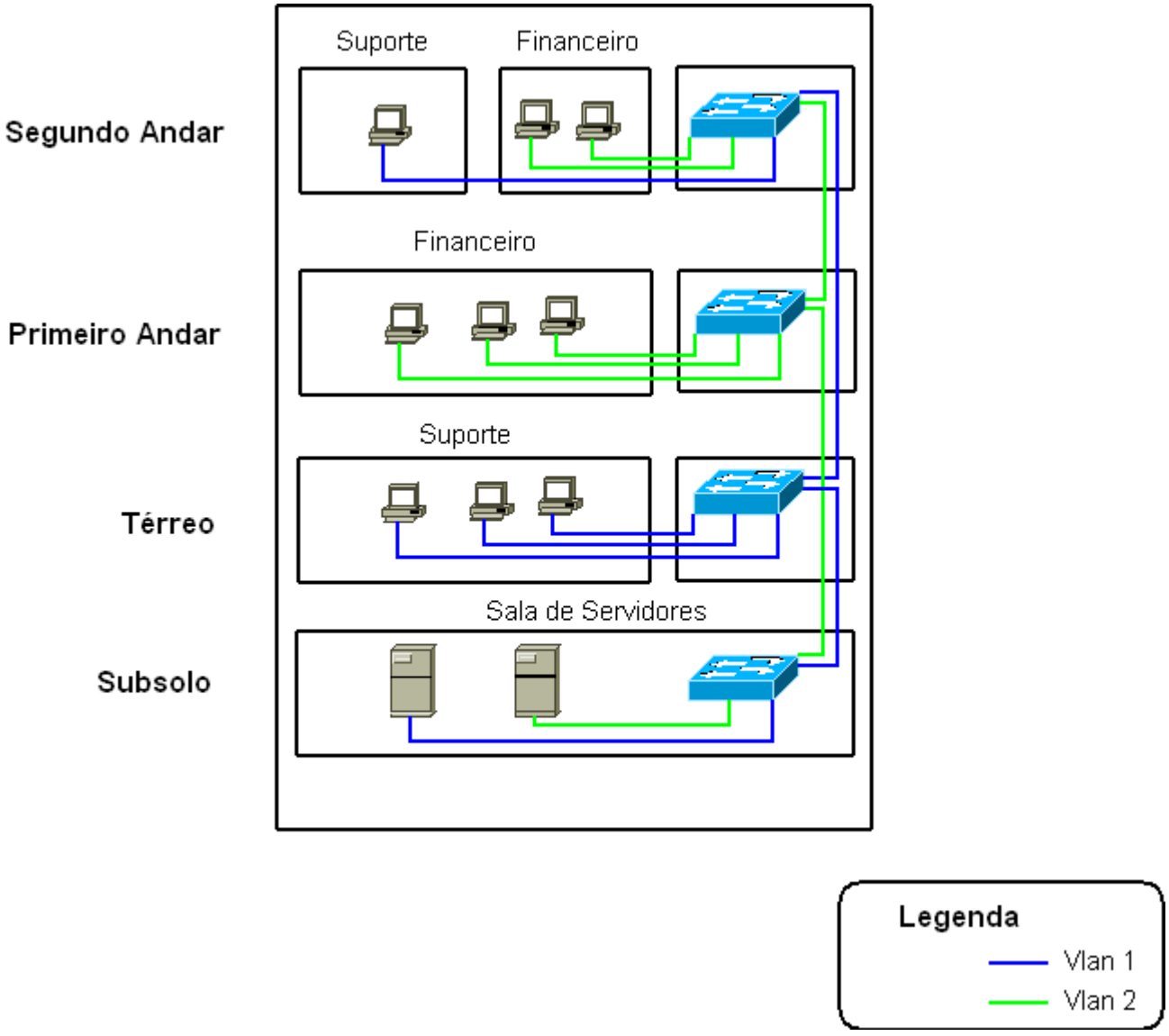
Pronto VLAN é isso! Acabou! Podemos dizer que quando criamos VLANs criamos switches virtuais. Uma relação de igualdade seria mais ou menos isso:



Quando criamos VLANs em um switch é como se "quebrássemos" o switch em switches menores. Vamos ver agora algumas aplicações de VLANs.

Imagine uma empresa onde temos alguns departamentos que não devem se comunicar. Por exemplo: O pessoal do suporte não pode ter acesso à rede do pessoal do financeiro. Imagina se alguém do suporte descobre que aquela planilha de pagamentos está numa pasta compartilhada em um PC do pessoal do financeiro! Ia ser bem divertido! Ou não...

As VLANs começam a ficar interessante quando precisamos de mais de um switch:



Aqui estendemos as VLANs através de outros switches. E da mesma forma o pessoal de uma VLAN não se comunica com o pessoal da outra. Interligamos essas VLANs utilizando um cabo para cada VLAN. Quem entende um pouco sobre configuração de VLANs deve ta me chamando de otário nesse exato momento! Temos uma maneira muito mais elegante de interligar essas VLANs: **Portas de Trunk**. Mas isso fica um pouco mais pra frente!

Referências

* Cisco CCNA - Guia de certificação do Exame, 3a Edição
Wendell Odom

* Redes de Computadores, 4a Edição
Andrew S. Tanenbaum,

* Conteúdo Cisco NetAcad, versão 3.1.1

Fechamento

Bem, fechamos agora a camada 2! Eu acabei por decidir em fechar essa camada por aqui mesmo. Sei que eu tinha dito que iria ensinar como configurar um switch mas achei mais prudente seguirmos adiante no conteúdo teórico e depois voltarmos em configurações. Eu não queria deixar essa camada 2 muito extensa. Inclusive ainda tem muita coisa

faltando na camada 2. Falta explicar como as VLANs funcionam, tagging, CLI do switch, portas de Trunk, VTP, STP, RSTP, PVSTP, PVGSTP, Port-Security, Entradas MAC...

Só STP ach oque daria o dobro do post de hoje!! Então vamos seguir em frente e após terminar as camadas do modelo ISO/OSI agente volta e começa a ver todos esses aspectos e protocolos incluindo os de camada 3, que são muito mais complexos e mais extensos!

Olá pessoal! Estou fazendo um novo "anexo" ao curso de redes. dessa vez é uma breve introdução a numerção binária. Isso é fundamental para redes! Então muita atenção e dedicação! Qualquer dúvida poste nos comentário ou use o ChatBox!

E pra quem ta acompanhando, não se preocupe que esse não vai ser o único conteúdo dessa semana! A próxima parte desse curso sai logo logo! E nela precisaremos desses conceitos!

Mãos à obra!!

Numeração Binária

Base Numérica

Uma base numerica se refere ao número de dígitos utilizados para expressar números. Ao longo da história tivemos diversas bases numéricas: Base 8, Base 16, Base 10, Base 8...

A numeração que utilizamos possui Base 10. Isso quer dizer que podemos escrever qualquer número utilizando apenas 10 algarismos:

0	1	2	3	4	5	6	7	8	9

A numeração com base dois utiliza apenas dois algarismos:

0	1

A numeração com base 8 utiliza os seguintes algarismos:

0	1	2	3	4	5	6	7

A numeração com base 16 utiliza os seguintes algarismos:

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Muita atenção pois, por exemplo, se o sistema tem base oito ele só chega até o digito 7. É muito comum as pessoas acharem que chega até o digito 8.

Como nosso objetivo é entender o endereçamento de camada 3, vamos ver somente numeros binários até 255 (1111 1111).

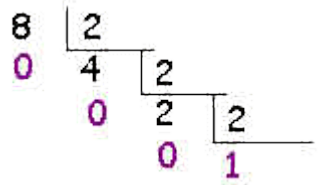
Conversão entre Bases

Vamos tomar alguns exemplos de um mesmo número escrito em diversas bases:

Base 10	Base 2	Base 8	Base 16
0	0000	0	0
1	00001	1	1
2	00010	2	2
3	00011	3	3
4	00100	4	4
5	00101	5	5
6	00110	6	6
7	00111	7	7
8	01000	10	8
9	01001	11	9
10	01010	12	A
11	01011	13	B
12	01100	14	C
13	01101	15	D
14	01110	16	E
15	01111	17	F
16	10000	20	10
17	10001	21	11

Para realizar a conversão de decimal para binário existem dois métodos: método das divisões e o método das subtrações.

No método da divisão são realizadas sucessivas divisões por zero a após chegar em 1 pegamos todos os restos em "ordem reversa".



Para o método da subtração, temo que ter em mente a seguinte tabela.

Decimal	Casa

Decimal	Casa
1	0000 000X
2	0000 00X0
4	0000 0X00
8	0000 X000
16	000X 0000
32	00X0 0000
64	0X00 0000
128	X000 0000

Vamos por exemplo converter 212 para binário. Para realizar a conversão vamos imaginar uma "caixa" com oito espaços: [X|X|X|X|X|X|X|X]. Em seguida pegamos o último Decimal da tabela (128). Se 128 for menor que o numero que queremos converter (212) nos subtraímos os dois: 212-128=84

Como foi possível dividir, marcamos um 1 na casa respectiva ao 128: [1|X|X|X|X|X|X|X]. Passamos para o próximo número: 64. O restante da subtração (84) é maior que 64?? Então realizamos a subtração: 84-64 = 20

Depois marcamos o 1 na sua casa respectiva: [1|1|X|X|X|X|X|X]. O próximo número é 32, só que 32 é maior que 20, logo não podemos subtrair. Então marcamos um 0 na sua casa respectiva: [1|1|0|X|X|X|X|X]

O próximo número é 16, realizamos a subtração (20-16=4) e adicionamos o 1 a sua casa: [1|1|0|1|X|X|X|X]

O próximo número é 8, não podemos subtrair! Marcamos a casa com 0: [1|1|0|1|0|X|X|X]

O próximo número é 4, realizamos a subtração (4-4=0) e marcamos a casa com 1: [1|1|0|1|0|1|X|X]

O próximo número é 2, não podemos subtrair! Marcamos a casa com 0: [1|1|0|1|0|1|0|X]

O próximo número é 1, não podemos subtrair! Marcamos a casa com 0: [1|1|0|1|0|1|0|0]

Assim temos nosso numero decimal convertido para binário: 11010100

A conversão inversa, binário para decimal é bem mais simples. Ao pegarmos um numer binário, por exemplo 1011 1101, vamos pensar na mesma tabela do método da subtração. Vamos ver quais "casa" desse numero possuem o 1:

1	0	**1**	**1**	**1**	**1**	0	**1**
X	X	**X**	**X**	**X**	**X**	X	**X**

Depois disso pegamos os números correspondentes a essas "casa" (128, 32, 16, 8, 4 e 1) e somamos:
128 + 32 + 16+ 8 + 4 +1 = 189

Pronto! 1011 1101 em binário é igual a 189 em decimal!

Operações Lógicas

Existem várias operações lógicas feitas com números binários. Essas operações muitas vezes são chamadas de lógica Booleana. Vamos ver duas delas:

AND
A operação AND é expressa pela seguinte equação A * B = Q (Lê-se: A and B é igual a Q)
Ela tem a seguinte tabela verdade:

A	**B**	**Q**
F	F	F
F	V	F
V	F	F
V	V	V

Vocês devem estar se perguntando: "V?!?! F?!?! Mas agente não tava falando de numeração binária???". Acontece que em computação o 0 é equivalente a falso e o 1 é equivalente a verdadeiro! Dessa forma podemos reescrever a tabela da seguinte forma:

A	**B**	**Q**
0	0	0
0	1	0
1	0	0
1	1	1

Toda operação lógica pode ser encarada como a veracidade de uma afirmação baseada na resposta amostrada por indivíduos.

Por exemplo:
Temos 2 pessoas (joão e josé). A afirmação é: João e José são irmãos. Logo faremos uma pergunta a ambos: "Sua mãe é a mesma que a dele?!". Podemos dizer que eles serão irmãos se a mãe do José **e** (AND em português) a mãe de João forem a mesma pessoa. As possíveis respostas (tabela verdade) serão:

- Se todos responderem que não, a afirmação é falsa;
- Se um deles responder que sim e o outro não a afirmação é falsa, pois um deles está mentindo;
- Se ambos responderem que sim, a afirmação é verdadeira!

Agora vamos ver como podemos fazer isso com vários números binários: 1010 * 1100

Para realizar essa operação vamos colocar um número sobre o outro e pensar em colunas:

A	1	0	1	0
B	1	1	0	0
Resultado	1	0	0	0

Passo-a-passo: Da coluna da direita para a esquerda, consultando na tabela verdade:
coluna 1: 0*0=0
coluna 2: 1*0=0
coluna 3: 0*1=0
coluna 4: 1*1=1

Reescrevendo: 1010*1100=1000

OR
A operação OR é expressa pela seguinte equação A + B = Q (Lê-se: A or B é igual a Q)
Ela tem a seguinte tabela verdade:

A	B	Q
F	F	F
F	V	V
V	F	V
V	V	V

ou

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

Podemos pensar da mesma forma que no caso da AND. Se tivermos uma sentença como, "Um deles gosta da cor preta", e perguntarmos ao José e ao João se ele gosta da cor preta, essa afirmação será verdadeira mesmo que apenas um deles goste da cor preta. A afirmação será false somente se a ambos não gostarem da cor preta. Podemos dizer que será verdade se João **ou** (OR em português) gostar da cor preta.

Pessoal, estamos chegando um momento crucial do nosso curso!! Agora vamos explicar sobre o endereçamento de camada 3. Ainda é um pouco introdutório, vamos ver o básico primeiro! Vamos ver sobre máscaras, endereço de rede, endereço de broadcast, endereçamento classfull e ter um overview do conceito de hierarquia.

No final há uns exercícios para vocês fixarem mais isso! Quem quiser me enviar as respostas por e-mail, ou mensagem pessoal aqui no forum, eu irei analisar e responder!

Camada 3 - Rede (Parte 1)

Camada de Rede

Essa camada define a entrega **fim a fim** dos pacotes definindo um endereçamento lógico, de forma que qualquer extremidade possa ser identificada, e métodos de roteamento para que qualquer grupo existente na rede possa ser alcançado.

Diferentemente do endereço de camadas inferiores que possuem, geralmente, um significado local, este endereçamento possui um **significado global** isto é, será utilizado e compreendido por toda a rede. Com este endereçamento de significado global é possível definir a forma como serão encaminhados os pacotes para que cheguem ao seu destino e a forma como os **roteadores** (dispositivos ativos desta camada) aprendem as regras que serão utilizadas para o encaminhamento dos pacotes.

É também responsabilidade desta camada definir como fragmentar um pacote em tamanhos menores tendo em vista a**MTU** (Maximum Transmision Unit - Unidade de transferência máxima) da tecnologia utilizada.

MTU é o maior "pacote" que o protocolo de uma camada pode suportar.

Classificação dos protocolos de Rede

Dentre as funções da camada de rede, o endereçamento e o roteamento podem ser destacados como sendo as principais. Esses dois serviços criam uma classificação de protocolos desta camada, são eles: Protocolos **Roteáveis**, Protocolos de **Roteamento** e protocolo **Não-Roteáveis**.

- **Protocolos Roteáveis ou Roteados** – São protocolos que especificam o endereçamento lógico referente à camada inter-rede. Os endereços especificados por estes protocolos são utilizados para a decisão de encaminhamento de um pacote. Alguns exemplos desses protocolos são o IP (Internet Protocol), o IPX (Internetwork Packet Exchange), o AppleTalk e o OSI;
- **Protocolos de Roteamento** – São responsáveis pelo preenchimento da tabela de roteamento. Esses protocolos especificam como as rotas serão aprendidas e divulgadas a outros roteadores. Geralmente diz-se que entre dois roteadores é "falado" um protocolo de roteamento. Alguns exemplos desses protocolos são o RIP (Routing Information Protocol) e o OSPF (Open Shortest Path First);
- **Protocolos Não-Roteáveis** - Um protocolo não-roteável é um protocolo que não pode ser encaminhado por roteadores. Geralmente isso se deve ao fato do protocolo não rodar sobre um procolo de camada 3. Alguns exemplos de protocolos não roteáveis são o DEC LAT e o NetBIOS ou NetBEUI.

O Endereçamento Hierárquico

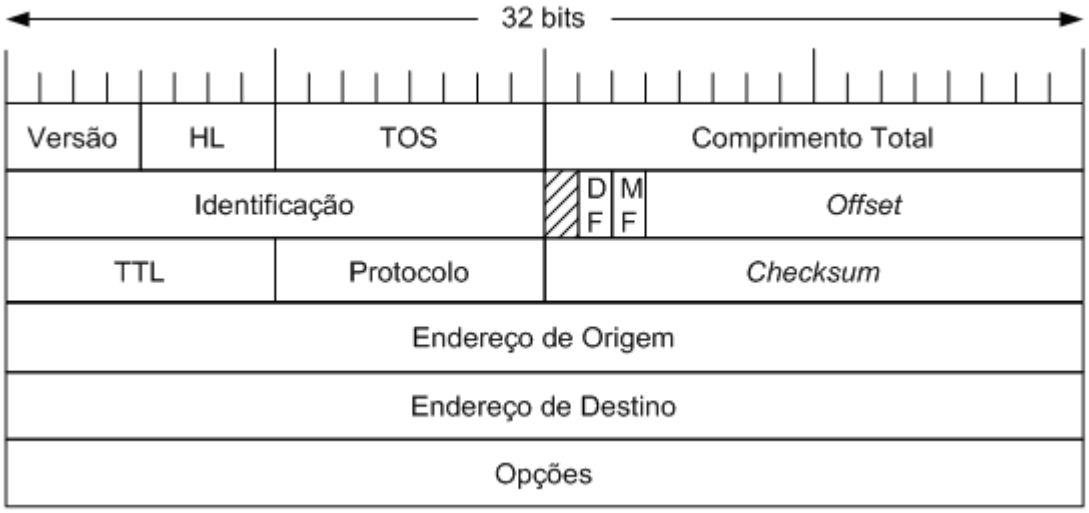
Para que dois sistemas quaisquer se comuniquem, eles precisam ser capazes de se **identificar** e **localizar** um ao outro. Para isso cada computador em uma rede recebe um identificador **exclusivo**, ou endereço. Na camada de rede este endereço é composto por uma identificação de um **grupo** (rede) e de **integrante** do grupo (host). Um endereço combina esses dois identificadores em um **único** número.

Esta estrutura cria um endereçamento hierárquico capaz de agrupar os hosts e endereça-los como um todo (uma rede). Em um ambiente como este um host só é capaz de receber dados de um integrante do **mesmo grupo** que o seu. Para a intercomunicação entre **redes diferentes** utiliza-se um roteador que deve possuir uma interface devidamente endereçada para cada rede.

Protoclo IP

Definido no RFC-790, o IP é o protocolo roteável utilizado atualmente na Internet. Sua versão 4 (IPv4 definido na RFC-791) tornou-se o protocolo de rede mais conhecido e utilizado atualmente.
Primeiramente adotado, no final dos anos 70, pela ARPA (Advanced Research Projects Agency) do DoD (Department of Defense) dos Estados Unidos no seu projeto ARPANET, precursora da internet e primeira rede de computadores operacional baseada em comutação de pacotes. No início dos anos 80 o DoD (Department of Defense) dos Estados Unidos adotou o IPv4 como sendo o protocolo obrigatório em sua rede (No documento MIL-STD-1777).

Da mesma forma que a camada 2, a camada 3 "pega" os dados passados pela camada superior a adiciona a eles algumas informações de protocolo/controle que chamados de **headers**. Lembrando que esse processo de adicionar dados de protocolo é chamado de encapsulamento. O protocolo IP define um pacote composto por um cabeçalho e um campo de dados. O seu cabeçalho é composto por uma parte fixa de 20 bytes e uma parte de comprimento variável. A seguir um esquema do cabeçalho utilizado pelo protocolo IP para realizar o encapsulamento:



- **Versão** - Este campo controla a versão do protocolo a que pertence o pacote;
- **HL (Header Length)** – Este campo indica o comprimento do cabeçalho – uma vez que este possui comprimento variável – em palavras de 32 bits. Seu valor mínimo é igual a 5;
- **TOS (Type of Service – Tipo de Serviço)** – Informa o tipo de serviço que o host deseja. Pode haver diversas combinações de confiabilidade e velocidade desejada. Esse campo é muito utilizado para prover alguns tipos de QoS;
- **Comprimento Total** – Comprimento total do pacote – tanto quadro quanto cabeçalho. O comprimento máximo é de 65.536 Bytes (MTU);
- **Identificação** – Permite ao host de destino determinar a que datagrama pertence um fragmento recém-chegado. Todos os fragmentos de um datagrama contêm o mesmo valor em Identificação;
- **DF (Don't Fragment – Não Fragmente)** – Este bit informa ao gateways para não fragmentar este pacote, devido ao destino ser incapaz de juntar novamente as partes;
- **MF (More Fragments – Mais Fragmentos)** – Todos os fragmentos exceto o último possui esse bit ativado. Ele indica que este pacote não é o último de uma seqüência de fragmentos;
- **Offset** – Informa a que posição no datagrama atual pertence o fragmento. Todos os fragmentos com exceção do último em um datagrama devem ser múltiplos de 8 bytes, a unidade elementar de fragmentos. Como é composto por 13 bits, há um máximo de 8192 fragmentos por datagrama, o que dá um comprimento máximo de 65.535 Bytes por datagrama, de acordo com o campo Comprimento Total;
- **TTL (Time to Live – Tempo de Vida)** – Inicialmente configurado como 255. A cada salto este campo é decrementado. No momento que ele atingir o valor zero este pacote será descartado. Este campo foi criado para evitar que pacotes trafeguem indefinidamente na rede;
- **Protocolo** – Este campo indica a qual, dentre os diversos processos da camada de transporte, pertence o datagrama. Este campo somente é analisado após a remontagem completa do quadro;
- **Checksum (Soma de Verificação)** – Esta verificação diz respeito somente ao cabeçalho;
- **Endereço de Origem e Endereço de Destino** – Indicam o numero de rede e de host da origem e do destino;
- **Opções** – Utilizado para informações de segurança, roteamento na origem, relatório de erros, depuração, fixação da hora e outros. Também utilizado para implementação compatível de versões subseqüentes.

Antes de vermos como é um endereço IP vale a pena vermos um pouco sobre numeração binária e operações lógicas com numeros binários.

Estrutura de um endereçamento IP

Um endereço IP é uma seqüência de **32 bits** (1s e 0s). Para facilitar a utilização do endereço IP ele é escrito no formato **decimal pontuado**. Neste formato, cada endereço é escrito em quatro partes separadas por pontos. Cada parte do endereço é denominada **octeto**, já que é formada de oito dígitos binários. Cada octeto varia entre **0 e 255**. Esta notação evita a grande quantidade de erros de transposição ou omissão que ocorreriam se fosse usada a numeração binária.

Juntamente com o endereço IP é apresentada a **máscara de rede**, ou mascara de sub-rede. Esta mascara possui o mesmo comprimento que o endereço IP. A função da máscara de rede é **identificar**, dentro do endereço IP, a porção

que identifica a **rede** e a porção que identifica o **host**. A máscara pode ser escrita de duas formas, em quatro octetos, que variam de 0 a 255, separados por pontos (da mesma forma que o endereço IP) ou por uma barra ("/") seguida de um numero decimal que varia entre **1 e 32**. O número que sucede a barra indica quantos dos 32 bits da máscara são 1s. Esta notação sempre sucede o endereço (endereço/máscara).

Para identificar a qual rede pertence, um host submete seu endereço IP e sua máscara à operação lógica **AND**. O resultado desta operação é o endereço de rede. O endereço de rede é usado para identificar uma rede. Nos endereços a porção que corresponde à identificação do host será sempre 0. Este endereço é utilizado pelos roteadores de uma rede para poder encaminhar um pacote para seu destino correto

Vamos tomar como exemplo o IP 192.168.1.32 e a máscara 255.255.255.0. Esse par, IP/máscara, pode ser reescrito da seguinte forma: 192.168.1.32/24. Porque /24? Vamos olhar a máscara com mais cuidado!

Primeiro converter 255 para binário: 1111 1111. Agora vamos reescrever a máscara:

Octeto 1	**Octeto 2**	**Octeto 3**	**Octeto 4**
1111 1111	1111 1111	1111 1111	0000 0000

Se contarmos as ocorrências do digito 1, teremos o número 24. Temos também 8 digitos 0, totalizando 32 dígitos (bits), que é o comprimento do endereço/máscara. Agora vamos ver como descobrir o endereço de Host e o endereço de rede:

Vamos converter o IP 192.168.1.32 para binário:

Octeto 1	**Octeto 2**	**Octeto 3**	**Octeto 4**
192	168	1	32
1100 0000	1010 1000	0000 0001	0010 0000

Vamos sobrepor a máscara e o IP e em seguida fazer um **AND** bit a bit:

	Octeto 1	**Octeto 2**	**Octeto 3**	**Octeto 4**
IP	1100 0000	1010 1000	0000 0001	0010 0000
Mascara	1111 1111	1111 1111	1111 1111	0000 0000
AND	1100 0000	1010 1000	0000 0001	0000 0000

Se convertermos o resultado da operação da AND para decimal teremos o IP 192.168.1.0, este é um endereço de rede. Se observarmos, a operação AND bit a bit, tem a função "inserir zeros", pois os zeros da mascara foi "inserido" no IP.

Agora se pegarmos a máscara, a invertermos (trocar uns por zeros) e realizar o mesmo procedimento teremos outro resultado:

	Octeto 1	**Octeto 2**	**Octeto 3**	**Octeto 4**
IP	1100 0000	1010 1000	0000 0001	0010 0000
Mascara	0000 0000	0000 0000	0000 0000	1111 1111

	Octeto 1	Octeto 2	Octeto 3	Octeto 4
AND	0000 0000	0000 0000	0000 0000	0010 0000

O resultado será 0.0.0.32, ou somente 32. Esse é o "identificador" do host. Podemos dizer que 192.168.1.32/24 é o host 32 da rede 192.168.1.0.

Mais um exemplo: o IP 172.16.32.124 com a máscara 255.255.0.0. Se realizarmos os mesmos procedimentos, podemos reescrever como 172.16.32.124/16 e teremos a rede 172.16.0.0 e o host 32.124.

Último exemplo: o IP 10.1.16.63 com a máscara 255.0.0.0. Se realizarmos os mesmos procedimentos, podemos reescrever como 10.1.16.63/8 e teremos a rede 10.0.0.0 e o host 1.16.63.

Como visto, a máscara define quantos hosts podem existir em uma rede.

Por exemplo, na rede 192.168.1.0/24 podemos ter do host 192.168.1.1 até o host 192.168.1.254, ou seja 254 hosts. Quem está prestando antenção deve ter se perguntado, "por que não até 192.168.1.255?", que é limite do octeto. Isso é um detalhe que será abordado em seguida!

A rede 172.16.0.0/16 pode ter do host 172.16.0.1 até o host 172.16.255.254, 65.024 hosts.

Por último, a rede 10.0.0.0/8 pode ter do host 10.0.0.1 até o host 10.255.255.254, 16.646.144 hosts.

Endereços Reservados

Existem alguns endereços que não podem ser utilizados no endereçamento de hosts:

- O **endereço de rede possui** a parte de host preenchida por zeros. Ele é reservado para identificação da rede e não deve ser atribuída a nenhum dispositivo. Acima, quando nos referenciávamos às rede sempre deixamosa porção do host como zero: 192.168.1.0/24
- O **endereço de Broadcast** possui a porção de host é 255 (todos os bits 1s). Este endereço é utilizado para enviar um pacote para todos os hosts conectados na rede. Um exemplo de endereço de broadcast será apresentado a seguir.

Broadcast de Camada 3

Da mesma forma que vimos na camada dois, um broadcast é uma transmissão que é endereçada para todos os dispositivos. A diferença de broadcast de camada 2 e de camada 3 é a forma como ele é feito (utilizando endereço de camda 2 ou 3) e sua "cobertura".

O endereço de broadcast possui a porção de host do endereço IP preenchida por 1. Vamos tomar como exemplo a rede 192.168.1.0/24. Convertendo para binário teremos:

	Rede	Rede	Rede	Host
Endereço de rede (decimal)	192	168	1	0
Endereço de rede (binário)	1100 0000	1010 1000	0000 0001	0000 0000
Endereço de broadcast (binário)	1100 0000	1010 1000	0000 0001	1111 1111
Endereço de broadcast (decimal)	192	168	1	255

Por isso anteriormente falamos que os endereços de host dessa rede vão de 192.168.1.1 a 192.168.1.254 e não até 192.168.1.255, pois este é reservado para Broadcast. Este endereço de broadcast é "mais restrito" pois, ele é

"endereçado" somente a rede 192.168.1.0. Existe um broadcast mais genérico que é o 255.255.255.255. Se fizermos uma comparação a grupos de pessoas, podemos dizer que o IP 192.168.1.255 seria a mesma coisa como alguém falando: "Ow! Pessoal do grupo 1..." já o 255.255.255.255 seria algo como: "Todo mundo!! Atenção ai...".

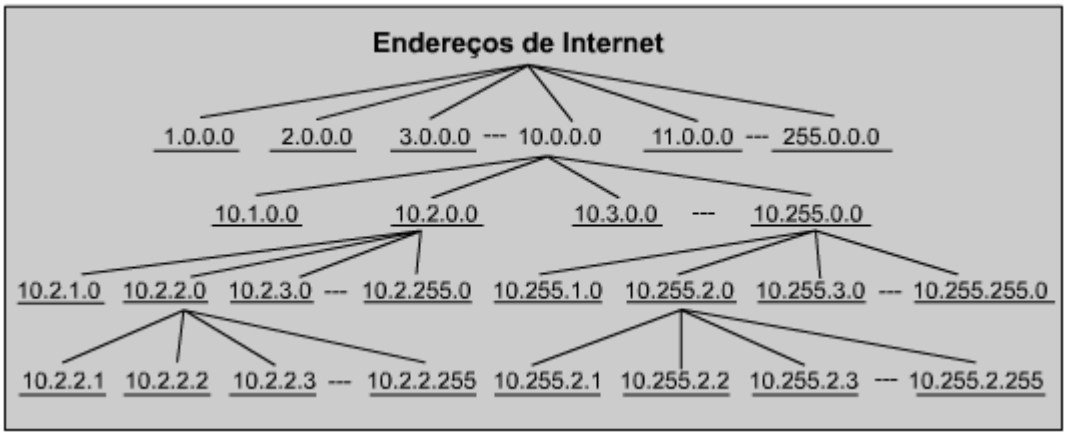
Dessa forma os Broadcasts podem ser divididos em **dois** tipos:

- O **broadcast local** é aquele restrito somente a rede ao qual o host está conectado, 255.255.255.255.
- O **broadcast remoto** é endereçado a uma rede remota isto é, a qual a origem não está diretamente conectada. Para isto deve existir o endereço de uma rede seguido da parte de host preenchida por 1s, exemplo: 192.168.1.255.

Hierarquia do endereçamento IP

Esse tipo de endereço é chamado de endereço hierárquico, porque contém diferentes níveis.

Ao se referir ao endereço do grupo diretamente acima de um grupo na hierarquia, todos os grupos que se ramificam desse endereço podem ser mencionados como uma única unidade



Endereçamento Classfull

Para acomodar redes de diferentes tamanhos e ajudar na classificação dessas redes, os endereços IP são divididos em grupos chamados classes. Isto é conhecido por endereçamento classfull. Essas classes definem redes pequenas, médias e grandes.

Cada endereço IP completo de 32 bits é dividido em uma parte da rede e uma parte do host. Um bit ou uma seqüência de bits no início de cada endereço determina a classe do endereço. Há cinco classes de endereços IP.

- Classe A – Redes maiores;
- Classe B – Redes de porte médio;
- Classe C – Redes pequenas;
- Classe D – Utilizado para multicast;
- Classe E – Reservado para utilização futura.

Classe A

O endereço de classe A foi criado para suportar redes extremamente grandes, com mais de 16 milhões de endereços de host disponíveis. Os endereços IP de classe A usam somente o primeiro octeto para indicar o endereço de rede. Os três octetos restantes são responsáveis pelos endereços de host.

O primeiro bit de um endereço de classe A é sempre 0. Com esse primeiro bit fixo como 0 o menor número que pode ser representado é 00000000 e o maior é 01111111, em decimal 0 e 127, respectivamente. Porém os endereços 0 e 127 são reservados e não podem ser usados como endereços de rede.

Qualquer endereço que comece com um valor entre 1 e 126 no primeiro octeto é um endereço de classe A. Esta classe pode discriminar 16.646.144 hosts em cada uma das suas 126 redes. A máscara de rede Classe A é 255.0.0.0, 8 bits.

Classe B

O endereço classe B foi criado para dar conta das necessidades de redes de porte médio a grande. Um endereço IP de classe B usa os dois primeiros octetos para indicar o endereço da rede. Os outros dois octetos especificam os endereços dos hosts.

Os dois primeiros bits do primeiro octeto de um endereço classe B são sempre 10. Os seis bits restantes podem ser preenchidos com 1s ou 0s. Portanto, o menor número que pode ser representado é 10000000, equivalente a 128 em decimal, enquanto o maior número que pode ser representado é 10111111, equivalente a 191 em decimal.

Qualquer endereço que comece com um valor no intervalo de 128 a 191 no primeiro octeto é um endereço classe B. Esta classe pode discriminar 65.024 hosts em cada uma das suas 15.120 redes (desconsiderando as redes reservadas). A máscara de rede Classe B é 255.255.0.0, 16 bits.

Classe C

Das classes de endereços originais, o espaço de endereços de classe C é o mais usado. Esse espaço de endereços tinha como objetivo suportar redes pequenas com no máximo 254 hosts.

Um endereço classe C começa com o binário 110. Assim, o menor número que pode ser representado é 11000000, equivalente a 192 em decimal e o maior número que pode ser representado é 11011111, equivalente a 223 em decimal.

Se um endereço contém um número entre 192 e 223 no primeiro octeto é um endereço classe C. Esta classe pode discriminar 2.023.680 redes (desconsiderando as redes reservadas). A máscara de rede Classe C é 255.255.255.0, 24 bits.

Classe D

O endereço classe D foi criado para permitir multicasting em um endereço IP. Um endereço de multicast é um endereço de rede exclusivo que direciona os pacotes com esse endereço de destino para grupos predefinidos de endereços IP. Assim, uma única estação pode transmitir simultaneamente um único fluxo de dados para vários destinatários.

O espaço de endereços de classe D, de forma muito semelhante aos outros espaços de endereços, é limitado matematicamente. Os primeiros quatro bits de um endereço classe D devem ser 1110. Assim, o intervalo de valores no primeiro octeto dos endereços de classe D vai de 11100000 a 11101111, ou de 224 a 239 em decimal.

Um endereço IP que comece com um valor no intervalo de 224 a 239 no primeiro octeto é um endereço classe D. Os endereços de classe D não possuem uma máscara padrão. Na RFC 3171 são designadas algumas redes com máscaras distintas e suas funções, porém generaliza-se a classe D com uma máscara de rede 240.0.0.0 ou /4.

Classe E

Também foi definido um endereço classe E. Entretanto, a IETF (Internet Engineering Task Force) reserva esses endereços para utilizações futuras. Dessa forma, nenhum endereço classe E foi liberado para uso na Internet. Os primeiros quatro bits de um endereço classe E são sempre definidos como 1s. Assim, o intervalo de valores no primeiro octeto dos endereços de Classe E vai de 11110000 a 11111111, ou de 240 a 255 em decimal.

Exercícios

1 - Converter para binários os seguintes IPs:

1. 172.31.128.12
2. 199.37.52.23
3. 230.220.2.1
4. 192.165.8.8

2 - Classificar como classe A, B ou C e calcular o endereço de rede, broadcast e de host dos seguintes IPs:

1. 172.16.32.54/16
2. 192.168.32/24
3. 192.168.1.255/24
4. 10.12.1.21/8

3 - [Desafio] Calcular o endereço de rede, broadcast e de host do seguinte IP: 192.168.1.172/28

Referências

* Cisco CCNA - Guia de certificação do Exame, 3a Edição
Wendell Odom

* Redes de Computadores, 4a Edição
Andrew S. Tanenbaum,

* Conteúdo Cisco NetAcad, versão 3.1.1

Galera!!

Chegamos em um momento decisivo do curso! Um dos conteúdos que as pessoas mais demoram pra assimilar é esse. Na verdade a camada 4 é mais difícil, mas a camada 3 é mais essencial pra quem está começando!

Qualquer **dúvida** utilize os **comentários** ou **chatbox** para as pessoas **não registradas** no Fórum!

Camada 3 - Rede (Parte 2)

O Protocolo ICMP

A operação de uma rede IP é monitorada rigorosamente pelos roteadores. Quando algo inesperado ocorre o evento é reportado pelo protocolo **ICMP** (Internet Control Message Protocol) definido na RFC 729. O protocolo ICMP também é utilizado para **testes** de rede através do comando **ping**.

O ICMP utiliza mensagens para realizar suas tarefas. Na sua RFC são definidas 16 mensagens, dentre estas as mais importantes são:

- **Destination Unreachable –** Utilizado quando sub-rede ou roteador não pode localizar o destino;
- **Time Exceeded –** Notifica o descarte do pacote, pois seu TTL atingiu o valor zero;
- **Source Quench –** Essa mensagem solicita ao emissor uma redução dos dados enviados;
- **Redirect –** Um roteador envia esta mensagem quando chega a ele um pacote porém existe uma rota melhor através de outro roteador;
- **Echo –** Usado pelo comando ping para verificar conectividade;
- **Echo Reply –** Resposta à requisição Echo.

Entrega de Pacotes

A entrega de pacotes na camada de rede pode ser dividida em dois tipos:

- **Entrega Direta –** Entrega de dados quando o remetente e destinatário se encontram na mesma rede lógica. Também conhecida como **Entrega Local**;

- **Entrega Indireta –** Entrega de dados quando o destinatário e remetente se encontram em redes lógicas diferente e se faz necessário a utilização de um "destino intermediário". Também conhecida como **Entrega Remota**.

Para saber se o host irá executar uma entrega local ou remota basta saber se o remetente encontra-se **na mesma rede que o destinatário**. Esta verificação é feita pelo **remetente** com base em dois dados: o resultado da operação lógica **"sua máscara de rede AND seu endereço"**; e da operação lógica **"sua máscara de rede AND o endereço do destino"**. Estes dois dados são comparados, se forem **iguais**, ambos estão na **mesma rede**, caso**contrário** estão em **redes distintas**.

Vamos ver **dois** exemplos:

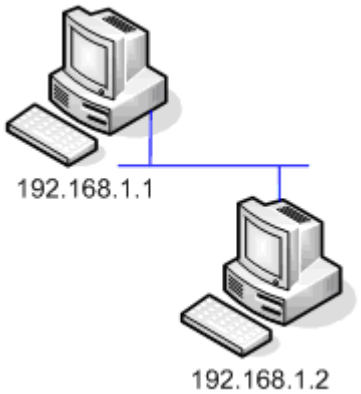
1. O host 192.168.1.10/24 que enviar uma mensagem para o host 192.168.1.32.
Primeiro o host de origem calcula a sua própria rede através de um AND entre o seu endereço IP (192.168.1.10) e a sua máscara (255.255.255.0) e obtém o resultado 192.168.1.0. Depois ele calcula a rede do host de destino realizando um AND entre o endereço IP de destino (192.168.1.32) e a sua máscara (255.255.255.0), obtendo o resultado 192.168.1.0. Como os dois resultados são idênticos o host de origem sabe que ambos estão na mesma rede lógica e que pode realizar uma entrega direta/local.

2. O host 192.168.1.10/24 que enviar uma mensagem para o host 192.168.2.20.
Primeiro o host de origem calcula a sua própria rede através de um AND entre o seu endereço IP (192.168.1.10) e a sua máscara (255.255.255.0) e obtém o resultado 192.168.1.0. Depois ele calcula a rede do host de destino realizando um AND entre o endereço IP de destino (192.168.2.20) e a sua máscara (255.255.255.0), obtendo o resultado 192.168.2.0. Como os dois resultados são distintos o host de origem sabe que ambos não estão na mesma rede lógica e que pode realizar uma entrega indireta/remota.

Entrega Direta/Local

Na entrega local o remetente precisa saber **todos** os dados do destinatário. Pois ao enviar o pacote o remetente preenche o endereço **IP de origem e destino** e o endereço **MAC de origem e destino**. Os Endereços MAC e IP de**origem** são conhecidos pelo remetente. Já o endereço IP de **destino** deve ser informado pela **aplicação** de camada superior, enquanto o endereço **MAC de destino** é descoberto através de uma **requisição ARP**. Desta forma a entrega direta pode ser separada em 4 etapas:

1. Preenchimento do endereço IP de origem e destino;
2. Requisição ARP para descobrir qual endereço MAC está vinculado ao endereço IP de destino;
3. Preenchimento do campo endereço MAC de origem e destino;
4. Envio do pacote.



Entrega Indireta/Remota

A entrega remota é aquela em que o remetente **não** se encontra no mesmo segmento de rede que o destinatário sendo necessário a entrega indireta isto é, o envio do pacote a um **destino intermediário** que irá redirecioná-lo para o destinatário, caso possível, ou para um **outro** destino intermediário. A esses sucessivos encaminhamentos a destinos intermediários dá-se o nome de **roteamento**. Aos dispositivos (destinos intermediários) que executam este

roteamento é dado o nome de **roteador**.

O roteamento se baseia no princípio que um grupo **não** estará em duas localizações distintas, isto é todos os hosts de uma mesma rede estarão agrupados em um **único enlace**. Ou seja, todos os hosts da rede 192.168.1.0, por exemplo, estarão agrupados e não espalhados ao longa da rede.

Como explicitado acima o roteamento pode ser interpretado como um processo de 3 etapas:

1. Enviar os dados para um roteador próximo;
2. Passar os dados deste roteador para outro, sucessivamente, até alcançar o último roteador;
3. Passar os dados do último roteador para o destino final.

Etapa 1 - Enviando os dados para um roteador próximo

Quando o host de origem não está diretamente conectado ao host de destino e é necessário fazer a entrega remota, host de origem envia os dados para seu ***gateway***. Desta forma, mesmo que a origem não tenha informações sobre a rede de destino, ele sabe que pode alcançá-la enviando os dados ao seu gateway. Não confundir **gateway** com**default gateway**, que será explicado em breve. Esse conceito ficará mais claro com as demonstrações!

Uma vez que este pacote será enviado a um roteador, mas este não é seu destino, no campo endereço **IP de destino**do cabeçalho IP irá constar o endereço do ***host* B**, porém o endereço **MAC de destino** será o endereço de **R1**.

Este preenchimento é necessário para que o roteador **receba** o quadro (através da camada de enlace) porem não o interprete como sendo **para ele**, pois o IP de destino não é dele, mas sim para que este seja **roteado** de acordo com sua tabela de roteamento. Ficou confuso né?! Vamos novamente, baseado na figura acima.

O Host A que mandar uma mensagem para o Host B porém ambos estão em redes diferentes. O Host A sabe que não irá alcançar o Host B diretamente então resolve utilizar-se do seu gateway (R1). Mas para que o R1 saiba pra quem enviar o pacote é preciso que exista essa informação. Então no campo IP de destino é mantido o IP do Host B e não o IP do R1 como muitos pensam. Mas então como o R1 recebe esse pacote? Porque no campo MAC de destino é colocado o Mac do R1. Dessa forma ao receber o pacote a camada 2 do R1 irá ver que o MAC de destino é o seu e ira receber o pacote. A camda 3 do R1 irá verificar que o IP de destino não é o do R1, dessa forma ele sabe que de encaminhar o pacote.

Para entender melhor podemos pensar nisso como uma correspondência. Nós colocamos o endereço no destinatário e entregamos a carta aos Correios. Os Correios se responsabilizam em entregar a carta ao seu destino. Nós não precisamos nos preocupar como essa carta chega lá, só temos que lembrar de entregá-la aos Correios. Da mesma forma que o Host A entrega o pacote ao R1 com o endereço do Host B.

Etapa 2 - Roteando dados em toda a rede

Após receber o quadro da camada de enlace, o roteador tentará descobrir o próximo destino deste pacote com base no **IP de destino** informado no *header* da camada de rede do pacote. Diferentemente do ocorrido na etapa anterior, a decisão de encaminhamento é tomada após analisar uma **tabela de roteamento** complexa podendo chegar a possuir dezenas de rotas. Estas rotas sempre correspondem a um grupo isto é, ela identifica a rede ao invés de *hosts*definidos. Para isto é necessário possuir o par endereço e máscara de rede. Para cada rota na tabela de roteamento está vinculado o *next hop* (próximo salto, endereço do próximo roteador ao qual deve ser enviado o pacote) e a interface de saída.

Descoberto o próximo salto o pacote será encaminhado com o endereço MAC de origem, roteador R1, e o MAC de destino do próximo salto indicado pela tabela de roteamento.

Todos os roteadores intermediários repetem esta operação até que o pacote seja entregue ao roteador diretamente conectado ao destino.

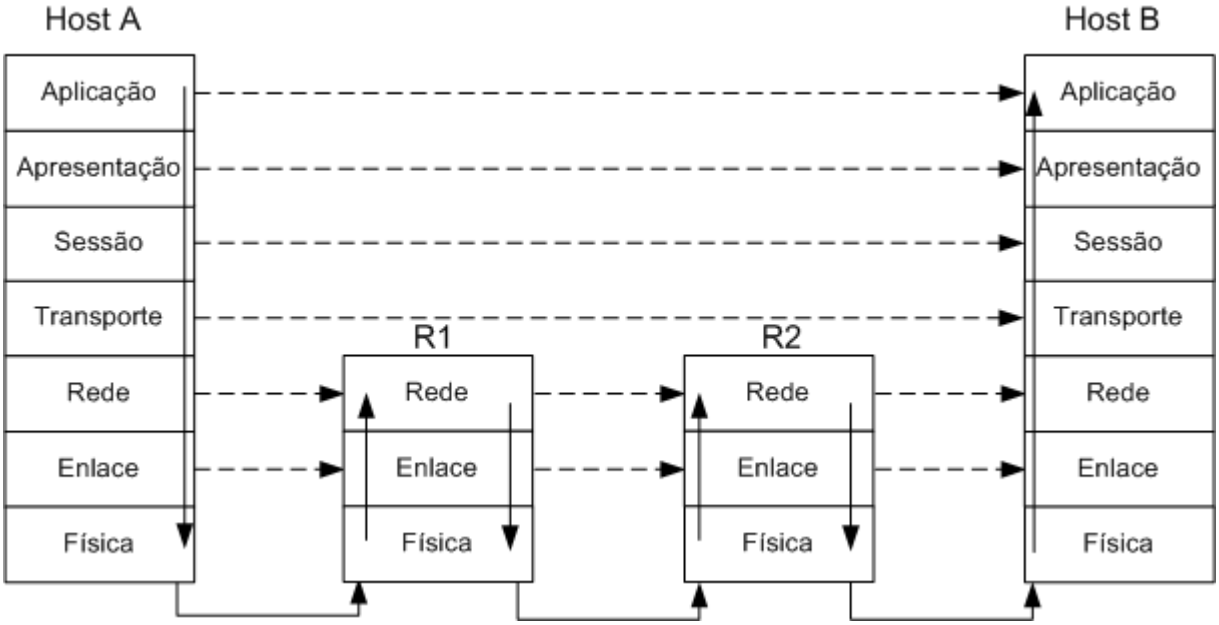
Etapa 3 - Passando os dados para o destino final

Quando R2 receber o quadro e fizer a consulta do **IP de destino** na sua **tabela de roteamento** ele ira constatar que está **diretamente conectado** ao destino, pois sua rota não indica o próximo salto, **somente a interface de saída**. Este é o único roteador que interpreta o endereço **IP de destino como um todo**, não somente o endereço de**rede** do destino como feito anteriormente.

Para encaminhar este quadro para seu destino final é necessário encaminhá-lo, porem dessa vez com o MAC de destino como sendo do *host* B, isto será feito através de uma resolução ARP. Após a consulta o quadro será enviado e o *host* B irá recebê-lo, pois o MAC e o IP que constam nos *headers* de suas respectivas camadas correspondem ao dele.

Pode-ser notar que o *header* da camada rede **não foi alterado** em nenhum momento, mantendo assim a origem e o destino original para que, se necessário, o *host* B saiba para quem enviar uma resposta.

Desta forma pode-se representar a entrega desse pacote através das camadas implementadas por cada dispositivo da seguinte forma:



Primeiro vamos entender o que significa essa imagem. Temos, representados pelas 7 camadas OSI, o host A e o host B. Os roteadores são representados apenas pelas camadas 1, 2 e 3. Isso se deve ao fato dos hosts implementarem as 7 camadas mas os roteadores apenas as 3 primeiras. As setas pontilhadas mostram a interação entre camadas e as setas com linhas normais mostram o fluxo dos dados.

Podemos ver que os dados são gerados nas camadas superiores do Host A e enviadas "para baixo" até a camada física. A camada física faz a comunicação com o R1 que recebe os dados. A camada de enlace do Host A se comunica com a camada de enlace do R1 e a camada de rede se comunica com a camada de rede do R1. Essa interação entre camadas pode ser entendida mais ou menos assim:
Camada 1: Fluxo de bits, não possui inteligência
Camada 2: Host A falando para o R1: "R1! esse **quadro** é pra você! Quem mando fui eu, o Host A"
Camada 3: Host A falando para o R1: "R1! esse **pacote** NÃO é pra você! é pro Host B! Encaminha ele. Quem mandou foi o Host A"

Esse processo se repete entre o R1 e R2:
Camada 1: Fluxo de bits, não possui inteligência
Camada 2: R1 falando para o R2: "R2! esse **quadro** é pra você! Quem mando fui eu, o R1"
Camada 3: R1 falando para o R2: "R2! esse **pacote** NÃO é pra você! é pro Host B! Encaminha ele. Quem mandou foi o Host A"

Agora entre o R2 e o Host B:
Camada 1: Fluxo de bits, não possui inteligência
Camada 2: R2 falando para o Host B: "Host B! esse **quadro** é pra você! Quem mando fui eu, o R2"
Camada 3: R2 falando para o Host B: "Host B! esse **pacote** é pra você! é pro Host B! Encaminha ele. Quem mandou foi o Host A"

Podemos ver que a mensagem de camada 3 não foi alterada ao longo da transmissão, diferente da mensagem da camada 2. Atentem para a diferença entre quadro e pacote! Vamos ver isso com mais calma adiante quando essa idéia de cabeçalho e protocolo estiver mais firme.

Já as mensagens da camada 4 em diante não são analisadas pelos roteadores, por isso essas camadas "se comunicam diretamente" da origem ao destino.

Protocolos de Roteamento

O roteamento é o processo usado por um roteador para encaminhar pacotes para a rede de destino. Todos os dispositivos ao longo do caminho usam o endereço **IP de destino** contido no **cabeçalho do pacote** para orientar-lo na direção correta, a fim de que ele chegue ao seu destino.

As decisões são tomadas com base nas rotas contidas em sua tabela de roteamento. Essas rotas podem ser estabelecidas através da utilização de duas classes dos protocolos de roteamento: Protocolos de **Roteamento Dinâmicos** e **Estáticos**.

Ao se utilizar um protocolo de **roteamento dinâmico**, essa informação é obtida dos outros roteadores, enquanto que com um protocolo de **roteamento estático**, as informações sobre as redes remotas são configuradas manualmente por meio de comandos na CLI (*command line interface*) ou pela interface gráfica de um roteador.

Roteamento Estático

Como as rotas estáticas precisam ser configuradas manualmente, qualquer alteração na topologia da rede requer que o administrador adicione e exclua rotas estáticas para refletir essas alterações.

Em uma rede grande, essa manutenção das tabelas de roteamento pode exigir uma quantidade enorme de tempo de administração. Mesmo possuindo desvantagens as rotas estáticas são utilizadas atualmente, porém em conjunto com um protocolo de roteamento dinâmico.

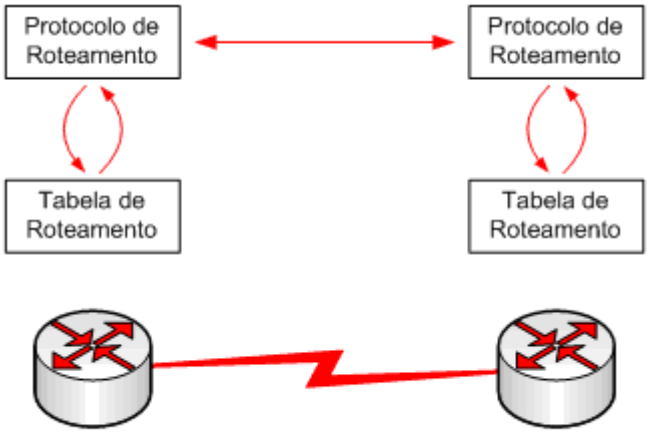
Roteamento Dinâmico

O roteamento dinâmico evita o demorado e rigoroso processo de configuração de rotas estáticas, possibilitando que os roteadores **reajam** a alterações na rede e **ajustem** suas tabelas de roteamento adequadamente, sem a **intervenção do ser humano**.

Para haver o roteamento dinâmico é necessário que os roteadores estejam utilizando um protocolo de roteamento. Um protocolo de roteamento é escolhido entre diversos outros com base em várias considerações: O tamanho da rede; a largura de banda dos *links* disponíveis; o poder de processamento dos roteadores; as marcas e modelos desses roteadores; e os protocolos que já estão em uso na rede.

Um protocolo de roteamento é a **comunicação** usada entre os roteadores, pode-se dizer que é uma **linguagem** entre roteadoes. Um protocolo de roteamento permite que um roteador **compartilhe informações** com outros roteadores a respeito das redes que ele conhece. Essas informações são usadas para construir e manter uma tabela de roteamento de forma dinâmica. Sempre que houver alteração na topologia de uma rede devido à expansão, re-configuração ou falha, a **base de conhecimentos** da rede (*network knowledgebase*) também deve mudar. Quando todos os roteadores de um grupo de redes interconectadas estiverem operando com as **mesmas informações** sobre a topologia da rede, diz-se que esse grupo **convergiu**. É desejável uma convergência rápida, pois isso reduz o período durante o qual os roteadores tomam decisões de roteamento incorretas.

Um protocolo de roteamento dinâmico escolhe a melhor rota com base em um fator chamado **métrica de roteamento**. A métrica pode ser composta de vários fatores, sendo eles: Latência, largura de banda, confiabilidade, carga, contagem de saltos e custo. Cada protocolo de roteamento dinâmico determina quais e como estes valores serão utilizados na fórmula para cálculo da métrica.



A imagem representa a comunicação entre os protocolos de roteamento resultando na distribuição alteração e alteração da tabela de roeamento.

A Tabela de Roteamento

Falamos diversas vezes sobre a tabela de roteamento hoje, então vamos ver uma básico dela!

Uma tabela de roteamento tem, pelo menos, as seguintes informações:

Rede de Destino	Próximo Salto	Interface de Saída
192.168.1.0/24	Diretamente Conectado	eth0
192.168.2.0/24	Diretamente Conectado	eth1
172.16.32.0/24	Diretamente Conectado	eth2
10.1.100.0/24	172.16.32.254	eth2
10.1.200.0/24	172.16.32.254	eth2

Nesse exemplo estou usando todos os IPs com a máscara 24 para simplificar nas próximas simulações. A tabela de roteamento pode dizer muitas coisas sobre um roteador. Por exemplo, olhando essa tabela vemos que esse roteador tem, pelo menos, 3 interfaces (eth0, eth1 e eth2) e muito provavelmente ele está conectado a duas redes locais (192.168.1.0/24 e 192.168.2.0/24) e apenas uma saída (172.16.32.0/24).

Isso é devido a informação que está na coluna próximo salto. Vemos que ele está diretamente conectado as redes 192.168.1.0/24, 192.168.2.0/24 e 172.16.32.0/24, o que significa que ele possui interfaces nessas redes. Podemos saber que interfaces são olhando a coluna "interface de saída".

Vemos também que nessa rede existem outras duas redes: 10.1.100.0/24 e 10.1.200.0/24. Que podem ser alcançadas pela eth2 desse roteador (pela rede 172.16.32.0/24). O próximo salta para essas redes (outro roteador) é o 172.16.32.254. Com isso podemos inclusive traça um diagrama simplificado da rede.

Agora, vamos fazer uma análise mais simples dessa tabela:

1. Se esse roteador precisar enviar um pacote para a rede 192.168.1.0/24 ele simplesmente envia para eth0;
2. Se esse roteador precisar enviar um pacote para a rede 192.168.2.0/24 ele simplesmente envia para eth1;

3. Se esse roteador precisar enviar um pacote para a rede 172.16.32.0/24 ele simplesmente envia para eth2;
4. Se esse roteador precisar enviar um pacote para a rede 10.1.100.0/24 ele deve encaminhar o pacote para o roteador (172.16.32.254);
5. Se esse roteador precisar enviar um pacote para a rede 10.1.200.0/24 ele deve encaminhar o pacote para o roteador (172.16.32.254).

Vemos ai o conceito de "roteador mais inteligente". Esse roteador considera que o 172.16.32.254 é mais inteligente que ele e pode encontrar as redes 10.1.100.0/24 e 10.1.200.0/24. Ele não precisa se preocupar como esse outro roteador vai fazer isso, ele simplesmente acredita que ele consegue!

Aqui entra o conceito de gateway. Dizemos que o gateway da rede 10.1.200.0 é o roteador 172.16.32.254. O gateway é o próximo salta. Trazindo ao pé da letra gateway é um portal, nesse caso o partal de acesso para uma determinada rede. Diferente de default gateway.

Então o que é default gateway?? Bem, essas duas últimas regras poderiam ser simplificadas utilizando um Default Gateway.

Rota *Default* ou *Default Gateway*

As rotas default são usadas para rotear pacotes com destinos que não correspondem a nenhuma das outras rotas da tabela de roteamento. Geralmente, os roteadores são configurados com uma rota default para o tráfego dirigido à Internet, já que normalmente é impraticável ou desnecessário manter rotas para todas as redes na Internet.

Uma rota default, na verdade, é uma rota estática especial endereçada a 0.0.0.0 e com máscara 0.0.0.0. Quando um IP de destino é submetido à operação lógica AND com a máscara definida resultará sempre na rede 0.0.0.0.

Utilizando o exercício anterior vamos reescrever a tabela:

Rede de Destino	Próximo Salto	Interface de Saída
192.168.1.0/24	Diretamente Conectado	eth0
192.168.2.0/24	Diretamente Conectado	eth1
172.16.32.0/24	Diretamente Conectado	eth2
0.0.0.0/0	172.16.32.254	eth2

Trocamos 2 rotas por uma única rota. Isso é útil em certos cenários. Vamos entender o que essa rota quer dizer: 0.0.0.0/0 = qualquer rede; 172.16.32.254 = envie para o 172.16.32.254. Ou seja, qualquer rede deve ser enviada para o roteador 172.16.32.254. "Como assim qualquer rede??" Qualque rede que não tenha sido especificada antes na tabela de roteamento. Para a galera de firewall iptables isso é similar a política padrão (default policy). O Default Gateway é utilizado quando o roteador não conseguiu descobrir pra onde ele tem que mandar esse pacote. Ele é a última opção, o último recurso.

Vamos a outro exemplo:

Rede de Destino	Próximo Salto	Interface de Saída
192.168.1.0/24	Diretamente Conectado	eth0
192.168.2.0/24	Diretamente Conectado	eth1
192.168.3.0/24	Diretamente Conectado	eth2
172.16.32.0/24	192.168.2.254	eth1
10.1.100.0/24	192.168.3.254	eth2

Rede de Destino	Próximo Salto	Interface de Saída
0.0.0.0/0	192.168.1.254	eth0

Vamos fazer umas "simulações". Tenha em mente que o roteador varre a tabela de cima para baixo em busca de uma ocorrência:

Pacote com destino a 192.168.3.12:

1. Olha na primeira linha: 192.168.3.0/24 é igual a 192.168.1.0/24? Não...
2. Olha na segunda linha: 192.168.3.0/24 é igual a 192.168.2.0/24? Não...
3. Olha na terceira linha: 192.168.3.0/24 é igual a 192.168.3.0/24? SIM!!!!
4. Como sou diretamente conectado a essa rede, só então envio o pacote pela eth2.

Pacote com detino a 10.1.100.33:

1. Olha a primeira linha. 10.1.100.0/24 é igual a 192.168.1.0/24? Não...
2. Olha a segunda linha. 10.1.100.0/24 é igual a 192.168.2.0/24? Não...
3. Olha a terceira linha. 10.1.100.0/24 é igual a 192.168.3.0/24? Não...
4. Olha a quarta linha. 10.1.100.0/24 é igual a 172.16.32.0/24? Não...
5. Olha a quinta linha. 10.1.100.0/24 é igual a 10.1.100.0/24? Sim!
6. Não sou diretamente conectado... Envio pro próximo salto: 192.168.3.254

Pacote com destino a 13.1.1.23:

1. Olha a primeira linha. 13.1.1.0/24 é igual a 192.168.1.0/24? Não...
2. Olha a segunda linha. 13.1.1.0/24 é igual a 192.168.2.0/24? Não...
3. Olha a terceira linha. 13.1.1.0/24 é igual a 192.168.3.0/24? Não...
4. Olha a quarta linha. 13.1.1.0/24 é igual a 172.16.32.0/24? Não...
5. Olha a quinta linha. 13.1.1.0/24 é igual a 10.1.100.0/24? Não...
6. Cheguei no default gateway (0.0.0.0/0)! Acabou minha tabela... O meu Default Gateway deve saber como chegar nessa rede, vou encaminhar pra ele.
7. Encaminha pra 192.168.1.254.

De grosso modo, essa é a inteligência do roteador!

Simulação

Vamos ver um vídeo que simula e mostra os aspectos aqui abordados:

http://under-linux.org/blogs/magnun/curso-de-redes-camada-de-rede-parte-2-490/

Conteúdo da Tabela de roteamento do roteador R1 nesse vídeo:

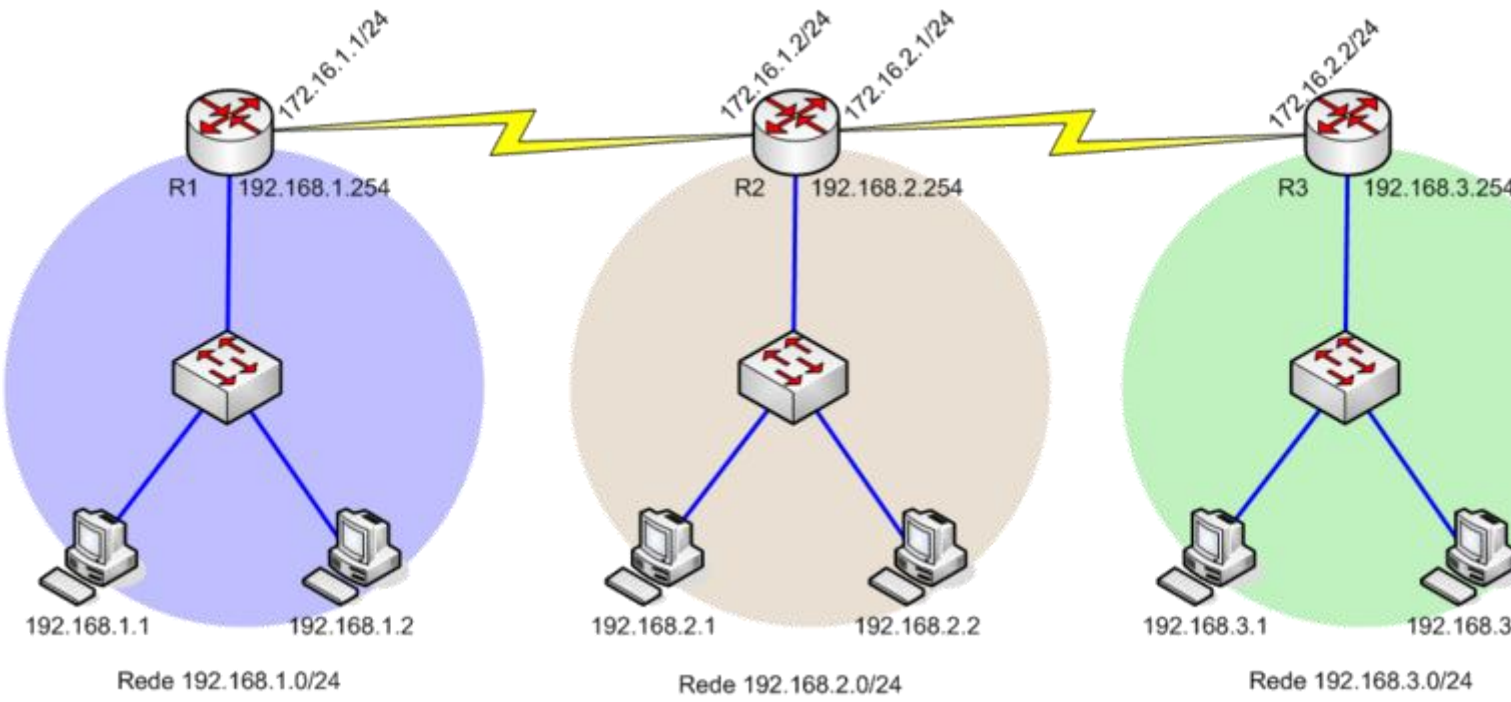
```
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
* - candidate default, U - per-user static route, o - ODR
P - periodic downloaded static route

Gateway of last resort is not set

C 192.168.1.0/24 is directly connected, FastEthernet0/0
C 192.168.2.0/24 is directly connected, FastEthernet1/0
Router#
```

Esse é um exemplo bem simples pois só possui redes diretamente conectadas. Essa rede se torna operacional com apenas 4 linhas de comando!

Vamos agora pensar um uma topologia mais completa:



Vamos analisar um pouco a topologia:

1. Temos 3 redes locais: 192.168.1.0/24, 192.168.2.0/24 e 192.168.3.0/24;
2. Cada rede tem 3 hosts: 1, 2 e 254;
3. O gateway das redes locais são sempre terminados em 254: 192.168.1.254, 192.168.2.254 e 192.168.3.254;
4. Existem 2 redes de interligação: 172.16.1.0/24 e 172.16.2.0/24. Elas são chamadas de redes de interligação pois não possuiem redes e são utilizadas somente para interligas as redes 192.168.1.0/24, 192.168.2.0/24 e 192.168.3.0/24.

Uma possível tabela de roteamento para os roteadores R1, R2 e R3 são:

R1

Rede de Destino	Próximo Salto	Interface de Saída
192.168.1.0/24	Diretamente Conectado	eth0
192.168.2.0/24	172.16.1.2	eth1
192.168.3.0/24	172.16.1.2	eth1
172.16.1.0/24	Diretamente Conectado	eth1
172.16.2.0/24	172.16.1.2	eth1

R2

Rede de Destino	Próximo Salto	Interface de Saída
192.168.1.0/24	172.16.1.1	eth1
192.168.2.0/24	Diretamente Conectado	eth0
192.168.3.0/24	172.16.2.2	eth1

Rede de Destino	Próximo Salto	Interface de Saída
172.16.1.0/24	Diretamente Conectado	eth1
172.16.2.0/24	Diretamente Conectado	eth2

R3

Rede de Destino	Próximo Salto	Interface de Saída
192.168.1.0/24	172.16.2.1	eth1
192.168.2.0/24	172.16.2.1	eth1
192.168.3.0/24	Diretamente Conectado	eth0
172.16.1.0/24	172.16.2.1	eth1
172.16.2.0/24	Diretamente Conectado	eth1

Ou podemos simplificar essas tabelas utilizando default gateways:

R1

Rede de Destino	Próximo Salto	Interface de Saída
192.168.1.0/24	Diretamente Conectado	eth0
172.16.1.0/24	Diretamente Conectado	eth1
0.0.0.0/0	172.16.1.2	eth1

R2

Rede de Destino	Próximo Salto	Interface de Saída
192.168.1.0/24	172.16.1.1	eth1
192.168.2.0/24	Diretamente Conectado	eth0
192.168.3.0/24	172.16.2.2	eth1
172.16.1.0/24	Diretamente Conectado	eth1
172.16.2.0/24	Diretamente Conectado	eth2

R3

Rede de Destino	Próximo Salto	Interface de Saída
192.168.3.0/24	Diretamente Conectado	eth0
172.16.2.0/24	Diretamente Conectado	eth1
0.0.0.0/0	172.16.2.1	eth1

Olá a todos!!

Estamos quase acabando a camada de rede! Creio que essa seja a penúltima parte. Ela é um pouco longa poque é um dos principais focos do nosso curso.

Muitos de vocês, que já tenham um conhecimento da área irão perceber que estão faltando alguns conteúdos. Eu estou saltando certas coisas pois esse é um curso introdutório e visa dar uma visão geral de redes. Então não acho que seja interessante entrar no mérito de protocolos IGP Link State e protocolos EGP. Mas futuramente (não muito distante) irei retornar com a parte intermediária desse curso, onde espero abordar esses protocolos e suas configurações.

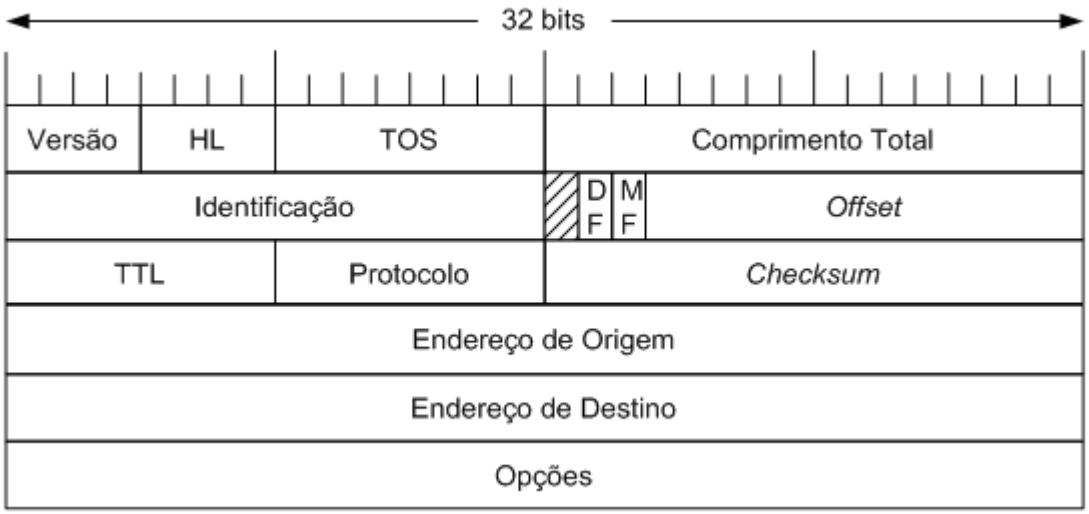
Ah, conforme solicitado pelo usuário Outorgas no comentário na última parte dessa série de posts os exercícios estão de volta (e retroativos!). Essa parte tem 6 exercícios, ficou um puco grande porque juntei com os exercícios que eu ia postar na parte 2 (na hora desanimei pela falta de retorno). Agradeço ao Outorgas por ter se mostrado interessado! As respostas podem ser enviadas atraves do sistema de mensagens particulares no Forum ou no meu e-mail da under: **magnun@underlinux.com.br**

Qualquer **dúvida** utilize os **comentários** ou **chatbox** para as pessoas **não registradas** no Fórum!

Camada 3 - Camada de Rede (Parte 3)

O Que é TTL

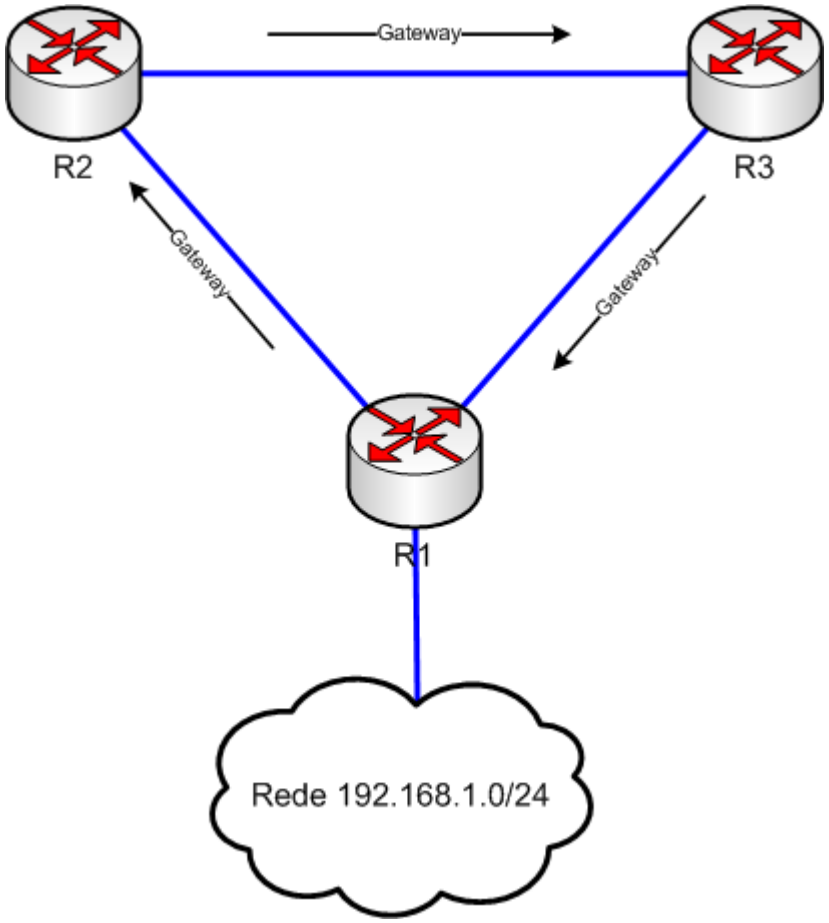
Antes de voltarmos para os protocolos de roteamento vale a pena ressaltar algumas informações que deixamos passar despercebido. Vamos rever o cabeçalho IP:



Nesse cabeçalho há um campo muito importante chamado TTL. Vamos rever sua função e compreende-la melhor:

TTL (*Time to Live* – Tempo de Vida) – Inicialmente configurado como 255. A cada salto este campo é decrementado. No momento que ele atingir o valor zero este pacote será descartado. Este campo foi criado para evitar que pacotes trafeguem indefinidamente na rede

Vamos imaginar uma topologia em loop:



Digamos que essa topologia foi criada por um administrador de redes que não prestou atenção nas aulas e colocou o default gateway dos roteadores da forma mostrada na imagem. R1 apontando para R2, R2 apontando para R3, R3 apontando para R1. Dessa forma oq aconteceria se alguém da rede 192.168.1.0/24 desse um ping para uma rede desconhecida, por exemplo: 200.200.200.1. Iria acontecer o seguinte fluxo:

1. O host de origem iria encaminhar o pacote para o R1;
2. R1 desconhece a rede, encaminha para o R2;
3. R2 desconhece a rede, encaminha para o R3;
4. R3 desconhece a rede, encaminha para o R1;
5. Volte para o passo 2.

Agora imagina se 20 hosts tiverem tentando acessar esse IP. Serão vinte pacotes rodando naquele triangulo. Isso gera processamento desnecessário nos roteadores, além de ocupar uma banda que poderia estar sendo utilizada.

Para evitar que esse pacote fique trafegando indefinidamente pela rede foi definido o campo TTL. O campo TTL, por padrão, é definido como 255. Cada vez que um roteador encaminha o pacote ele decrementa o conteúdo do campo TTL, quando o valor do campo chega a 0 o pacote é descartado pelo roteador.

Classificação dos Protocolos de Roteamento Dinâmico

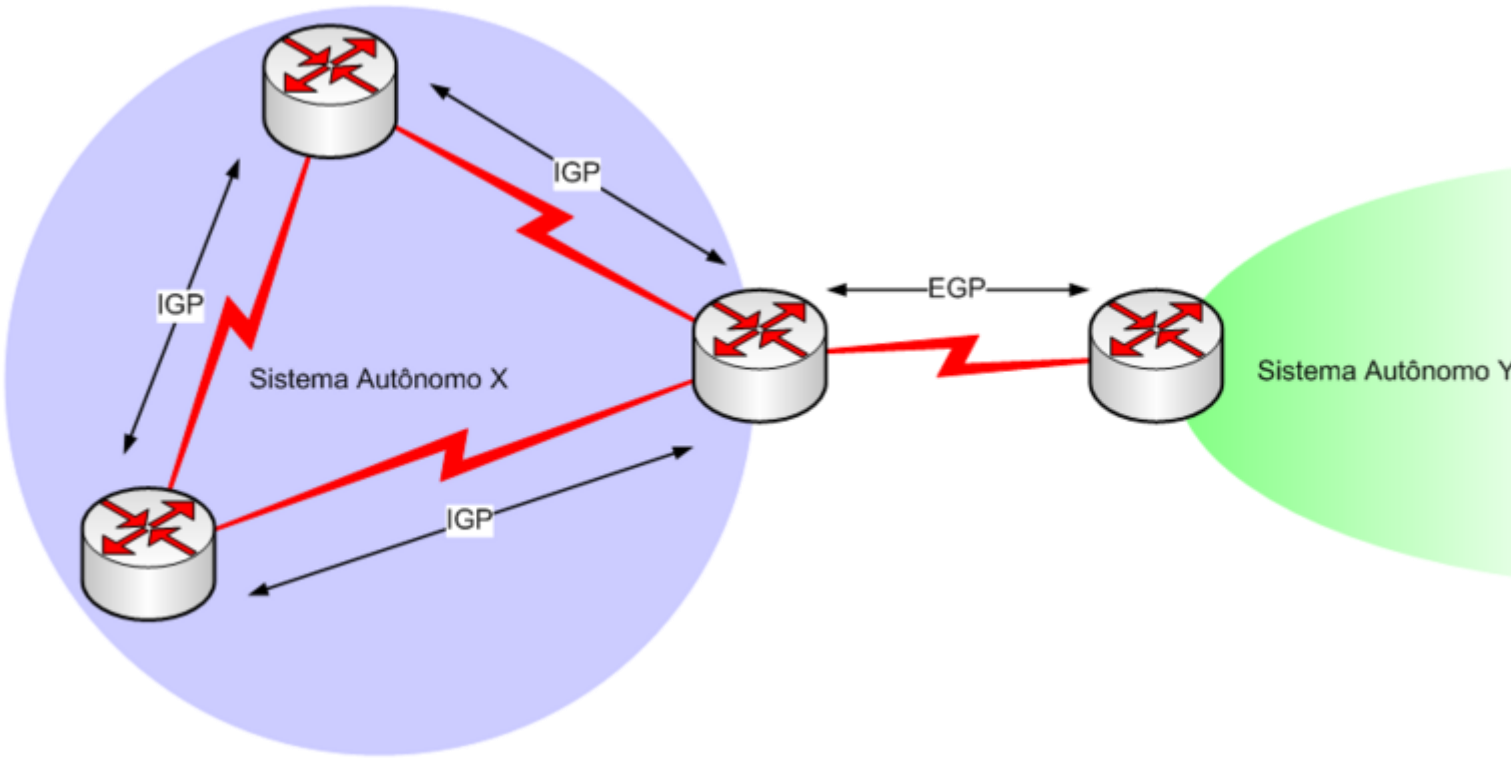
Os protocolos de roteamento são separados em duas classes: Quanto a onde ele é utilizado, IGP ou EGP, ao comportamento da lógica do protocolo, *Distance Vector* ou *Link-state*. Para tratar deste assunto, devemos primeiro definir Sistema Autônomo ou AS (*Autonomous System*).

Um AS é uma coleção de redes sob uma administração comum, que compartilha uma estratégia comum de roteamento. Para o mundo exterior, um AS é visto como uma única entidade. Cada AS tem seu próprio conjunto de regras e diretivas e um número de AS, composto por 16 bits, que o distingue de maneira exclusiva dos outros sistemas autônomos no resto do mundo. O ARIN (*American Registry of Internet Numbers*) gerencia a atribuição destes números de forma a não haver duplicação. O conceito de AS é mais um daqueles conceitos que você simplesmente tem que

aceitar e que você entende por osmose.

Protocolos IGP e EGP

Um protocolo de roteamento do tipo IGP (*Interior Gateway Protocol* – Protocolo de Roteameto Interno) é utilizado na intercomunicação de redes dentro de um mesmo Sistema Autônomo enquanto um EGP (*Exterior Gateway Protocol* – Protocolo de Roteamento Externo) é utilizado, dentre outras coisas, para a intercomunicação de diferentes Sistemas Autônomos.



Geralmente as pessoas pensam que protocolos EGP são somente para interconectar sistemas autônomos ou para ISPs (Internet Service Providers – Provedores de Internet). A verdade é que os protocolos EGP são mais interessantes quando você tem dois links com a internet e o "site" (não é site no sentido de página, mas site no sentido de grupo de ativos) possui um AS cadastrado, o que geralmente ocorre somente para empresas grandes. Dentre outras funções que o EGP pode prover é o balanceamento de links, redundância por rotas alternativas segmentação de tráfego por "sentido", dentre outras coisas.

Roteamento Distance Vector

A abordagem de roteamento pelo vetor de distância, ou algoritmo de roteamento Ford-Fulkerson*, determina a direção (vetor) e a distância para qualquer link no grupo de redes interconectadas.

O protocolo vetor de distância é também chamado de Algoritmo de Ford-Fulkerson, em homenagem aos inventores do algoritmo (L.R. Ford Jr. E DR Fulkerson, Flows in Networks, Princeton University Press, 1962), ou de algorítmo Bellman-Ford, porque foi baseado na Equação de Bellman (R. E. Bellman, Dynamic Programming, Princeton Univercity Press, 1957).

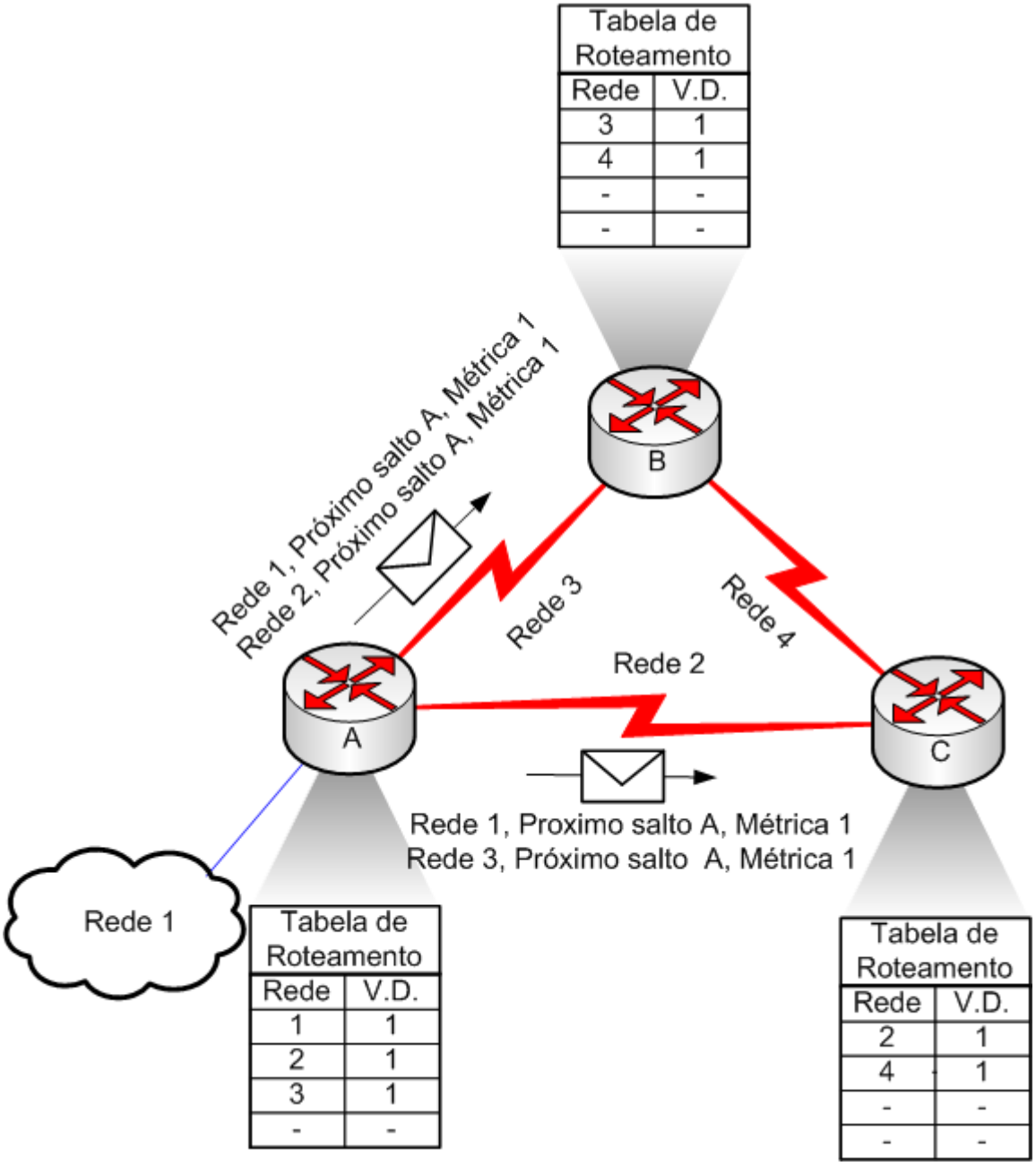
Este algoritmo é caracterizado por passar cópias periódicas da tabela de roteamento de um roteador para seu vizinho (mesmo enlace) em *broadcasts*. Essas atualizações periódicas entre os roteadores comunicam as alterações de topologia. Devido a isto o tempo de convergência de um protocolo vetor distância é considerado alto.

Quando um roteador recebe a tabela de roteamento do seu vizinho ele verifica as rotas que desconhece, incrementa o vetor de distância destas e acrescenta-as em sua tabela. O vetor de distância indica por quantos saltos, ou *hops*, este pacote irá passar até alcançar o destino. Para redes diretamente conectadas o vetor de distância é um.

http://under-linux.org/members/magnun/albums/imagens-para-posts/7113-vetor-distancia/

Com este algoritmo cada roteador vê somente os roteadores que são seus vizinhos, o que impede que um roteador conheça a topologia exata da rede, tornando difícil a prevenção de *loops* gerados por enlaces redundantes.

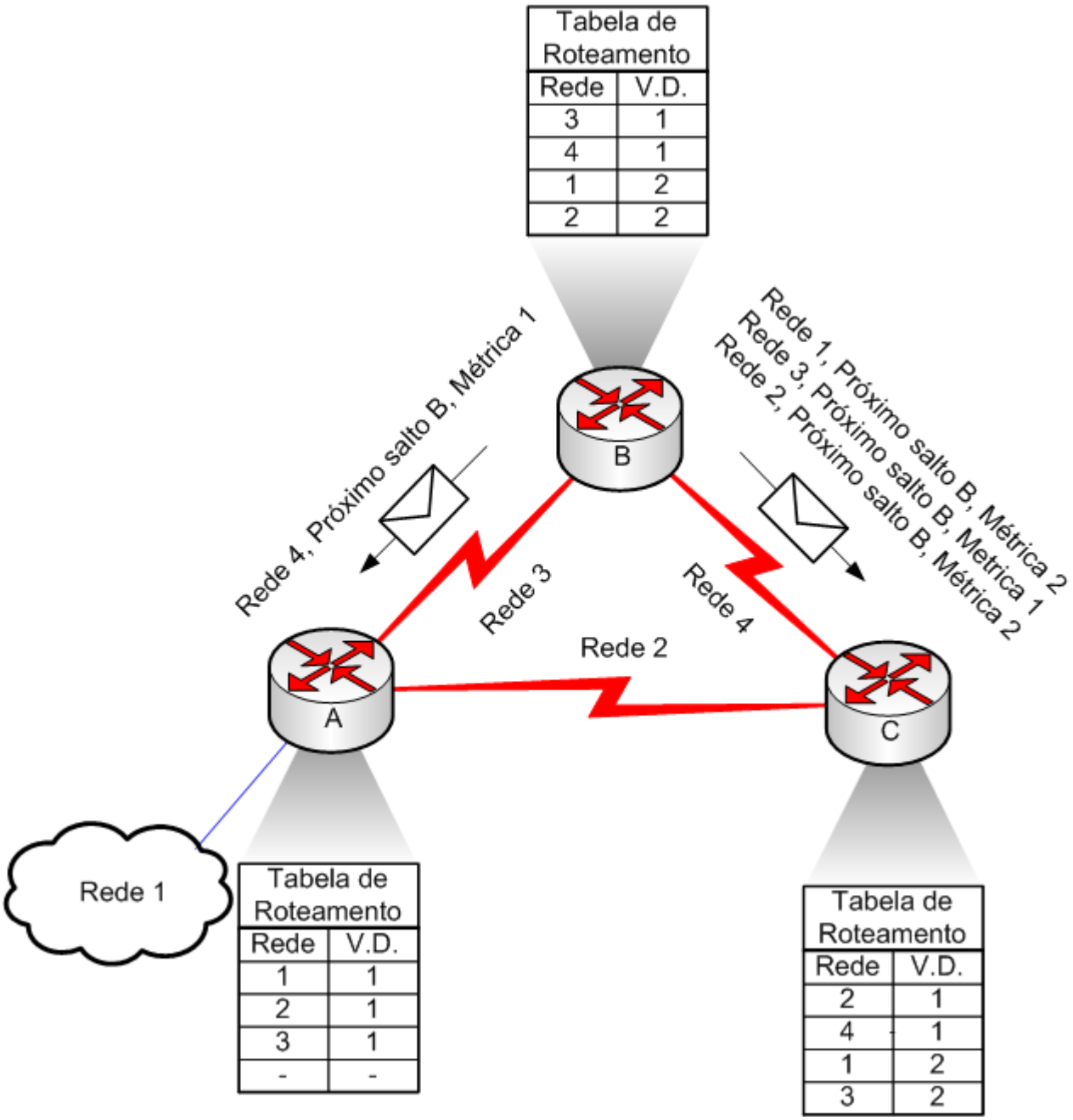
Juntamente com este protocolo de roteamento foi definida uma técnica chamada de *split horizon*. O *split horizon*consistem em não enviar informações sobre uma rede ao roteador que o informou sobre aquela rede. Isto faz com que um roteador não receba atualizações sobre uma rede já conhecida por ele e com uma métrica pior.



Esse é o momento inicial da atualização. A tabela de cada roteador está "resumida", apenas com a rede e com o vetor distância (V.D), ou métrica. O Roteador A envia sua tabela de roteamento para os hosts vizinhos. Vocês devem estar pensando: "Ei! Ele não ta enviando a tabela completa". Se você prestar atenção verá que o Roteador A anuncia as redes 1 e 3 para o roteador C. Porque ele não anuncia a rede 2? Porque ele sabe que o Roteador C já conhece a rede 2, pois é a rede que os interliga. O mesmo acontece com o Roteador B.

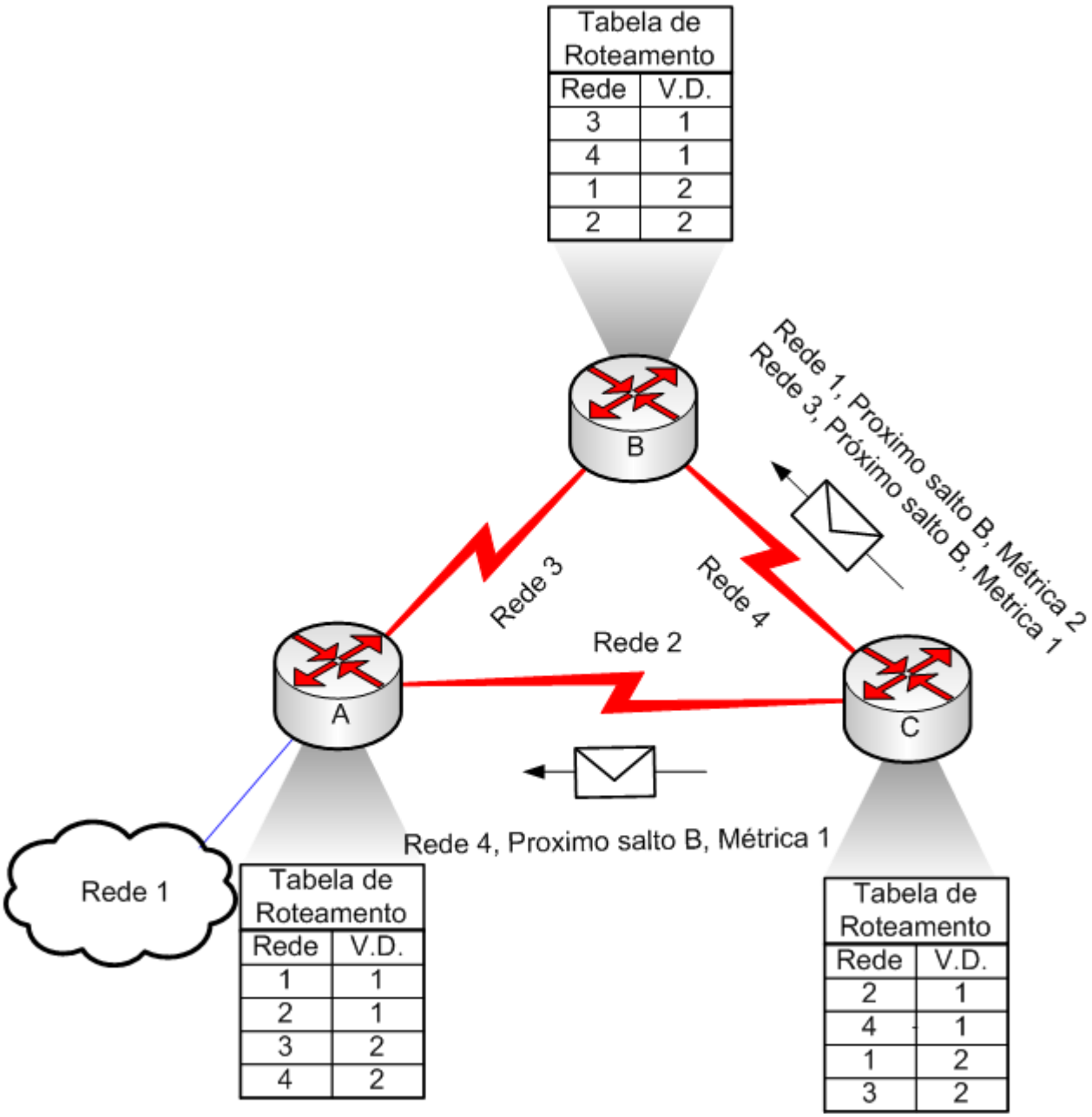
Vocês verão na próxima imagem que as rotas são anunciadas com o valor n mas armazenadas com o valor n+1. Isso porque o roteador conta a si mesmo como um salto.

Próximo estágio:



O roteador B anuncia suas rotas. "Agora piorou! O Roteador B só ta anunciando uma rede pro A!! Mas ele conhece 4 redes!". Isso é exatamente o Split Horizon em ação! Pergunto, que redes o B deveria anunciar? As redes 1, 2, 3 e 4. Vamos começar a eliminar. Qual a rede que interconecta os 2 roteadores?! Então por isso ele não anuncia a 3. Resta as redes 1, 2 e 4. De quem ele aprendeu as redes 1 e 2? Do roteador A! Então porque anunciar pra ele redes que você aprendeu dele?? Realmente é inútil! Até mesmo porque se ele anunciasse para ele essas regras seriam aprendidas com a métrica 3, que é bem pior que 1! Isso já não acontece com o anuncio pro Roteador C.

Último estágio:



O Roteador C anuncia para o roteador A somente a Rede 4. Da mesma forma que vimos anteriormente, ele não anuncia a rede 2, pois é diretamente conectada a ela, e nem as redes 1 e 3, pois ele aprendeu essas regras dele. Só tem um pequeno detalhe: O roteador A já conhece a rede 4. Ao receber ele vai analisar que essa nova rota possui a mesma métrica que a rota que ele já conhece. Nesse caso o roteador mantêm a rota existente e insere a nova, caso haja suporte (isso varia muito de implementação do protocolo de roteamento e fabricante) pode haver um balanceamento de carga entre os destinos. Algo similar acontece com os anúncios para o roteador B.

Endereços na Internet

A estabilidade da Internet depende diretamente da exclusividade dos endereços de rede usados publicamente. Endereços IP de rede duplicados impedem que o roteador realize sua função de selecionar o melhor caminho. Para cada dispositivo de uma rede, é necessário um endereço exclusivo.

Foi necessário criar um procedimento que garantisse que os endereços fossem realmente exclusivos. Inicialmente, uma organização conhecida como InterNIC (*Internet Network Information Center* – Centro de Informações da Rede Internet) cuidou desse procedimento. A InterNIC foi substituída pela IANA (*Internet Assigned Numbers Authority*). A IANA gerencia cuidadosamente o estoque de endereços IP para garantir que não haja duplicidade de endereços usados publicamente. A duplicidade causaria instabilidade na Internet e comprometeria sua capacidade de entregar datagramas para as redes.

Os endereços classe A e B representam 75% do espaço de endereços do IPv4, embora menos de 17.000 organizações possam receber um número de rede classes A ou B. Os endereços de rede de classe C são muito mais numerosos do que os de classes A e B, embora representem somente 12,5% dos 4 bilhões de possíveis endereços IP.

Infelizmente, os endereços de classe C estão limitados a 254 *hosts* utilizáveis. Isso não atende às necessidades de organizações maiores, que não podem adquirir um endereço de classes A ou B. Mesmo se houvesse mais endereços classe A, B ou C, um excesso de endereços de rede faria com que os roteadores da Internet viessem a parar sob o peso do enorme tamanho das tabelas de roteamento necessárias para armazenar as rotas para alcançar cada rede.

Durante as duas últimas décadas, foram desenvolvidas diversas extensões do IPv4. Essas extensões foram projetadas especificamente para melhorar a eficiência de utilização do espaço de endereços de 32 bits. Duas das mais importantes extensões são as máscaras de sub-rede e o roteamento interdomínios classless (CIDR).

Nesse meio tempo, foi definida e desenvolvida uma versão ainda mais extensível e escalonável do IP, o IP versão 6 (IPv6). O IPv6 usa 128 bits em vez dos 32 bits usados atualmente no IPv4. O IPv6 usa números hexadecimais para representar os 128 bits. Ele oferece 640 sextilhões de endereços. Essa versão do IP deve oferecer endereços suficientes para as futuras necessidades das comunicações.

Sub-redes (breve introdução)

O uso de sub-redes é um método usado para gerenciar endereços IP dividindo classes inteiras de endereços de redes em pedaços menores. Este método impediu, temporariamente, o esgotamento completo dos endereços IP.

Para redes grandes ou extremamente grandes, a divisão em sub-redes é necessária. Dividir uma rede em sub-redes significa usar a máscara de sub-rede para dividir a rede em segmentos menores, ou sub-redes, mais eficientes e mais fáceis de gerenciar. Gerando assim números maiores de redes pequenas.

É importante saber quantas sub-redes, ou redes, e quantos *hosts* serão necessários em cada rede. Com as sub-redes, a rede não fica limitada às máscaras de rede padrão de classes A, B ou C, e há maior flexibilidade no projeto da rede.

Os endereços de sub-rede incluem a parte da rede, mais um campo de sub-rede e um campo do *host*. O campo da sub-rede e o campo do *host* são criados a partir da parte do *host* original para toda a rede.

Endereços Privados

Os endereços IP privados é outra solução para o problema da escassez iminente dos endereços IP públicos. Como dito, as redes públicas exigem que os *hosts* tenham endereços IP exclusivos. Entretanto, as redes privadas que não estão conectadas à Internet podem usar quaisquer endereços de *host*, contanto que cada *host* dentro da rede privada seja exclusivo.

Muitas redes privadas existem em paralelo com as redes públicas. Porém, não é recomendável que uma rede privada use um endereço qualquer, pois essa rede pode ser conectada à Internet algum dia podendo causar erros no roteamento para certas redes públicas.

O RFC 1918 reserva três blocos de endereços IP para uso interno e privado. Esses três blocos consistem de um endereço de classe A, um intervalo de endereços de classe B e um intervalo de endereços de classe C:

- De 10.0.0.0/8 até 10.255.255.255/8;
- De 172.16.0.0/16 até 172.32.255.255/16;
- 192.168.0.0/24 até 192.168.255.255/24.

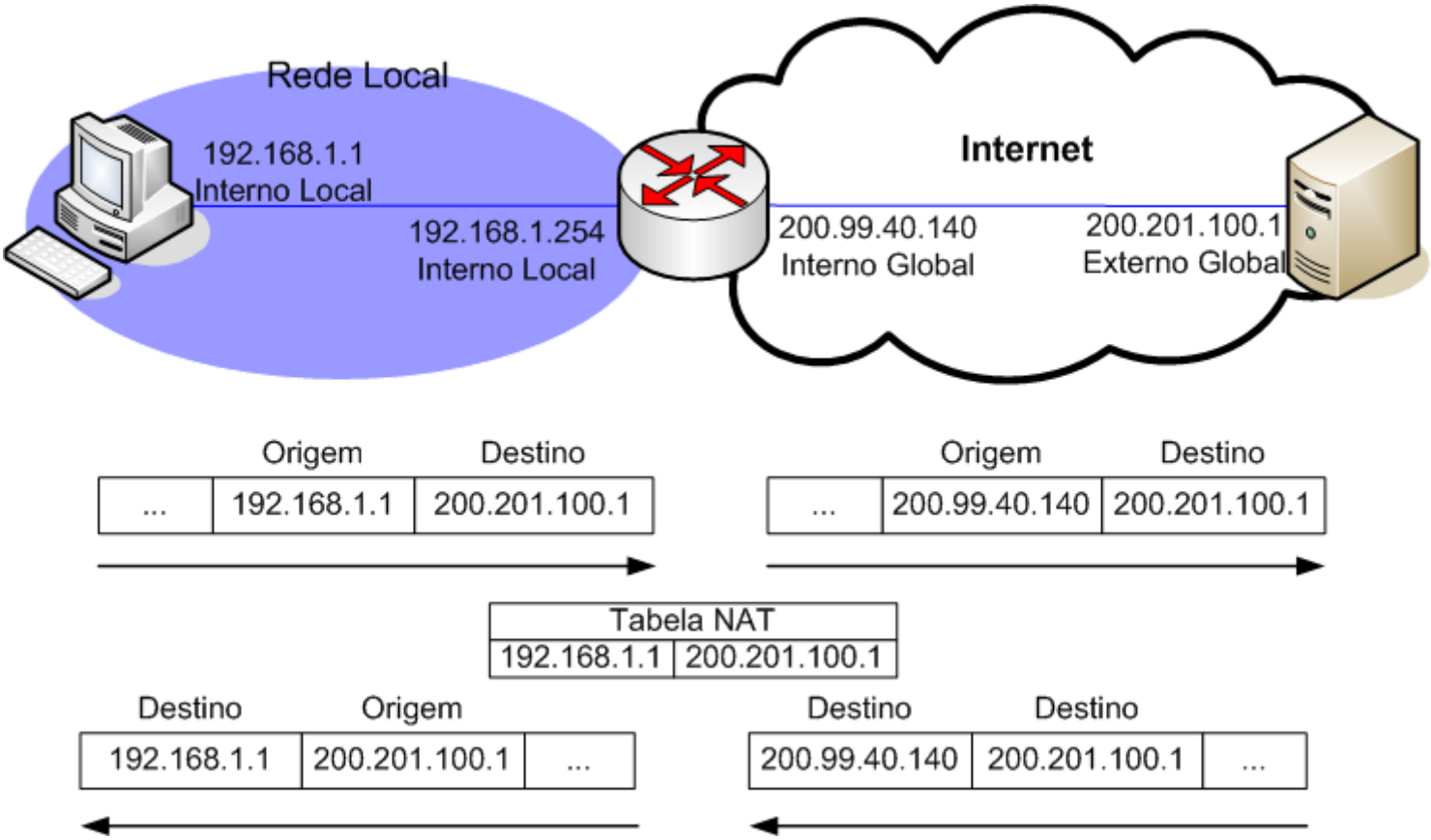
Os endereços dentro desses intervalos não são roteados no *backbone* da Internet. Os roteadores da Internet descartam imediatamente os endereços privados. Para endereçar uma intranet não-pública, um laboratório de testes ou uma rede doméstica, pode-se usar esses endereços privados no lugar dos endereços globalmente exclusivos.

A rede 127.0.0.0/8 é reservada para testes de *loopback*. Os roteadores ou as máquinas locais podem usar esse endereço para enviar pacotes para si mesmos. Por isso, esse número não pode ser atribuído a nenhuma rede.

NAT (Network Address Translation)

Definida na RFC-1631 a NAT (Tradução de Endereço de Rede) foi criada para reduzir o número de endereços públicos na internet permitindo que uma rede com endereço privado tenha acesso à internet. Para isto é feita a conversão dos endereços privados em endereços públicos.

Ao realizar uma NAT alguns endereços são mantidos e outros são alterados dependendo da direção do pacote em uma conexão. Um dispositivo habilitado para NAT geralmente opera na borda de uma rede *stub*. Uma rede *stub* é uma rede que tem uma única conexão para a rede externa.



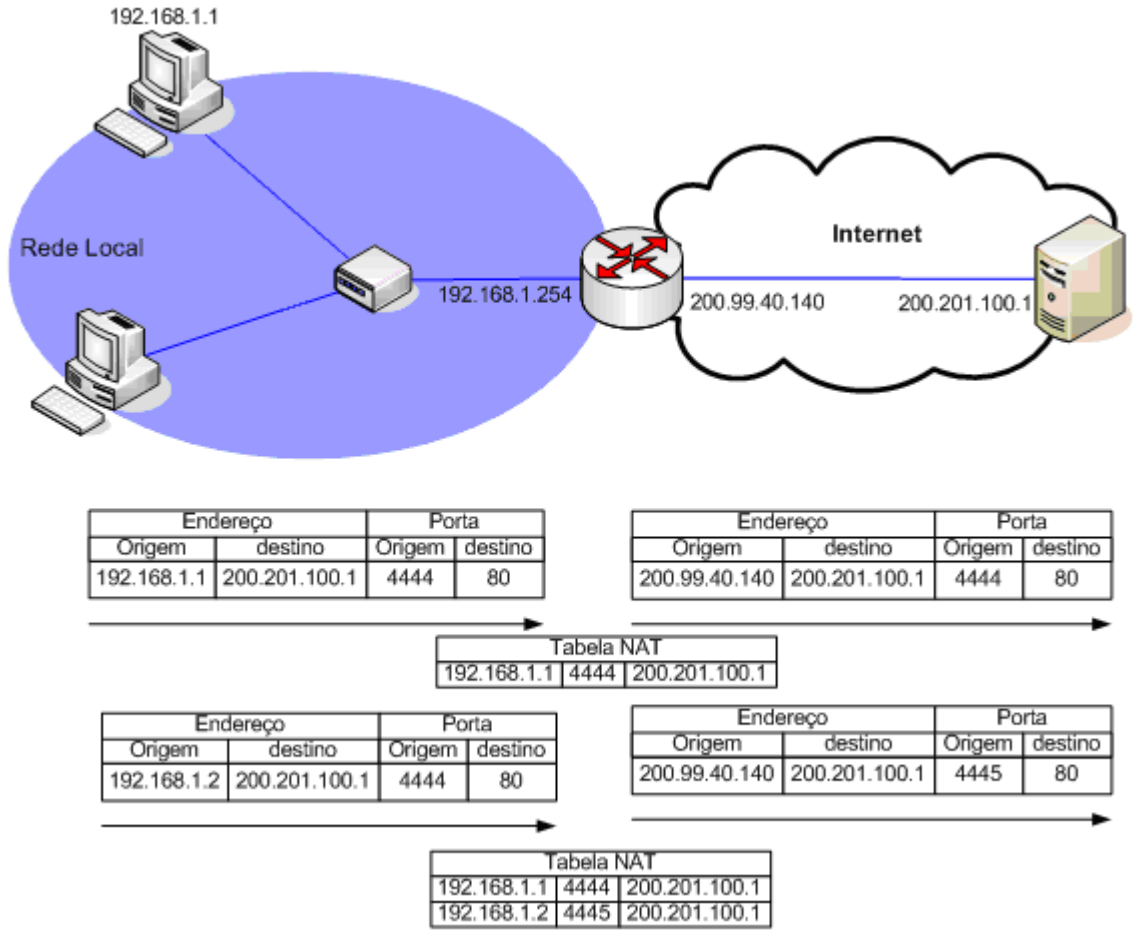
Ao realizar uma NAT para os endereços de uma rede local (Endereços Internos Locais) é necessário possuir ao menos um endereço público (Endereço Interno Global) que estará localizado no roteador que provê acesso à internet.

Este roteador irá alterar o conteúdo do cabeçalho do pacote trocando o endereço privado de origem pelo seu endereço publico. Este mapeamento é armazenado na tabela NAT e o pacote será encaminhado. Ao responder o *host* da internet irá endereçar o pacote ao endereço interno global, pois foi este quem o enviou. Ao receber a resposta o roteador saberá que esta é uma resposta para o *host* interno por meio do mapeamento existente na tabela NAT criada por ele.

A NAT não só torna desnecessária a utilização de endereços públicos para todos os sistemas que necessitam de acesso à internet, mas também provê segurança. Caso um *host* da internet tente se comunicar com um *host* da rede local esta comunicação será bloqueada, pois não existe na tabela NAT um registro dessa comunicação. Desta forma a NAT permite que somente sejam abertas conexões no sentido "rede local para Internet", impedindo ataques de *hackers*.

Como a NAT faz um mapeamento IP a IP (IP de origem a IP de destino) para que haja múltiplos acessos a um mesmo destino seriam necessários vários endereços. Para prover este serviço, sem que haja mapeamentos duplicados, a NAT utiliza uma multiplexação no nível das portas, isto é feito por meio da PAT (*Port Address Translation*).

Com o uso da PAT, os *hosts* internos podem compartilhar um único endereço IP público para toda comunicação externa. A PAT faz um mapeamento na tabela NAT mais detalhado utilizando IP de origem, IP de destino, e porta de origem e destino. Caso a porta de origem já esteja mapeada para outra origem, o roteador irá incrementar o número da porta do datagrama e realizar a tradução.



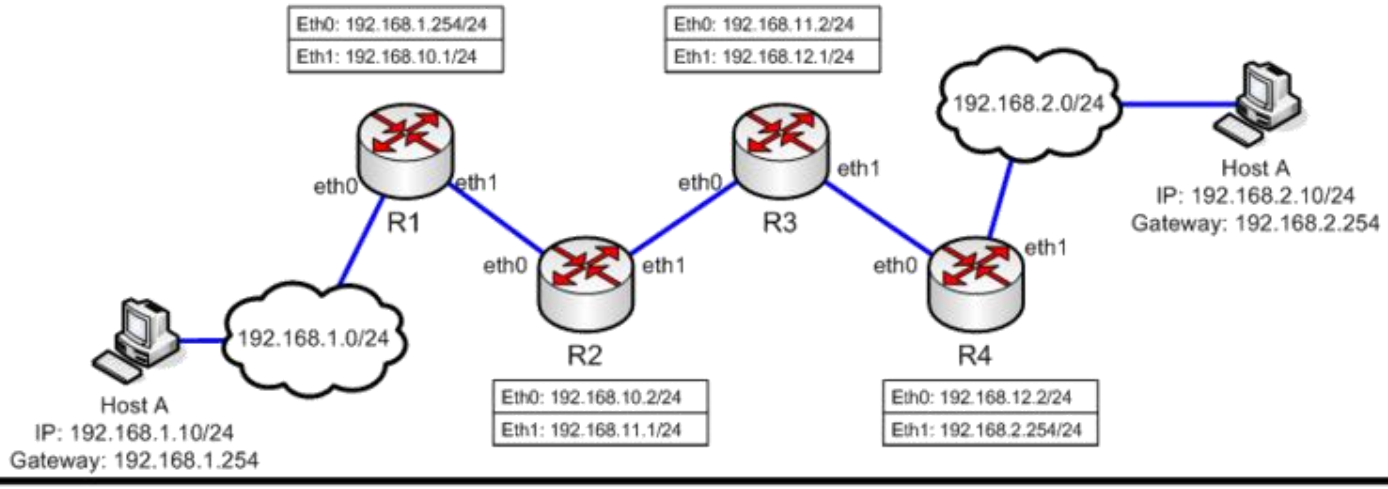
Apesar de todas as vantagens apresentadas pela NAT ela também possui desvantagens:

- Aumenta o atraso devido a tradução de cada endereço IP dentro dos cabeçalhos dos pacotes;
- Perda da rastreabilidade IP ponta-a-ponta. Torna-se muito mais difícil rastrear pacotes que passam por diversas alterações de endereço;
- Força alguns aplicativos que usam endereçamento IP a pararem de funcionar, porque oculta os endereços IP ponta-a-ponta.

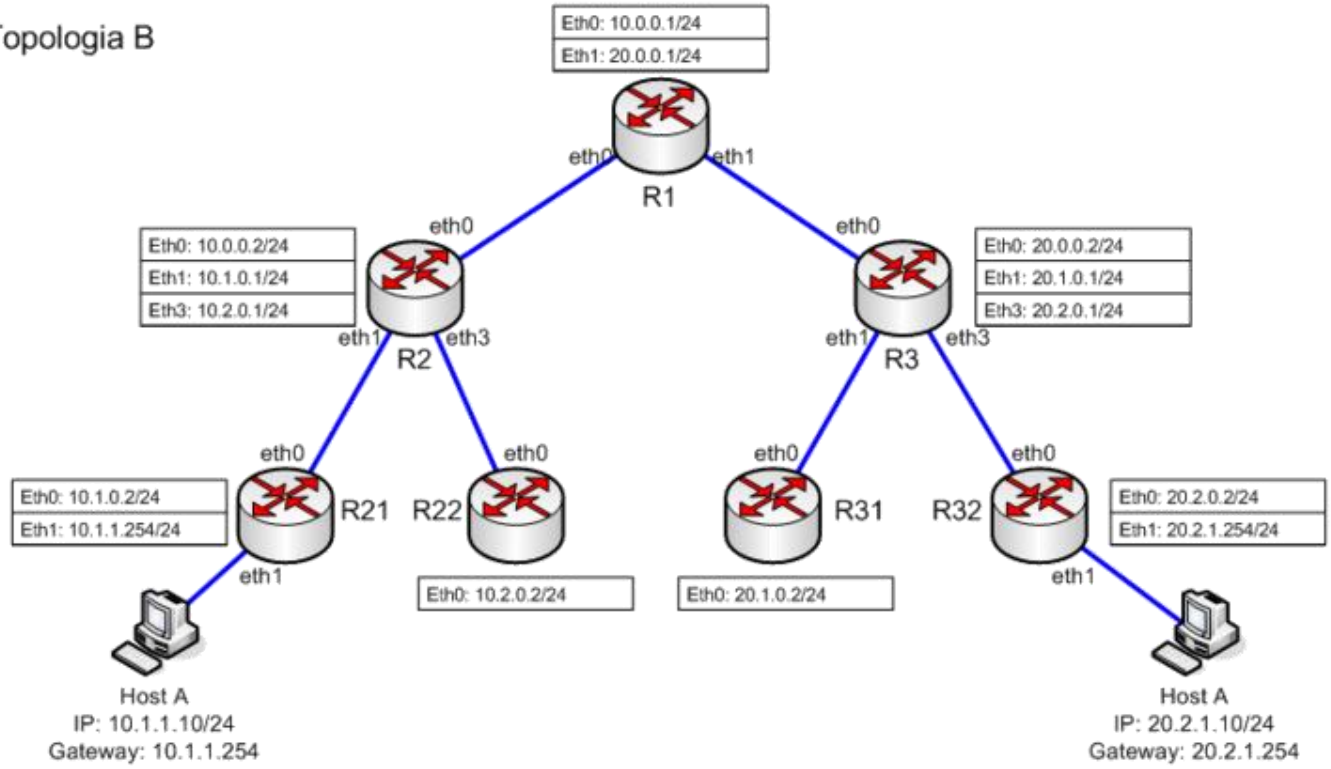
Exercícios Referente ao Capítulo Passado e Atual

1. Montar a tabela de roteamento, sem utilizar Default Gateway, dos roteadores nos seguintes ambientes:

Topologia A

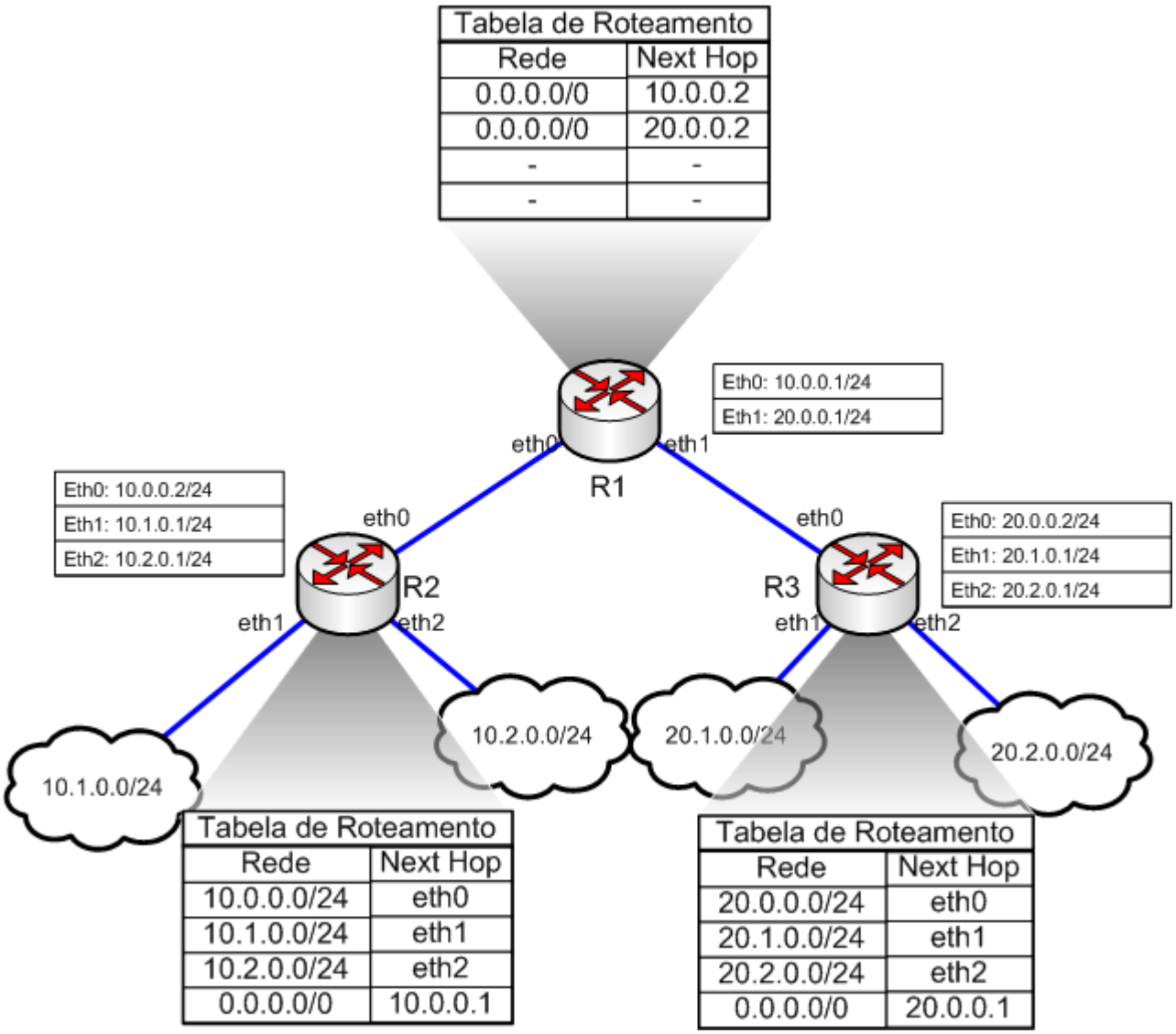


Topologia B



2. Montar a tabela de roteamento utilizando default gateway do exercício anterior.

3. Qual o erro na seguinte configuração:



4. Qual será o TTL do pacote enviado pelo Host A ao chegar no Host B nas topologias utilizadas no exercício 1?

5. Qual será o valor dos campos MAC de Origem, MAC de Destino, IP de Origem e IP de destino do pacote, em cada segmento de rede, durante a comunicação do Host A com o Host B nas topologias do exercício 1?

6. [Desafio] Projete uma rede local com acesso a internet com os seguintes requisitos:

- Utilizar 2 roteadores: Um somente para NAT (roteador A) e outro para interligar os segmentos de rede (roteador B);
- Ter 3 segmentos de redes distintos: Rede dos clientes, rede de servidores e rede de ligação (roteador A <-> roteador B);

Informando exemplos de configuração de hosts de cada rede (IP, máscara e gateway) e a configuração dos roteadores (endereço IP, máscara, default gateway e rotas).

Camada 3 - Camada de Rede (Parte 4)

Sub-redes

Como dito anteriormente, o uso de sub-redes é um método usado para gerenciar endereços IP dividindo classes inteiras de endereços de redes em pedaços menores. Este método impediu, temporariamente, o esgotamento completo

dos endereços IP. Com as sub-redes, a rede não fica limitada às máscaras de rede padrão de classes A, B ou C, e há maior flexibilidade no projeto da rede.

Para que usar subredes

Essa estória de dividir classes inteiras de endereços em pedaços menores eu já tenho feito aqui! Fiz sem avisar pra testar se alguém ia se tocar e falar que eu errei e também pra começar acostumar a todos com o conceito de sub-redes. Mas porque fazer isso?? Tem uma série de fatores, mas principalmente é feito para economizar endereços IPs. Você pode montar uma rede simples sem utilizar sub-redes, mas para redes grandes ou extremamente grandes, a divisão em sub-redes é necessária.

Como fazer suredes

Dividir uma rede em sub-redes significa usar a máscara de sub-rede para dividir a rede em segmentos menores, ou sub-redes, mais eficientes e mais fáceis de gerenciar. Gerando assim números maiores de redes pequenas. Como fazer isso?? Por exemplo, o IP 10.0.0.0/8. Se você for utilizar essa rede em uma rede local, voce vai ter "infinitos" endereços de hosts mas somente um endereço de rede! Se você precisar de uma rede voltada para servidores (serverfarm) você vai ter que usar outra rede, o que esse endereçamento não contempla! Então a solução é o uso de sub redes.

Antes de "reduzir" a rede 10.0.0.0/8, vamos analisar. Esse endereço provê uma rede (10.0.0.0) e inúmeros hosts (de 10.0.0.1 a 10.255.255.254). Como não precisamos de todos esses hosts vamos reduzir da seguinte forma: 10.0.0.0/16. Pronto, simples assim! Só mudei a máscara! Dessa forma tenho 255 redes! Quer que eu prove?! Vamos colocar o IP sobre a máscara:

-	Octeto 1	Octeto 2	Octeto 3	Octeto 4
Endereço	10	0	0	0
Mascara	255	255	0	0

Vamos converter pra binário:

Pra quem perdeu sobre numeração binária: Curso de Redes: Numeração Binária

-	Octeto 1	Octeto 2	Octeto 3	Octeto 4
Endereço	0000 1010	0000 0000	0000 0000	0000 0000
Mascara	1111 1111	1111 1111	0000 0000	0000 0000

Pra quem não o calcular do número de rede e host: Curso de Redes: Camada de Rede - Parte 1

Podemos ver que os 2 primeiros octetos se referem a rede e os dois últimos ao host. Dessa forma temos as seguintes características: Um endereçamento que provê 256 redes (de 10.0.0.0 até 10.255.0.0) e 65534 hosts por rede (de 10.0.0.1 a 10.0.255.254 ou de 10.1.0.1 até 10.1.255.255).

Muitos devem estar se perguntando: "porque não posso variar o primeiro octeto??" Porque a ideia é criar redes dentro da rede, e qual é a nossa rede?? 10.0.0.0/8. A máscara de 8 bits fixa o primeiro octeto. Fácil ne?? Se você parar pra

pensar, eu simplesmente pego emprestado alguns bits do endereço de host original e uso para endereçar a rede. Vamos analisar o IP 10.12.0.20

-	Octeto 1	Octeto 2	Octeto 3	Octeto 4
Endereço	**0000 1010**	**0000 1100**	**0000 0000**	**0001 0100**
Mascara	**1111 1111**	**1111 1111**	**0000 0000**	**0000 0000**

Assim podemos ver a porção de rede original (verde) a porção de rede pega emprestada (vermelha), da antiga parte de host, e a parte de host restante (preta).

Um exemplo prático

Agora vamos complicar um pouco! Vamos supor que o chefe quer implementar algumas redes para servidores: "Precisamos montar 4 redes para servidores! Cada rede tem que suportar até 10 servidores. Infelizmente só temos disponível para isso a rede 192.168.1.0/24.".

E agora?! Como implementar 4 redes utilizando a rede 192.168.1.0/24?? Com sub-redes! Vamos pensar em numeração binária. Quantos bits livre temos?? 8 bits, que é o octeto 4. Vamos utilizar esses 8 bits pra criar as redes.

Nossa rede tem que suportar até 10 servidores, considerando o gateway desses servidores, teremos que ter 11 endereços de Hosts nessa rede. Assim como em toda a computação, redes funciona sempre em potências de 2. Dessa forma não vamos conseguir prover exatamente 11 endereços de hosts. Qual o próximo multiplo de 2 mais próximo de 11?? Vamos lá:

Potência	Em termos de multiplicação	resultado
2^1	2	2
2^2	2x2	4
2^3	2x2x2	8
2^4	2x2x2x2	16
2^5	2x2x2x2x2	32
2^6	2x2x2x2x2x2	64
2^7	2x2x2x2x2x2x2	128
2^8	2x2x2x2x2x2x2x2	256

Ok!! A próxima potência de 2 é 16. Mas como sabemos, as redes tem 2 endereços reservados, os endereços de rede - com a porção do host preenchida com zeros - e o endereço de broadcast - com a porção de hosts preenchido com um. Dessa forma temos apenas 14 (16 - 2) endereços de hosts úteis.

Vamos pra segunda parte! Quantos bits são necessários para escrever 14? vamos converter: 14 em decimal = 1100 em binário. Utilizamos 4 bits. Como vimos, temos 8 bits "livres". Como precisamos de 4 para os hosts, nos sobram 4 para a rede (8 - 4 = 4). Então vamos ver na tabela!

-	Octeto 1	Octeto 2	Octeto 3	Octeto 4

-	Octeto 1	Octeto 2	Octeto 3	Octeto 4
Endereço	**1100 0000**	**1010 1000**	**0000 0001**	**0000 0000**
Máscara	**1111 1111**	**1111 1111**	**1111 1111**	**1111 0000**

Isso mesmo! Vamos pegar só uma parte do último octeto! Da mesma forma temos a parte fixa (verde), que nosso chefe mandou utilizar, a parte que pegamos emprestada (vermelho) e a parte de host (preto).

Agora começa a parte complicada. Qual a máscara dessa "nova rede"?? 255.255.255.240 (/28). Então vamos ver todas as possíveis subredes que teremos utilizando a máscara /28. Como já vimos anteriormente, os uns da máscara define a porção de rede. Variando os bits da porção de rede do endereço IP (bits em vermelho) temos as possíveis redes. Então vamos começar:

Octeto 1	Octeto 2	Octeto 3	Octeto 4	Notação decimal pontuada
1100 0000	**1010 1000**	**0000 0001**	**0000 0000**	192.168.1.0
1100 0000	**1010 1000**	**0000 0001**	**0001 0000**	192.168.1.16
1100 0000	**1010 1000**	**0000 0001**	**0010 0000**	192.168.1.32
1100 0000	**1010 1000**	**0000 0001**	**0011 0000**	192.168.1.48
1100 0000	**1010 1000**	**0000 0001**	**0100 0000**	192.168.1.64
1100 0000	**1010 1000**	**0000 0001**	**0101 0000**	192.168.1.80
1100 0000	**1010 1000**	**0000 0001**	**0110 0000**	192.168.1.96
1100 0000	**1010 1000**	**0000 0001**	**0111 0000**	192.168.1.112
1100 0000	**1010 1000**	**0000 0001**	**1000 0000**	192.168.1.128
1100 0000	**1010 1000**	**0000 0001**	**1001 0000**	192.168.1.144
1100 0000	**1010 1000**	**0000 0001**	**1010 0000**	192.168.1.160
1100 0000	**1010 1000**	**0000 0001**	**1011 0000**	192.168.1.176
1100 0000	**1010 1000**	**0000 0001**	**1100 0000**	192.168.1.192
1100 0000	**1010 1000**	**0000 0001**	**1101 0000**	192.168.1.208
1100 0000	**1010 1000**	**0000 0001**	**1110 0000**	192.168.1.225
1100 0000	**1010 1000**	**0000 0001**	**1111 0000**	192.168.1.240

Sim, por mais estranho que isso possa parecer, isso são endereços de rede! Tem uma "manha" pra calcular essas coisas. É só pegar a primeira rede (não zero), nesse caso a 192.168.1.16 e ir incrementando com o último número (16). Então teremos 16+16=32 (192.168.1.32), 32+16=48 (192.168.1.48), 48+16=64 (192.168.1.64) e etc...

Agora vamos sortear um endereço de rede e calcular os endereços de hosts. Hum... Vamos pegar a rede 192.168.1.80/28.

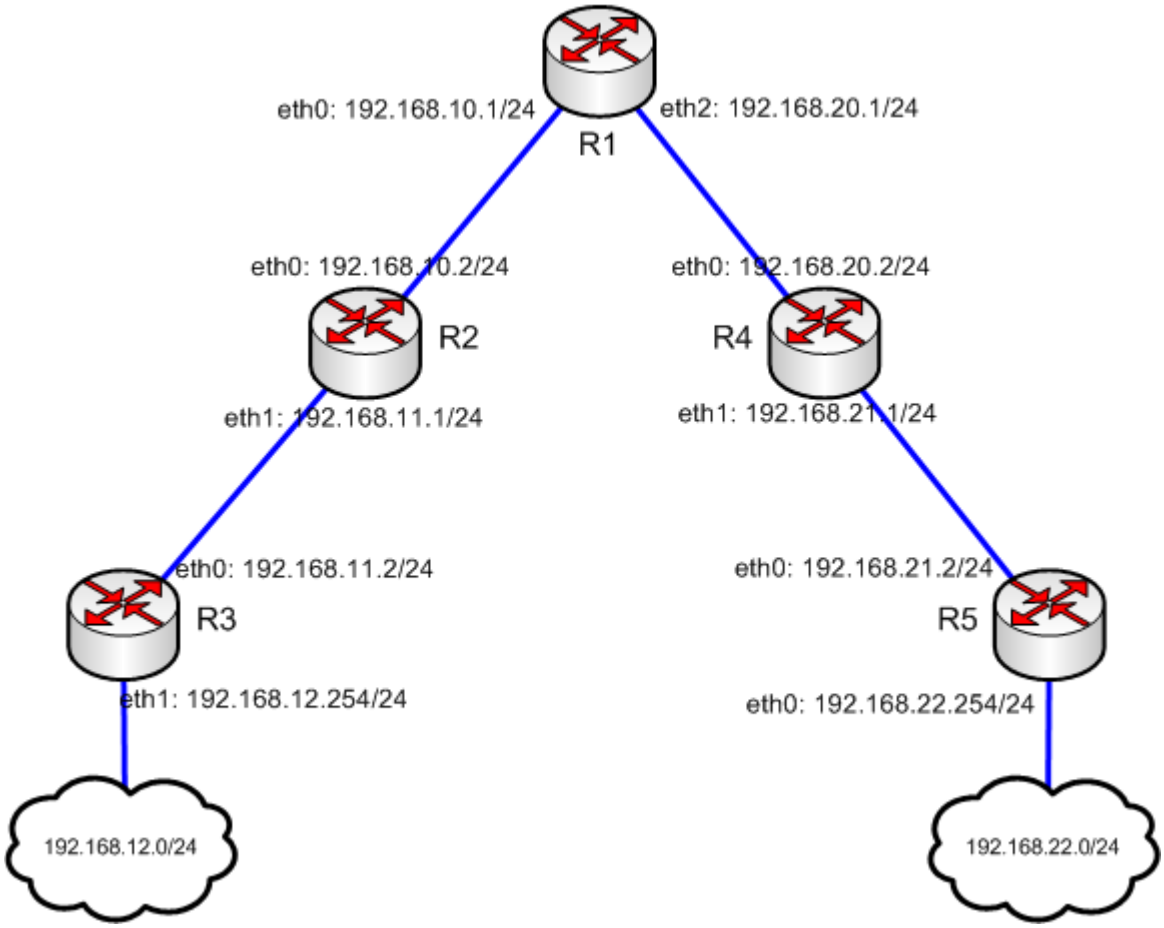
Octeto 1	Octeto 2	Octeto 3	Octeto 4	Notação decimal pontuada
1100 0000	**1010 1000**	**0000 0001**	**0101 0000**	192.168.1.80
1100 0000	**1010 1000**	**0000 0001**	**0101 0001**	192.168.1.81
1100 0000	**1010 1000**	**0000 0001**	**0101 0010**	192.168.1.82
1100 0000	**1010 1000**	**0000 0001**	**0101 0011**	192.168.1.83
1100 0000	**1010 1000**	**0000 0001**	**0101 0100**	192.168.1.84
1100 0000	**1010 1000**	**0000 0001**	**0101 0101**	192.168.1.85
1100 0000	**1010 1000**	**0000 0001**	**0101 0110**	192.168.1.86
1100 0000	**1010 1000**	**0000 0001**	**0101 0111**	192.168.1.87
1100 0000	**1010 1000**	**0000 0001**	**0101 1000**	192.168.1.88
1100 0000	**1010 1000**	**0000 0001**	**0101 1001**	192.168.1.89
1100 0000	**1010 1000**	**0000 0001**	**0101 1010**	192.168.1.90
1100 0000	**1010 1000**	**0000 0001**	**0101 1011**	192.168.1.91
1100 0000	**1010 1000**	**0000 0001**	**0101 1100**	192.168.1.92
1100 0000	**1010 1000**	**0000 0001**	**0101 1101**	192.168.1.93
1100 0000	**1010 1000**	**0000 0001**	**0101 1110**	192.168.1.94
1100 0000	**1010 1000**	**0000 0001**	**0101 1111**	192.168.1.95

Vale relembrar que o primeiro endereço de host (com todos os bits da porção de hosts como zero) é o endereço de rede (192.168.1.80) e o último endereço (com todos os bits da porção de hosts como uns) é o endereço de broadcast (192.168.1.95). Você pode ver que o endereço de broadcast é exatamente o endereço da próxima sub-rede (192.168.1.96) subtraído de um. Essa "manha" é útil para quando você precisa saber rápidamente o endereço de broadcast de uma rede. Observe também que o endereço de rede sempre é par e o endereço de broadcast sempre é impar. No caso do cálculo dos endereços de hosts é mais simples pois só precisamos incrementar com o número 1.

Pronto, missão cumprida! Temos agora 16 redes e cada uma com apenas 14 hosts. Atende a solicitação do chefe e garantimos nosso emprego!

Outras vantagens de subredes

As sub-redes são muito úteis também para reduzir o desperdício de redes. Em uma rede grande é normal ter links não populados entre roteadores. Ex:



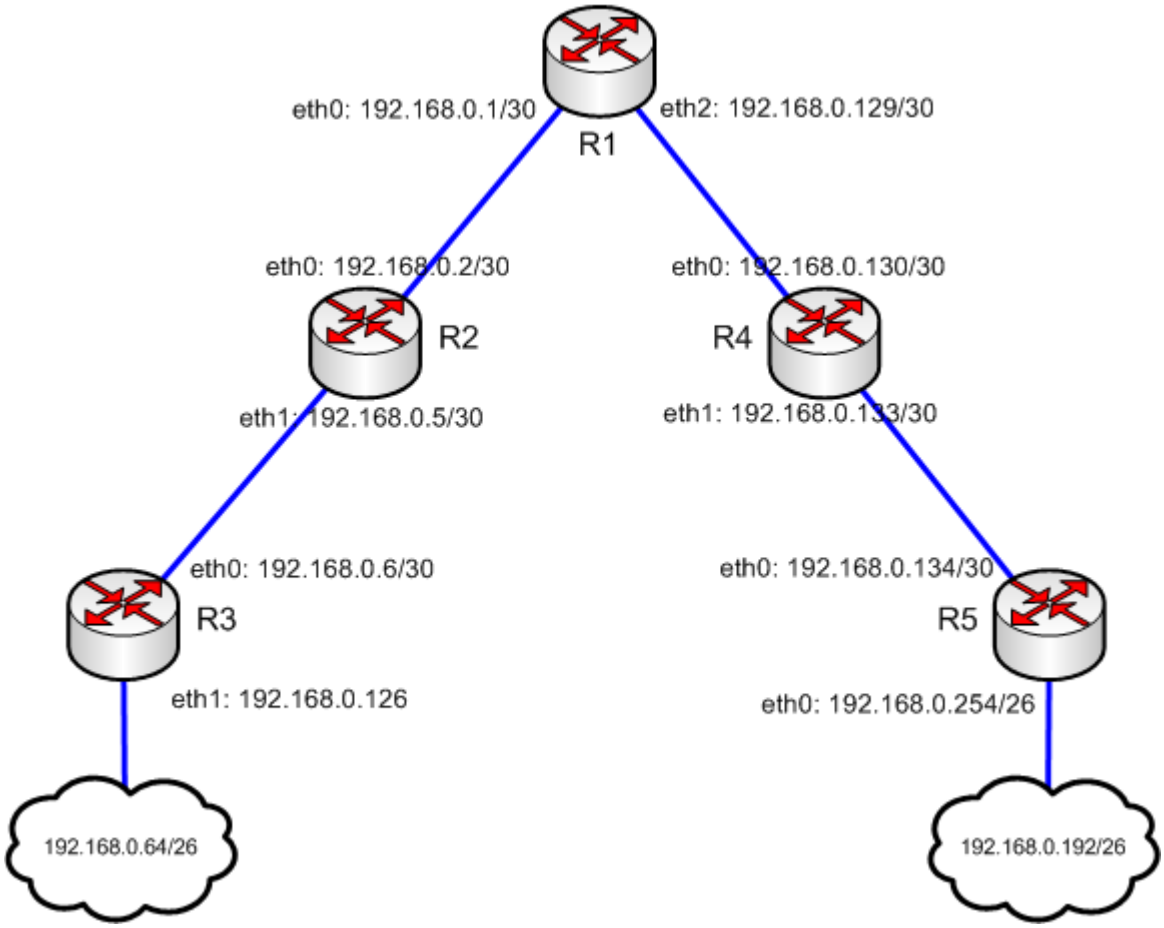
Se não utilizarmos subredes designaríamos uma rede de 254 hosts para conectar 2 roteadores, o que é um desperdício extremo pois precisamos de apenas 2 endereços de hosts. Ai entra a subrede. Se utilizarmos um IP com máscara /30 ou 255.255.255.252 teremos uma rede de apenas 2 hosts. Vamos ver um exemplo prático. Vamos pegar o IP 172.16.32.0/30. Vamos observar conforme feito anteriormente:

-	Octeto 1	Octeto 2	Octeto 3	Octeto 4
Endereço	1010 1100	0001 0000	0010 0000	0000 0000
Máscara	1111 1111	1111 1111	1111 1111	1111 1100

Se variarmos o campo de host (em preto) teremos os seguintes valores possíveis:

Octeto 1	Octeto 2	Octeto 3	Octeto 4	Notação decimal pontuada
1010 1100	0001 0000	0010 0000	0000 0000	172.16.32.0
1010 1100	0001 0000	0010 0000	0000 0001	172.16.32.1
1010 1100	0001 0000	0010 0000	0000 0010	172.16.32.2
1010 1100	0001 0000	0010 0000	0000 0011	172.16.32.3

Podemos ver que o endereços de rede é 172.16.32.0, o de broadcast é o 172.16.32.3 e os únicos IPs de hosts válidos é o 172.16.32.1 e o 172.16.32.2. Se dispormos de todo o bloco 172.16.32.0/24 podemos ter 64 redes. Por exemplo a rede 172.16.32.4, com os hosts 172.16.32.5 e 172.16.32.6 e broadcast com 172.16.32.7. Vamos rever a topologia anterior utilizando subredes /30:

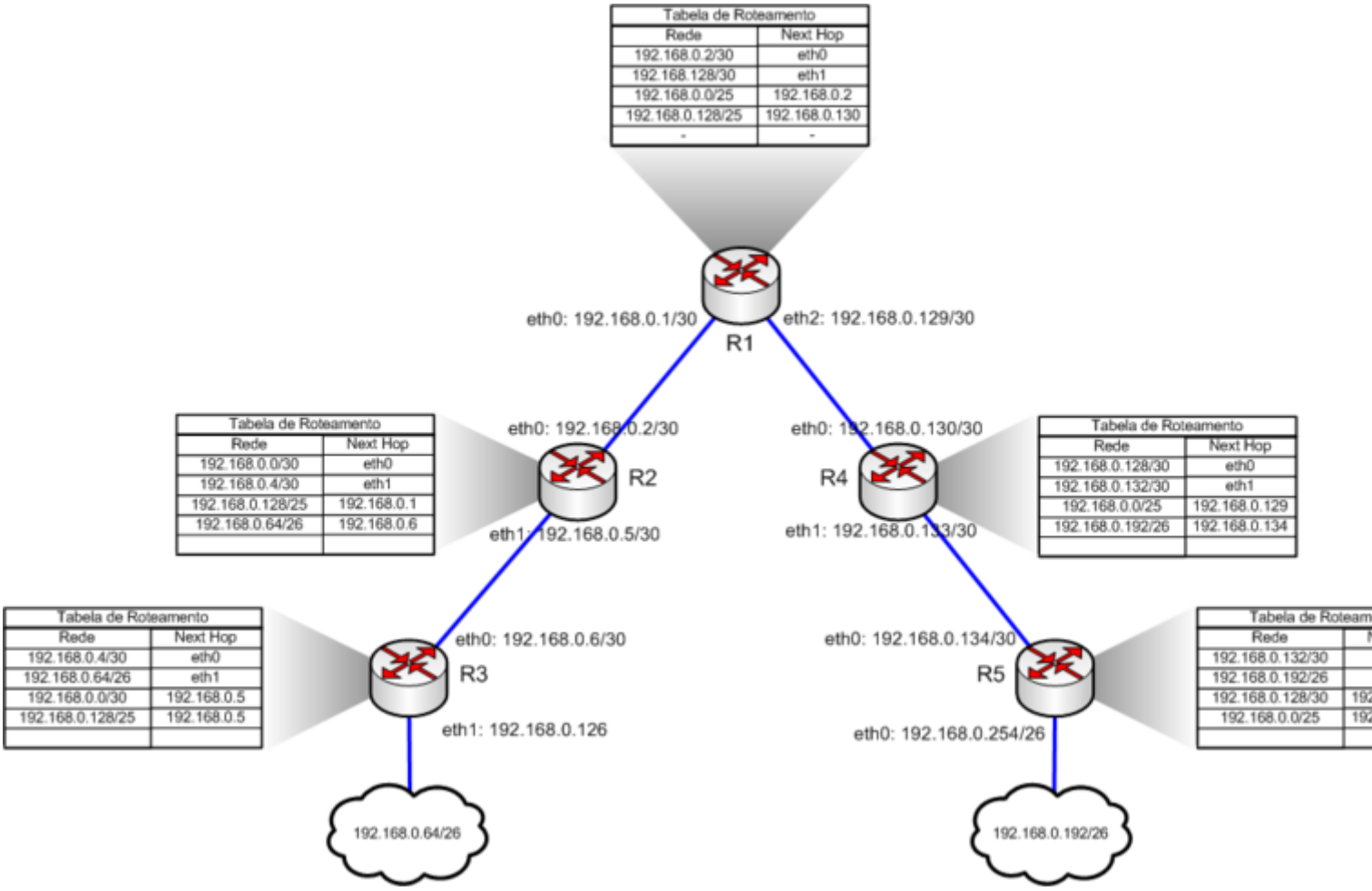


Como podemos ver temos uma economia enorme de endereços IPs. Isso é muito útil também para redes wireless, cone temos muitos rádios ligados ponto-a-ponto, ou até mesmo em redes ponto-a-multiponto que não tendem a crescer, desde que utilizamos uma máscara mais aberta, /28 por exemplo.

Sumarização de rotas

A sumarização de rotas é a "abreviação" de rotas. Ela serve para manter uma tabela de roteamento mais limpa. Tomando a imagem do exemplo anterior, podemos ver que cada roteador terá pelo menos 6 rotas (incluindo as redes diretamente conectadas).

Quanto mais a rede cresce mais complexa será a tabela de roteamento. Se utilizarmos a sumazação de rotas podemos ter tabelas mais consistentes e simples:



Mas como fazemos essa sumarização? Nos temos que projetar a rede de forma a obte-la, como a imagem acima. Para configurar a sumarização temos que pegar a tabela de roteamento de um roteador e analisar as rotas. Por exemplo, o R1. Sua tabela de roteamento seria a seguinte:

Destino	Próximo salto
192.168.0.0/30	eth0
192.168.0.128/30	eth1
192.168.0.4/30	192.168.0.2
192.168.0.64/26	192.168.0.2
192.168.0.132	192.168.0.130
192.168.0.192	192.168.0.130

Vamos "agrupar" as rede 192.168.0.0/30, 192.168.0.4/30 e 192.168.64/26. Esse agrupamento só pode ser feito quando ambos possuem o **mesmo gateway**. Vamos converter para binário:

Endereço	Octeto 1	Octeto 2	Octeto 3	Octeto 4
192.168.0.0	**1100 0000**	**1010 1000**	**0000 0000**	**0000 0000**

Endereço	Octeto 1	Octeto 2	Octeto 3	Octeto 4
192.168.0.4	**1100 0000**	**1010 1000**	**0000 0000**	**0000 0100**
192.168.0.64	**1100 0000**	**1010 1000**	**0000 0000**	**0100 0000**

Até onde esses 3 endereços IPs são iguais? Exatamente, todos os bits em verde (**25** primeiros bits) são comuns aos 3 endereços IPs, logo podemos sumarizar essas 3 redes em uma única rede: 192.168.0.0/**25**. Magicamente 3 rotas viram uma!

Fazemos o mesmo com os IPs 192.168.0.128, 192.168.0.132 e 192.168.0.192:

Endereço	Octeto 1	Octeto 2	Octeto 3	Octeto 4
192.168.0.128	**1100 0000**	**1010 1000**	**0000 0000**	**1000 0000**
192.168.0.132	**1100 0000**	**1010 1000**	**0000 0000**	**1000 0100**
192.168.0.192	**1100 0000**	**1010 1000**	**0000 0000**	**1100 0000**

Podemos ver que novamente os 25 primeiro bits são iguais. logo podemos resumir essas tres redes em uma: 192.168.0.128/25.

Se repitirmos esse mesmo processo para os outros hosts veremos que será possivel sumarizar as tabelas. Mas temos que ter muita atenção pois só podemos sumarizar rotas que possuem o mesmo gateway.

Muitos roteadores fazem a sumarização de rotas automaticamente, mas isso depende do fabricando e versão do firmware.

Acabou que nesse exemplo eu utilizei outro recurso chamado VLSM (Variable length subnet mask). O padrão de sub-redes é utilizar a a mesma mascara ao longo da sub-rede. Mas às vezes é necesário variar a máscara para ter uma rede com mais hosts como era o caso. O VSLM é um artifício que ajuda muito pra criar uma rede sumarizada.

Exercícios

1. Calcular o endereço de rede e broadcast dos seguintes endereços IPs:
1. 192.168.1.35/28
2. 172.16.193.0/10
3. 192.168.2.127/25

2. Divida em subredes a rede 192.168.1.0/24 de forma a obter 64 hosts por sub-rede.

3. Sumarize a seguinte tabela de roteamento:

Destino	Próximo salto
192.168.0.0/30	172.16.32.1
192.168.0.4/30	172.16.32.1
192.168.0.8/30	172.16.32.2
192.168.0.12/30	172.16.32.1

Destino	Próximo salto
192.168.0.128/30	172.16.32.1
10.0.0.0/8	172.16.32.1
172.16.10.0/16	172.16.32.2

Camada 4 - Camada de Transporte (Parte 1)

A função básica da camada de transporte é aceitar dados da camada de aplicação, dividi-los em unidades menores em caso de necessidade, passá-los para a camada de rede e garantir que todas essas unidades cheguem corretamente à outra extremidade. Além disso, tudo tem de ser feito com eficiência de forma que as camadas superiores fiquem isoladas das inevitáveis mudanças na tecnologia de hardware.

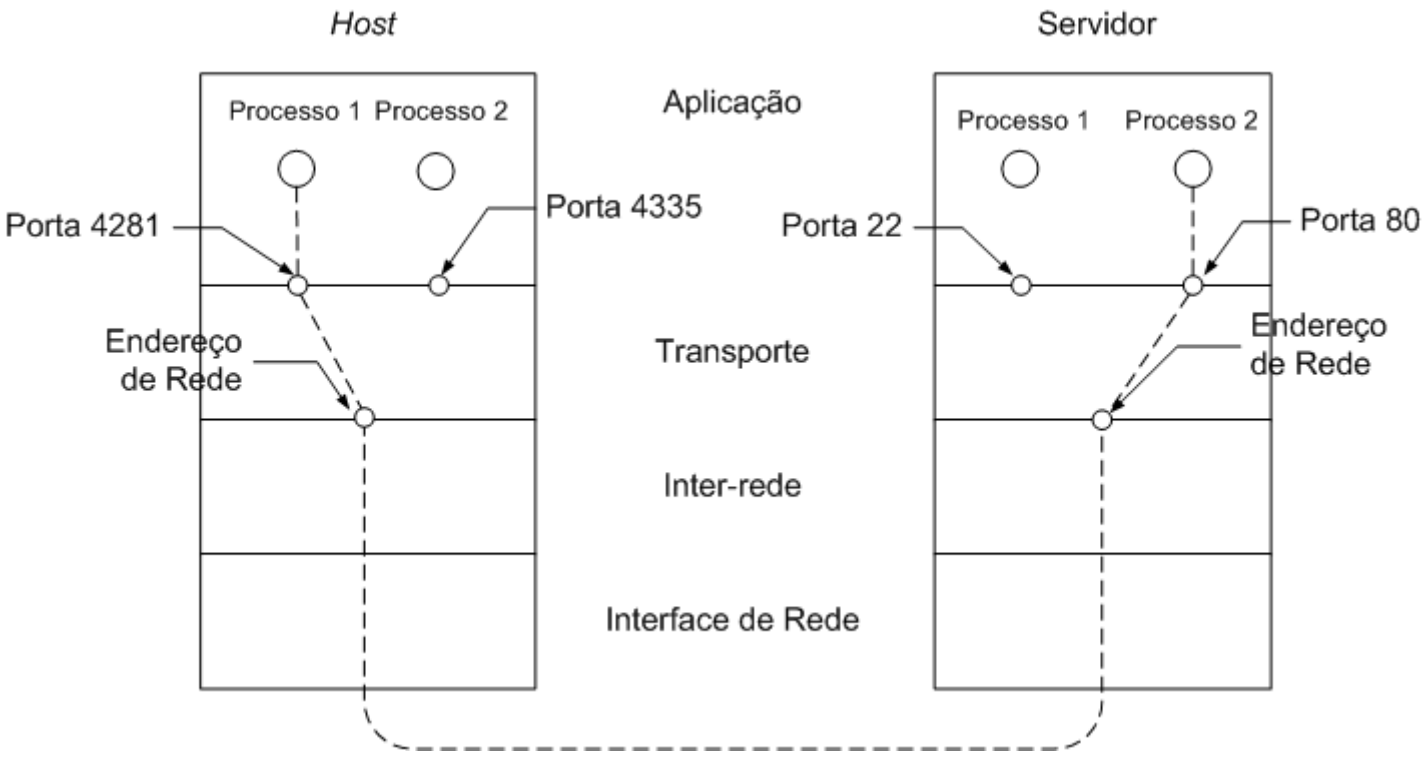
A camada de transporte é uma camada fim a fim, que liga a origem ao destino. Em outras palavras, um programa da máquina de origem mantém uma conversa com um programa semelhante instalado na máquina de destino, utilizando cabeçalhos de mensagem e mensagens de controle. Entre as camadas de transporte de diferentes hosts são trocadas TPDUs (Transport Protocol Data Units) chamados de segmentos. Um segmento é composto pelo cabeçalho da camada de transporte e os dados da camada de aplicação.

Muitos hosts são multiprogramados; isso significa que muitas conexões estarão entrando e saindo de cada host. Desta forma é responsabilidade da camada de transporte multiplexar todas as comunicações em um único canal determinando a qual conexão uma mensagem pertence. Além da multiplexação é responsabilidade desta camada estabelecer conexões, encerrá-las e controlá-las de forma que um host muito rápido não possa sobrecarregar um host muito lento (controle de fluxo). Em redes IP são utilizados dois protocolos para a implementação destas funções: o TCP e o UDP.

Endereçamento da Camada de Transporte

Da mesma forma que em outras camadas, a camada de transporte também possui um endereçamento. Quando um processo de aplicação deseja estabelecer uma conexão com um processo de aplicação remoto, é necessário especificar a aplicação com a qual ele irá se conectar. O método utilizado é definir os endereços de transporte que os processos podem ouvir para receber solicitações de conexão.

Os processos utilizam os TSAP (Transport Service Acess Point – Ponto de Acesso de Serviços de Transporte) para se intercomunicarem. Em redes IP, o TSAP é um número de 16 bits chamado de porta. O endereço da camada de transporte é um número de 48 bits, composto pela agregação do endereço IP do host e o número da porta. Os serviços da camada de transporte são obtidos através da comunicação entre os sockets do transmissor e do receptor.



Para uma melhor organização de serviços, algumas portas foram definidas pela IANA como "portas bem conhecidas" (well-known ports). Estas são as portas abaixo de 1024, para aplicações não padronizadas são utilizadas portas acima deste valor.

Os sockets são diferentes para cada protocolo de transporte, desta forma mesmo que um socket TCP possua o mesmo número que um socket UDP, ambos são responsáveis por aplicações diferentes. Os sockets de origem e destino são responsáveis pela identificação única da comunicação. Desta forma é possível a implementação da função conhecida como multiplexação. A multiplexação possibilita que haja várias conexões partindo de um único host ou terminando em um mesmo servidor.

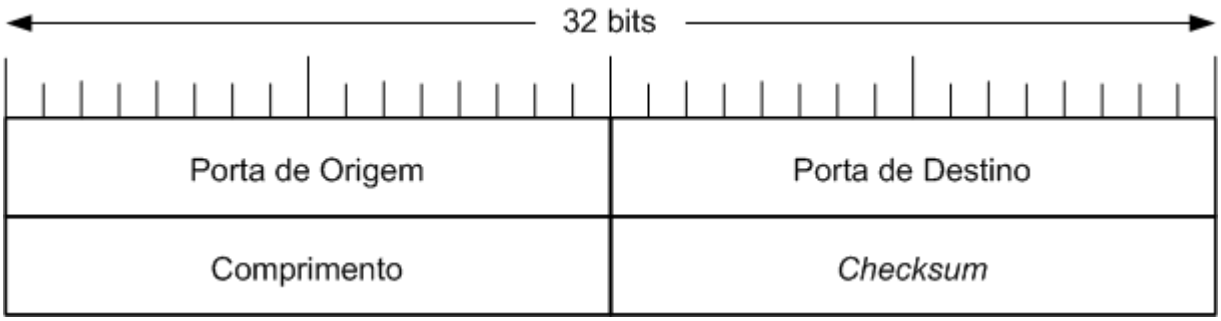
A formação do socket de origem e destino se dá da seguinte forma:

1. Ao iniciar uma comunicação é especificado para a aplicação o endereço IP de destino e a porta de destino;
2. A porta de origem é atribuída dinamicamente pela camada de transporte. Ele geralmente é um número seqüencial randômico acima de 1024;
3. O endereço IP de origem é atribuído pela camada 3.

User Datagram Protocol (UDP)

O UDP (Protocolo de Datagrama de Usuário) é um protocolo sem conexão não confiável para aplicações que não necessitam nem de controle de fluxo nem da manutenção da seqüência das mensagens enviadas. Ele é amplamente usado em aplicações em que a entrega imediata é mais importante do que a entrega precisa, como a transmissão dedados de voz ou vídeo. O UDP foi definido na RFC 768

O protocolo UDP é muito mais simples que o TCP, isto se deve ao fato dele não necessitar do estabelecimento de uma conexão (sinalização), controle de fluxo, controle de erros, retransmissão e sequênciamento dos dados. Todas essas funcionalidades são deixadas a cargo da aplicação desenvolvida. Devido a esta simplicidade seu cabeçalho possui apenas 8 bytes, composto por:

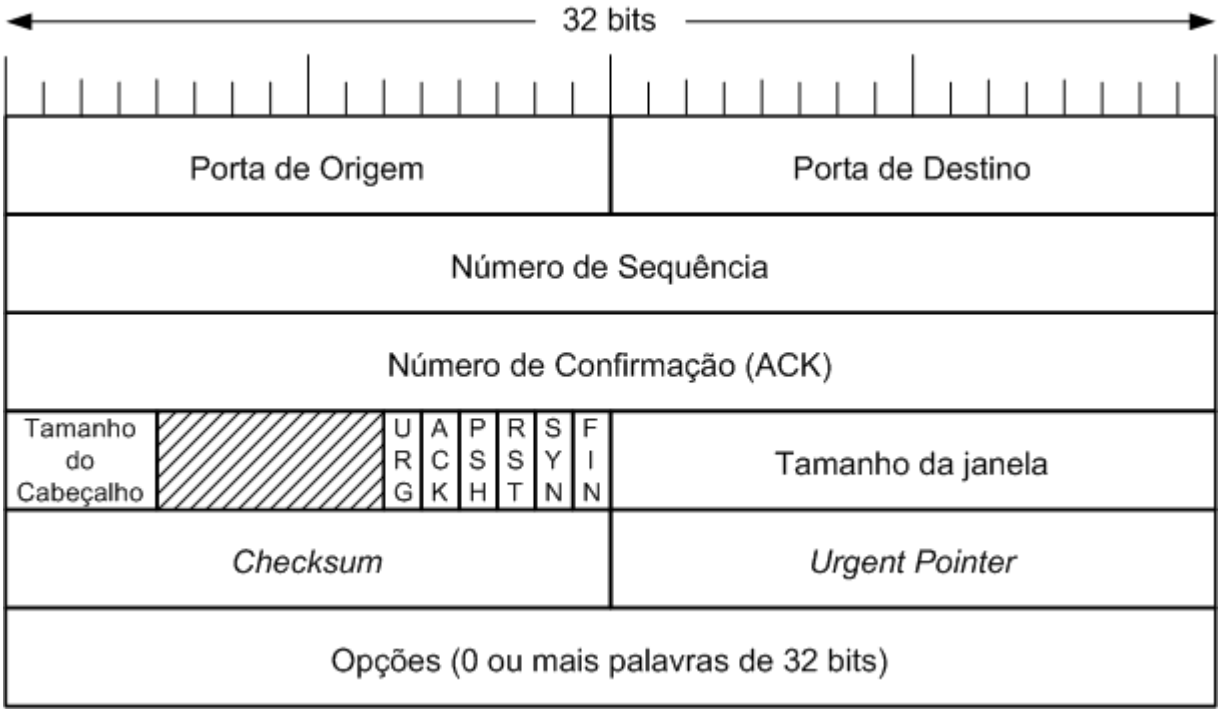


- Porta de Origem e Porta de Destino – Indicam os pares de porta que estão executando a comunicação;
- Comprimento – Indica o comprimento de todo o datagrama isto é, cabeçalho e dados;
- Checksum – Verificação de integridade do datagrama;

Transmission Control Protocol (TCP)

O TCP (Protocolo de Controle de Transmissão) foi projetado para oferecer um fluxo de bytes fim a fim confiável em uma inter-rede não confiável. Ele é um protocolo orientado a conexão que permite a entrega sem erros de um fluxo de bytes originado de uma determinada máquina para qualquer computados da rede. Esse protocolo fragmenta o fluxo de entrada em mensagens e passa cada uma delas para a camada de redes. No destino, o processo TCP remonta as mensagens recebidas gerando o fluxo de saída. O TCP foi projetado para se adaptar dinamicamente às propriedades da camada de rede e ser robusto diante dos muitos tipos de falhas que podem ocorrer.

O TCP foi formalmente definido na RFC 793, posteriormente alguns erros foram corrigidos e o TCP foi definido na RFC 1122. O protocolo TCP define um cabeçalho para suas mensagens composto dos seguintes campos:



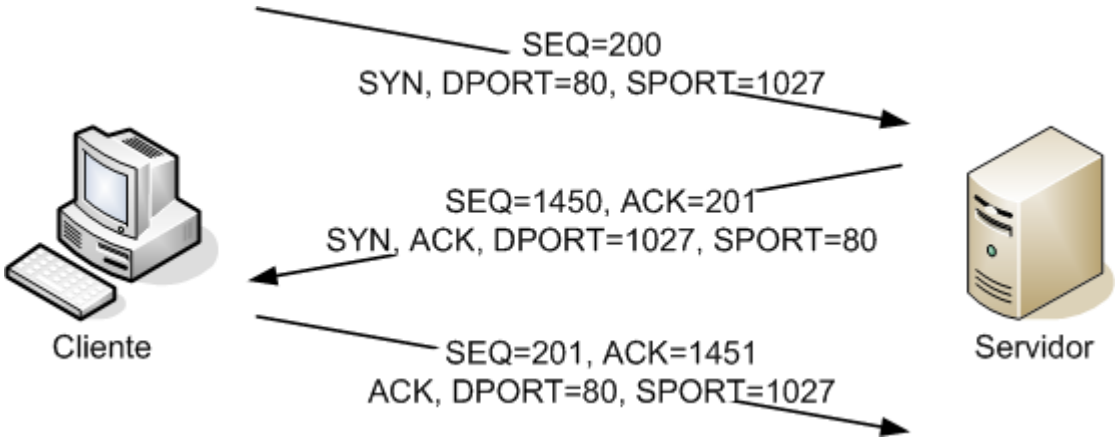
Cabeçalho TCP

- **Porta de Origem e Porta de Destino** – identifica os pontos terminais locais da conexão;
- **Número de Seqüência –** Identifica o fragmento dentro de todo o fluxo gerado;
- **Numero de Confirmação –** Indica qual o próximo byte esperado;
- **Tamanho do Cabeçalho** – Informa quantas palavras de 32 bits compõem o cabeçalho TCP;
- **URG** – Indica a utilização do *urgent pointer*;

- **ACK** – É utilizado para indicar que este segmento é um ACK e que o campo Número de Confirmação deve ser interpretado;
- **PSH** – Indica que este segmento não deve ser enfileirado como todos os outros, mas sim posto à frente na fila;
- **RST** – É utilizado para reiniciar uma conexão que tenha ficado confusa devido a falhas no *host* ou por qualquer outra razão;
- **SYN** – Este bit é utilizado para indicar um pedido de conexão e a confirmação da conexão;
- **FIN –** Utilizado para indicar que o emissor não possui mais dados para enviar e deseja finalizar a conexão;
- **Tamanho da Janela** – Indica quantos bytes podem ser enviados a partir do byte confirmado. Este campo é utilizado no controle de fluxo do TCP;
- ***Checksum*** – Indicador de integridade do segmento;
- ***Urgent Pointer*** – Indica um deslocamento de bytes a partir do número de seqüência atual em que os dados urgentes devem ser encontrados;
- **Opções** – Projetado para que o TCP possa oferecer recursos extras que não foram previstos em seu protocolo;

Estabelecimento de Conexões

O estabelecimento de uma conexão TCP ocorre antes que qualquer outro recurso TCP possa começar seu trabalho. O estabelecimento da conexão se fundamenta no processo de inicialização dos campos referentes à seqüência, aos ACKs e na troca dos números de *sockets* usados. As conexões são estabelecidas no TCP por meio do *three way handshake*(*handshake* de três vias).



Three Way Handshake

O estabelecimento da conexão é feito usando dois bits na cabeçalho TCP: SYN e ACK. Um segmento que possua a *flag*SYN ativa sinaliza uma requisição de sincronia do número de seqüência. Essa sincronização é necessária em ambos os sentidos, pois origem e destino utilizam números de seqüência distintos. Cada pedido de conexão é seguido de uma confirmação utilizando o bit ACK. O segundo segmento do *three way handshake* exerce as duas funções ao mesmo tempo: Confirma a sincronização do servidor com o cliente e requisita a sincronização do cliente com o servidor.

É interessante aqui analisar a imagem acima. Basicamente o Three way handshake 'simula' um acordo. O Cliente pergunta pro servidor: "Você está ai?" e servidor responde: "Sim estou...". Depois o servidor pergunta: "Você está ai?" e o cliente responde: "Sim estou...". Mas perai! Como tem 4 sentenças em apenas 3 trocas de mensagens?? Simples, a segunda mensagem contém uma resposta e uma pergunta. Como podemos verificar isso?? Através das Flags:

Cliente: *Servidor, você está ai?? (SYN)*
Servidor: *Sim estou... (ACK) E você, está ai? (SYN)*
Cliente: *Sim, estou... (ACK)*

Deu pra entender ne?! Mas porque o servidor precisa perguntar se o cliente está lá?? Pela simples necessidade de sincronização do número de seqüência. O número de seqüência é utilizado para garantir a entrega de todas as mensagens. Vamos imaginar da seguinte forma:

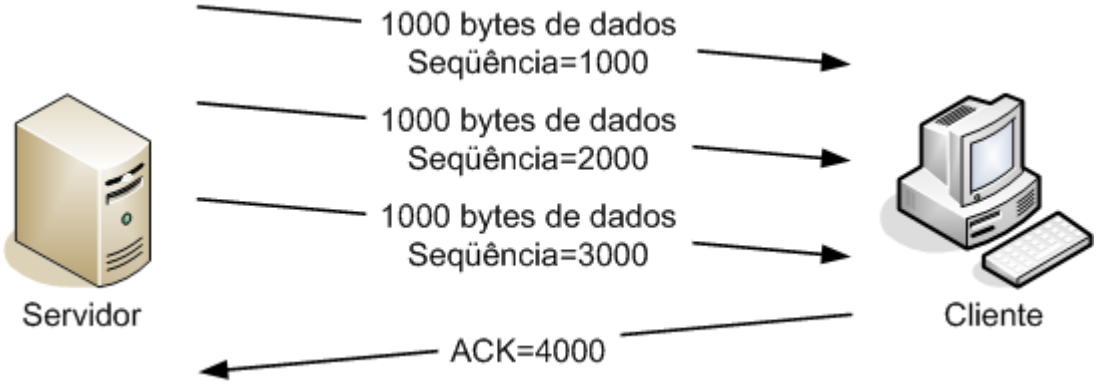
Cliente: *Cambio servidor, mensagem 200 (Numero de seqüência do cliente), o senhor está disponível (SYN)?*
Servidor: *Positivo cliente! Mensagem 1450 (Numero de seqüência do servidor) Prossiga com a mensagem 2001 (ACK=201), cambio.*
Cliente: *Positivo servidor! Mensagem 201 (numero de sequencia), confirmando número da próxima mensagem: 1451, cambio!*

Dessa forma eles trocam o número de seqüência, que tem como função "enumerar" as mensagens de cada um. Por exemplo, se a última mensagem foi a 201 e a mensagem que chegou pro servidor foi a 203, ele tem completa certeza que uma mensagem (202) se perdeu no caminho! Então basta somente solicitar uma retransmissão. O número de seqüência nem sempre é incrementado por 1, ele pode ser incrementado com base no número de bytes enviados pela origem.

O ACK tem como objetivo solicitar a continuidade das mensagens. Podemos interpretar um ACK=210 como sendo: "Pronto, recebi até a 209, pode mandar a 210". Isso vai ser demonstrado com mais calma para frente.

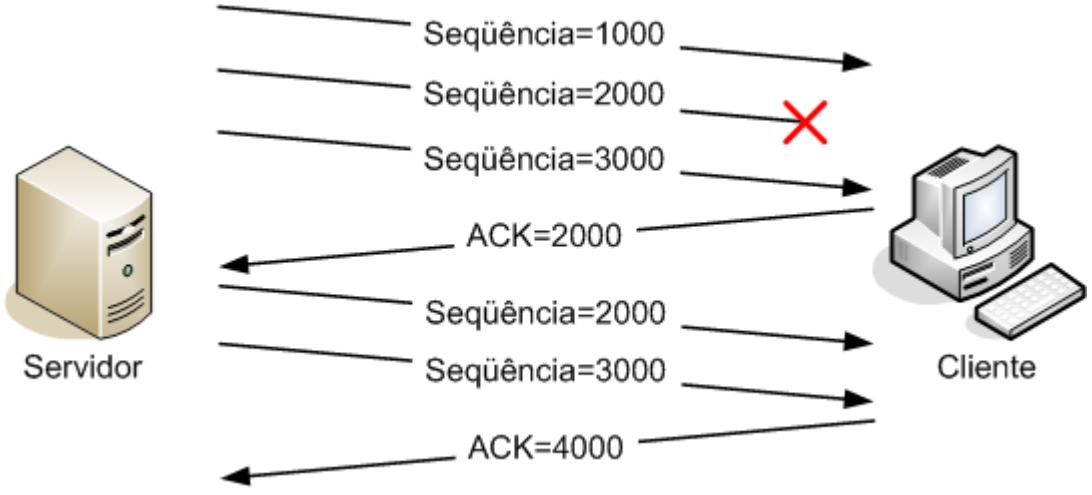
Recuperação de Erros

O TCP também proporciona uma transferência confiável de dados, o que também é chamado de confiabilidade ou recuperação de erros. Para conseguir a confiabilidade o TCP enumera os bytes de dados usando os campos referentes à seqüência e aos ACKs no cabeçalho TCP. O TCP alcança a confiabilidade em ambas as direções, usando um campo referente ao número de seqüência de uma direção, combinado com o campo referente ao ACK na direção oposta.



ACKs TCP sem erros

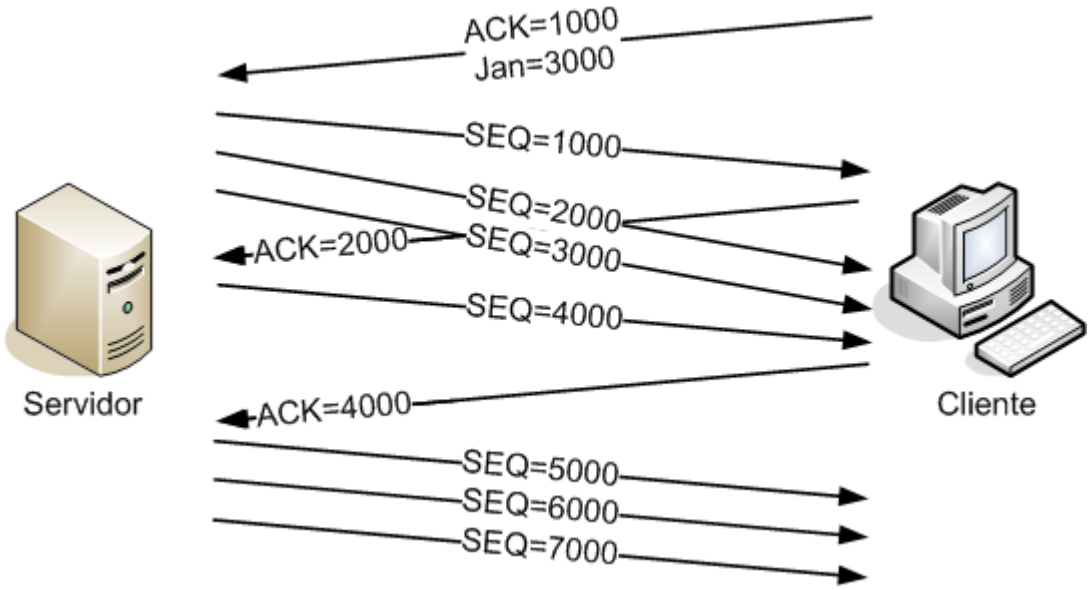
Como dito anteriormente o campo ACK implica o próximo byte a ser recebido. O número de seqüência indica o número do primeiro byte do segmento correspondente à sua posição no fluxo de dados



ACKs TCP com erros

Controle de Fluxo

O TCP implementa o controle de fluxo utilizando os dados dos campos Número de Seqüência, Número de Confirmação e Tamanha da Janela. O controle de fluxo no TCP pode utilizar uma janela de tamanho fixo ou uma janela deslizante.
O campo Tamanho da Janela indica, em tempo real, o número máximo de bytes sem confirmação que podem ser enviados. Com a utilização de janelas um emissor só poderá enviar o número de bytes, previsto na janela, antes de receber alguma confirmação.



Comunicação com janela fixa

Podemos interpretar a imagem a cima da seguinte forma:

O Cliente fala pro servidor: Olha servidor, to meio ocupado mas quero continuar esse donwload. Então não me envia mais que 3000 (tamanho da janela) bytes não confirmados OK??

O servidor enviar 3 pacotes cada um com 1000 bytes, mas por algum motivo o primeiro chega mais rápido e os dois últimos demoraram um pouco.

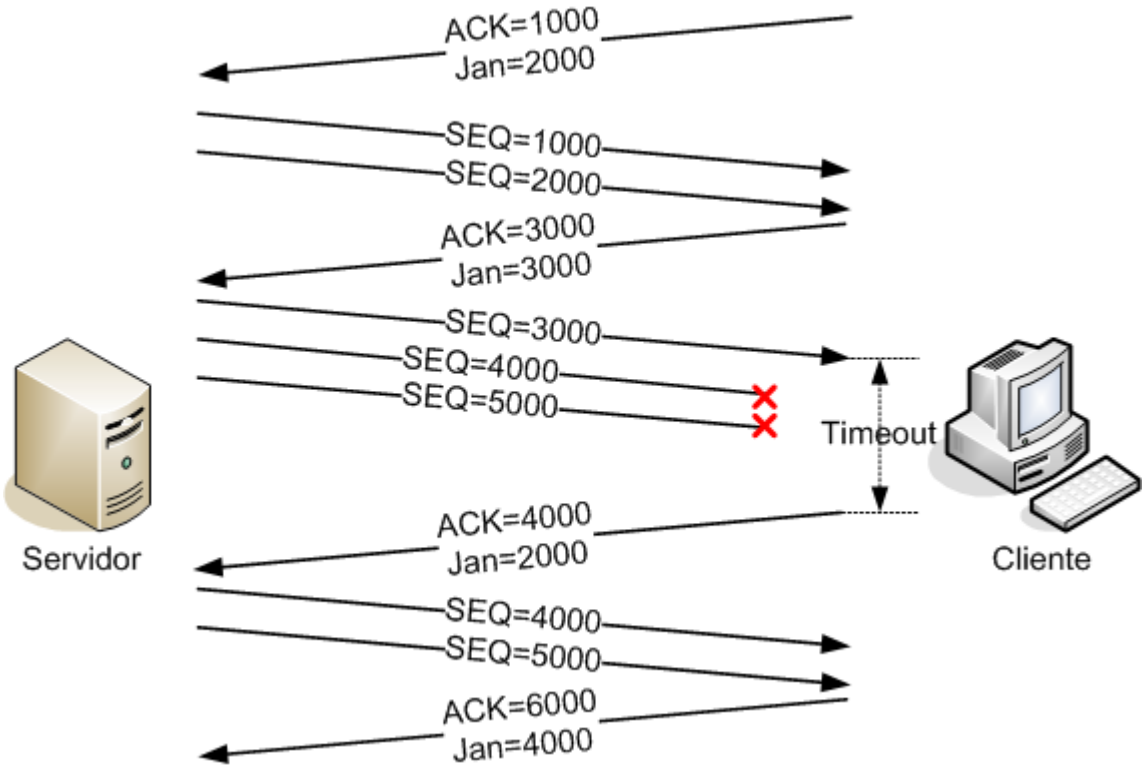
O cliente recebe o pacote de seqüência 1000 espera e não recebe mais nada. Então ele envia uma confirmação: "Bora ai cara!! Eu falei 3000!! Só recebi 1000, manda o próximo (ACK)"

O servidor recebe esse ACK e verifica que enviou 3000 mas só 1000 foram confirmados, ou seja tem 2000 não confirmados e 1000 de espaço livre. Então ele envia mais 1000.

O cliente de repente recebe todos os pacotes, então ele responde: "Beleza, recebi até o 4000, manda o 5000!!! (ACK)"

Então o servidor manda mais 3000 bytes.

Caso o protocolo TCP esteja utilizando janelas deslizantes o tamanho da janela irá variar ao longo de uma transmissão. Ao iniciar uma conexão a janela começa pequena e aumentará gradativamente até que ocorram erros ou o destinatário seja sobrecarregado. Ao serem detectados erros a janela diminui, após um tempo o tamanho da janela começa a aumentar novamente. Caso o destinatário perceba uma sobrecarga, no próximo ACK enviado por ele haverá um novo tamanho de janela, o qual ele acredita ser apropriado para sua recuperação. Caso seja enviado um valor igual à zero o destinatário esta informando que não possui condições de processar mais dados e a comunicação estará suspensa até que o remetente receba um tamanho de janela diferente de zero.



Comunicação com janela deslizante

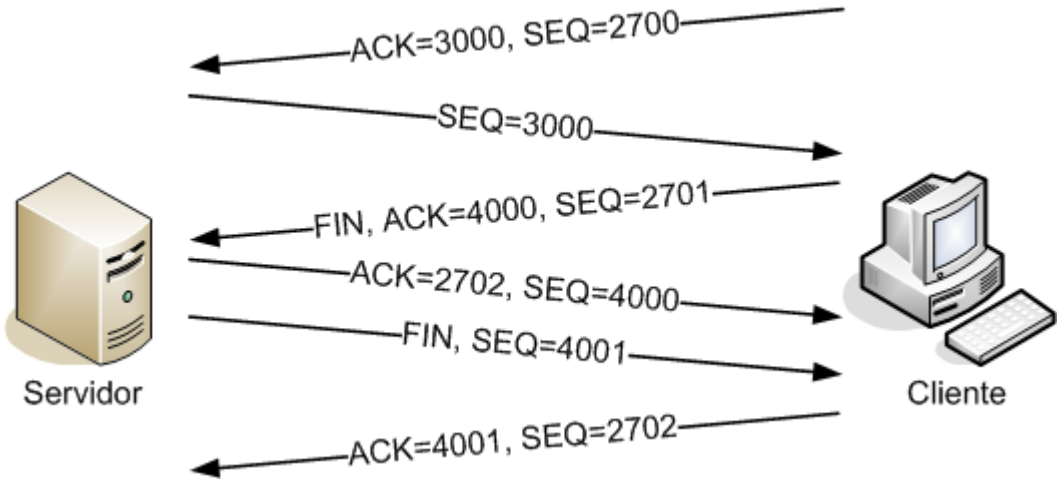
Aqui vemos uma inicio com uma janela de 2000, logo depois sendo incrementada pra 3000. Como houve um timeout, o cliente imagina que pode ter havido algum problema e solicita a redução do tamanho da janela para 2000. Como ele percebeu que tudo ocorreu bem, ele solicita uma aumento para 4000.

É por causa desse comportamento que o tempo de download nunca é confiável, pois o número de bytes transmitidos é variável. E também por isso que no início o download começa com uma taxa de transferência baixa e vai aumentando aos poucos

Finalizando Conexões

A finalização de uma conexão TCP é feita por meio de uma confirmação de três ou quatro vias. Nela é utiliza a *flag* FIN para indicar um pedido de desconexão. Este procedimento deve ser feito em ambas as direções.

Finalização em três vias



Portas de Serviço

Como vimos, a numnerção de portas utiliza 16 bits. Se calcularmos (2^16) veremos que existem ao todo 65536 portas a serem utilizadas. Isso para cada protocolo da camadas de transporte. Logo temos 65536 portas para o TCP, 65536 portas para o UDP, 65536 portas... Conhecer essas portas é fundamental para operar um Firewall de forma satisfatória.

"Ok, mas com tanta porta como vou poder saber todas elas??". Bem, você não precisa conhecer todas. Até mesmo porque a maior parte delas não são especificadas. Para a nossa alegria, apenas as primeiras 1024 são especificadas. Acho que não ajudou muito, ne?! Ok, vamos melhorar. Para um administrador de rede é imprecindível saber pelomenos as portas dos serviços básicos de Rede: telnet, SSh, FTP, SMTP, POP, HTTP, HTTPS... Não são muitas, mas antes de ver isso, vamos entender que controla essas portas.

O uso das portas de 1 a 1024 é padronizada pela IANA (**I**nternet **A**ssigned **N**umbers **A**uthority). Essa entidade é responsável por alocar portas para determinados serviços. Essas portas são chamadas de **well-known ports**.

Abaixo a lista de algumas principais portas TCP:

serviço	porta	protocolo
daytime	13	tcp e udp
ftp-data	20	tcp
ftp	21	tcp
ssh	22	tcp
telnet	23	tcp
smtp	25	tcp e udp
name	42	tcp e udp
nameserver	42	tcp e udp
tftp	69	tcp e udp
www	80	tcp

serviço	porta	protocolo
pop3	110	tcp e udp
netbios-ns	137	tcp e udp
netbios-dgm	138	tcp e udp
netbios-ssn	139	tcp e udp

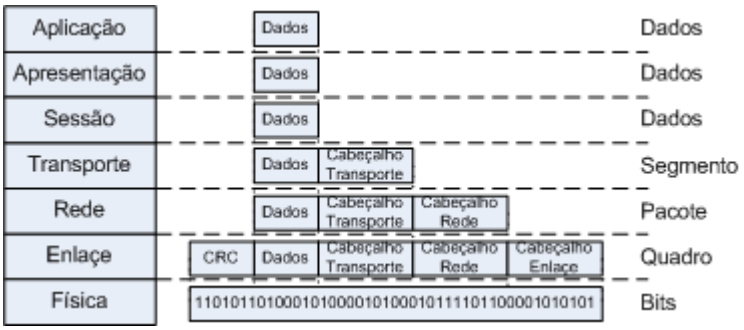
Uma lista com todas as portas você pode encontrar aqui.

É muito importante também conhecer a peculiaridade de cada serviço. Por exemplo, o FTP, tem a possibilidade de utilizar o modo ativo ou passivo, em cada modo as portas são utilizadas de uma forma e a sintaxe das regras de firewall irá mudar.

Uma recomendação de leitura para uma melhor entendimento sobre portas e protocolos pode ser acessada aqui.

Encapsulamento

O encapsulamento é um conceito muito importante. Ao longo do curso nos vimos os cabeçalhos de vários protocolos. E cada vez que era adicionado um cabeçalho eu chamava o conjunto com um nome. Abaixo uma imagem que exemplifica o encapsulamento de dados:



Como podemos ver na imagem, à medida que o dado vai descendo cada camada, ele recebe um cabeçalho. Esse conjunto, "dados da camada superior"+"cabeçalho" é chamado de PDU (Protocol Data Unit - Unidade de dados do protocolo). Cada PDU tem um nome:

- A PDU das camadas de 7 a 5 são chamados de dados, pois esses são os dados puramente ditos;
- A PDU da camada 4 é chamada de segmento;
- A PDU da camada 3 é chamada de pacote;
- A PDU da camada 2 é chamada de quadro/frame;
- A PDU da camada 1 é chamada de bits.

Vendo isso podemos entender a comunicação camada a camada na origem/destino.

Camada de Sessão

A camada de sessão define como iniciar, controlar e finalizar conversações (chamadas de sessões). Isso inclui o controle e o gerenciamento de múltiplas mensagens bidirecionais, de forma que a aplicação possa ser notificada se apenas algumas mensagens de uma série foram completadas. Isso permite que a camada de apresentação tenha pleno conhecimento de um fluxo de dados de entrada.

Chega a parecer que a camada de sessão faz a mesma coisa que a camada de transporte mas não se engane, são operações completamente diferentes. Enquanto a camada de transporte se responsabiliza pelo controle de mensagens e congestionamento utilizando confirmações e janelas deslizantes a camada de sessão controla a "logica" da troca de mensagens. Ela especifica parâmetros utilizados, seqüências e finalizações para uma dada aplicação.

Alguns exemplos de protocolos da camada de sessão:

- RPC;
- SQL;
- NFS;
- Nomes NetBios;
- AppleTalk ASP;
- DECnet SCP.

Quando vocês verem a camada de aplicação ai é que vai complicar mesmo, porque vai parecer que a aplicação faz a função da camada de sessão. Só que dessa vez é verdade! O controle de sessões na verdade passou a ser implementado na aplicação.

Camada de Apresentação

A principal finalidade dessa camada é a de definir formatos de dados, como textos ASCII e EBCDIC, binário, BCD e JPEG. Em outras palavras, a camada de apresentação define como as tudo deve ser representado, formatado e compactado. A criptografia também é definida pela OSI como um serviço da camada de apresentação, mas, como no caso do controle de sessão, essa função é atribuída à aplicação.

Alguns exemplos de protocolos da camada de apresentação:

- JPEG;
- ASCII;
- EBCDIC;
- TIFF;
- GIF;
- PICT;
- MPEG;
- MIDI.

Camada de Aplicação

Uma aplicação que se comunica com outros computadores está implementado os conceitos referentes à camada de aplicação do modelo OSI. A camada de aplicação se refere aos serviços de comunicação para aplicativos, em outras palavras a camada de aplicação é o próprio aplicativo e suas regras (protocolos) de comunicação.

Alguns exemplos de aplicativos/protocolos da camada de apresentação:

- Telnet;
- HTTP;
- FTP;
- Navegadores web;

- NFS;
- Gateways SMTP (clientes de e-mail);
- SNMP.

Fechamento do Curso

Bem galera é isso ai! Acabou o curso! Mas isso não quer dizer que vocês não podem tirar as dúvidas e fazer perguntas e/ou sugestões.

Espero que esse curso tenha ajudado alguém que estivesse começando no mundo das redes de computadores e que estivesse precisasando de uma mãozinha!

Até mais...
Magnun Leno

IPv6

Introdução

Com o crescimento exponencial da internet os endereços IPs começaram a se tornar escassos e algumas técnicas paleativas, como o NAT, foram criadas. Mas essas técnicas só atrasaram o que estava por vir!

Concebido a aproximadamente um década, o IPv6 é a evolução natural do IPv4. Enquanto o IPv4 utiliza 32 bits para endereçamento (4.294.967.296 endereços possíveis) o IPv6 utiliza 128 bits resultando em 340.282.366.920 seguido por mais 27 casas decimais, simplesmente absurdo! Antes que alguém pergunte, entre o IPv4 e o IPv6 houve o IPv5, que era um padrão de **streaming** que nunca vingou.

Ok, mas antes de falar em endereçamento vamos um pouco mais a fundo na estrutura do IPv6.

Arquitetura do IPv6

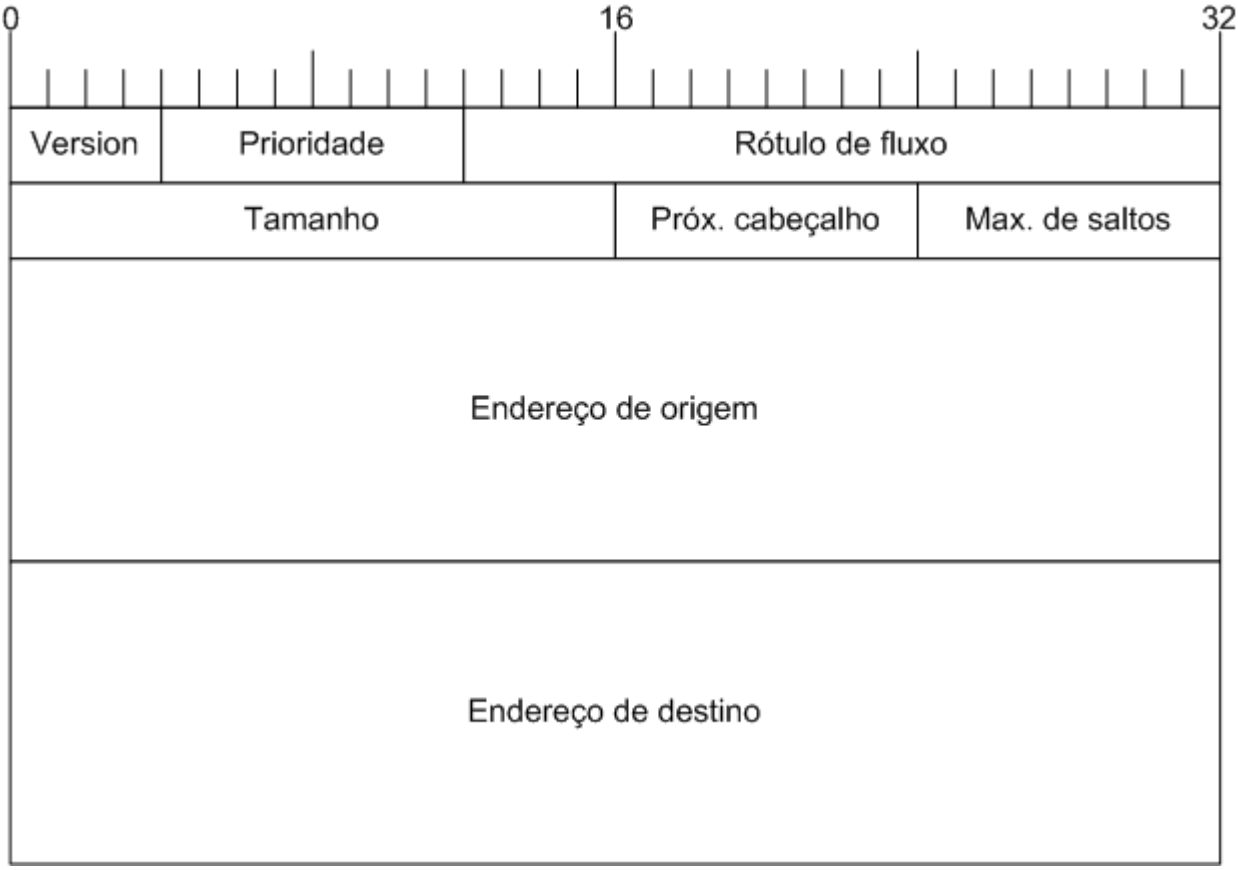
Quando os projetistas criaram o IPv6 uma das principais preocupações era a velocidade. Focando nesse ponto foram feitas algumas medidas para tentar tornar as redes IPv6 mais rápidas.

A primeira medida foi tornar o cabeçalho do pacote IPv6 de tamanho fixo. Essa medida reduz o esforço dos ativos de rede por eles sabem exatamente o que cada bit significa, sem haver a necessidade de um pre-processamento. Para que isso fosse possível alguns campos do antigo cabeçalho IPv4 deixaram de existir, foram criados novos campos e alguns tiveram seu significado modificado.

Outras medidas tomadas para tornar as redes IPv6 mais rápidas são:
- Suporte nativo a multicasting;
- Extinção do Broadcast;
- Tratamento diferenciado para diferentes tipos de serviço e fluxos de dados.

Outras características do IPv6 são:
- Aumento do número de bits de endereçamento, permitindo um número enorme de endereços;
- Criação de um novo tipo de endereçamento: o Anycast;
- Suporte nativo a autenticação e privacidade (segurança);

O cabeçalho do IPv6 tem o seguinte formato:



Vamos analisar os campos desse cabeçalho:

- Versão (4 bits) - Este campo identifica a versão do protocolo usado para criar o pacote.
- Prioridade (Classe de tráfego) (8 bits) - É equivalente ao campo Tipo de Serviço do IPv4, que classifica o tipo do pacote.
- Rótulo de fluxo (20 bits) - Serve para identificar um fluxo.
- Tamanho (16 bits) - Este é o número de bytes de dados contidos no pacote
- após o cabeçalho.
- Próximo cabeçalho (8 bits) - Informa o protocolo que deve tratar o conteúdo do pacote.
- Limite de saltos (8 bits) - É o número máximo de roteadores que o pacote deve passar, a cada roteador este campo é decrementado, quando chega em zero o pacote é descartado. Esse é o antigo Time to Live (TTL)
- Endereço de origem (128 bits) - Origem do pacote.
- Endereço de destino (128 bits) - Destino do pacote

"Perai, mas o que é um fluxo?!"
Um fluxo é uma seqüência de pacotes para os quais é necessário um tratamento especial, como para serviços que requerem alta prioridade tais como serviços de tempo real, transmissão de vídeo ou uma conexão entre dispositivos móveis onde qualidade durante a transmissão deve ser assegurada. Dados que pertençam a aplicações tradicionais como requisições http ou transferência de arquivos não precisam participar de um fluxo.

Analisando o cabeçalho

O cabeçalho do protocolo IPv6 é extremamente mais simples que o do seu antecessor. Só pra ter uma idéia, o cabeçalho do IPv6 tem 8 campos enquanto o cabeçalho do moribundo IPv4 possui 14 campos! Isso não quer dizer que todos os campos foram excluídos. Simplesmente quer dizer que foi mantido no cabeçalho somente os essenciais para os roteadores. Qualquer informação adicional às que constam no cabeçalho serão transmitidas encapsuladas e informadas no campo próximo cabeçalho.

Dentre os campos que foram eliminados, ressalto dois que merecem destaque: Checksum e fragmentação.

O falecido Checksum

Para quem não lembra, a função do Checksum era verificar a integridade do cabeçalho, em outras palavras, detectar

erros no cabeçalho. É de conhecimento geral que a maioria dos erros que ocorrem nas redes não é de transmissão mas sim nos roteadres, além do mais, as técnicas de verificação de integridade de camada 2 já são muito eficientes. Outro agravante do campo checksum era o desempenho dos roteadores. Como cada roteador decrementava o campo Time to Live (TTL) era necessário realizar novamente o calculo do checksum antes de retransmitir o pacote. Antes que alguém diga que isso é um esforço mínimo vamos pensar assim: O roteador calcula o checksum ao receber o pacote, para comparar com o que está no cabeçalho, realiza a decisão de roteamento, decrementa o TTL, calcula o novo checksum e o inclui no cabeçalho, retransmite o pacote. Considerando isso para milhares de pacotes por segundo é um esforço tremendo!

O "migrado" Fragmentação

Esse simplesmente é uma convenção, ele não morreu de verdade! Decidiu-se que os roteadores não serão mais responsáveis por fragmentar pacotes. caso o roteador receba um pacote com o tamanho maior que o permitido ele será descartado, simples assim! Ao descartar o pacote o roteador informará o ocorrido à origem, que deverá retransmitir o pacote já fragmentado. Dessa forma tiramos mais um pouco a carga do roteador! O porque de "migrado" vocês vão entender daqui a pouco!

Cabeçalhos complementares

Como dito, algumas informações foram colocadas em cabeçalhos encapsulados no campo de dados do pacote IPv6. Podemos dizer que eles são "sub-cabeçalhos". Os possíveis tipos são os seguintes:

- Hop-by-Hop - Transporta informações opcionais que são processadas em cada nó ao longo do camino do pacote, incluindo a origem e o destino;
- Destination Options - Transporta informações opcionais que são processadas apenas pelo destino final do pacote;
- Routing - Utilizado no suporte a mobilidade do IPv6, ele armazena o endereço original de um nó movel (Type 2);
- Fragmentation - Utilizado pela origem para enviar pacotes maiores que o Maximum Trnasmit Unit (MTU) de um caminho;
- Authentication - Utilizado pelo serviço IPSec (IP Security) para prover autenticação e garantia de integridade aos pacotes IPv6. Esse cabeçalho é idêntico ao utilizado no IPv4;
- Encapsulating security payload - Também utilizado pelo IPSec, provê integridade e confidencialidade para os pacotes.

Podemos notar que esses campos são voltados principalmente para origem e destino e não interessa para os roteadores, com excessão do campo Hop-by-Hop. Podemos ver também que o Fragmentação está lá, por isso chamei ele de "migrado"!

Endereçamento IPv6

Como dito anteriormente no IPv6 os endereços IPs passaram de 32 bits para 128 bits. Com essa mudança temos que alterar também a forma como escrevemos os endereço IP.

Antes o endereço IP era dividido em 4 octetos, onde cada octeto era composto por oito bits, e escrito utilizando a notação decimal pontuada.

Como o novo endereço IP é muito extenso não podemos utilizar a notação decimal, dessa forma utilizaremos caracteres hexadecimais. No total teremos 32 caracteres organizados em oito quartetos e separados por dois pontos. Este é um exemplo de um endereço IPv6 válido:

2001:bce4:5641:3412:341:45ae:fe32:65

Vocês devem estar pensando: "Vou ter que decorar tudo isso?!?! 192.168.1.1 era muito mais fácil!!!". Pensando nisso o endereço IPv6 pode ser abreviado, por exemplo, fee::1. Agora ficou melhor... Só ressaltando que essa abreviação é válida para os zeros. Isso quer dizer que entre o fee e o 1 existem 6 campos de 4 caracteres todos preenchidos com zero, por exemplo, o endereço FEDC:0034:0000:0000:0000:0012:0ABC:00FF pode ser abreviado para FEDC:34:0:0:0:12:ABC:FF ou FEDC:34::12:ABC:FF.

O mais interessante dos endereços IPv6 é a retrocompatibilidade. Você pode 'usar' o seu antigo IPv4 como se fosse um IPv6 sem problemas. Digamos que antes seu IP era 192.168.1.1, no IPv6 ele será 0:0:0:0:0:FFFF:192.168.1.1 ou ::FFFF:192.168.1.1. Esse é uma definição relativamente nova. Anteriormente a compatibilidade era feita adicionando dois dois pontos, ::, da seguiinte forma, ::192.168.1.1, mas essa utilização caiu em desuso e o notação iniciada por ::FFFF passou a ser adotada (vide RFC 4038). Mas esse é um assunto que abordaremos mais tarde!

Segmentação de redes

Assim como no IPv4 o IPv6 também possui a estrutura de endereçamento de grupos lógicos chamados de redes. Da mesma forma que no IPv4, utilizamos a máscara de rede para definir o "tamanho" da rede. Porém, no IPv6 não teremos a mesma flexibilidade de utilização de sub-redes.

Como no IPv6 teremos muitas redes disponíveis não será necessário a utilização de NAT e redes privadas. Com isso serão distribuídas redes de tamanho fixo.

Vamos ver como essa atribuição é feita:

- A RIR (Regional Internet Registries) recebe da IANA (Internet Assigned Numbers Authority) um bloco de endereços /12.
- Grandes provedores recebem do seu respectivo RIR blocos de endereços /32.
- Os provedores por sua vez podem distribuir para seus clientes blocos entre /48 e /56, dependendo da necessidade.
- Caso o usuário tenha plena certeza de que apenas uma rede é o suficiente, o provedor pode entregar um bloco /64.

Para termos uma idéia, um bloco /48 pode dividido em até 65.536 redes diferentes, cada uma com 18.446.744.073.709.551.616 endereços diferentes. Um bloco /56 pode ser dividido em 256 redes diferentes, dada uma com 18.446.744.073.709.551.616 endereços diferentes.

Dessa forma podemos notar que o menor bloco possível é o /64 que corresponde à metade do endereço IPv6. Considerando o menor bloco IPv6 os primeiros 64 bits, primeiros 4 grupos, são designados para definição de redes. Por exemplo, no endereço 2001:bce4:0:0:0:0:0:1 a rede é definida por 2001:bce4:0:0 enquanto o host é definido por 0:0:0:1.

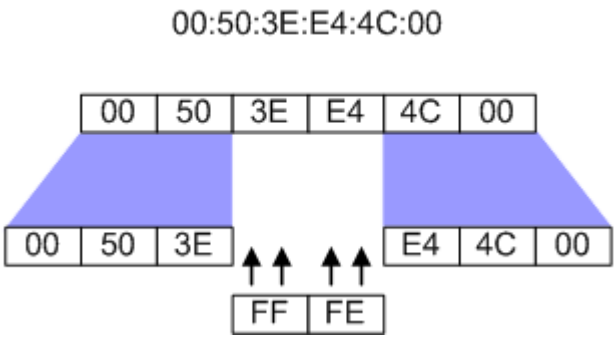
O objetivo dessa segmentação definida é que cada conjunto de números do IPv6 servirá como um identificador.

Atribuição de endereços

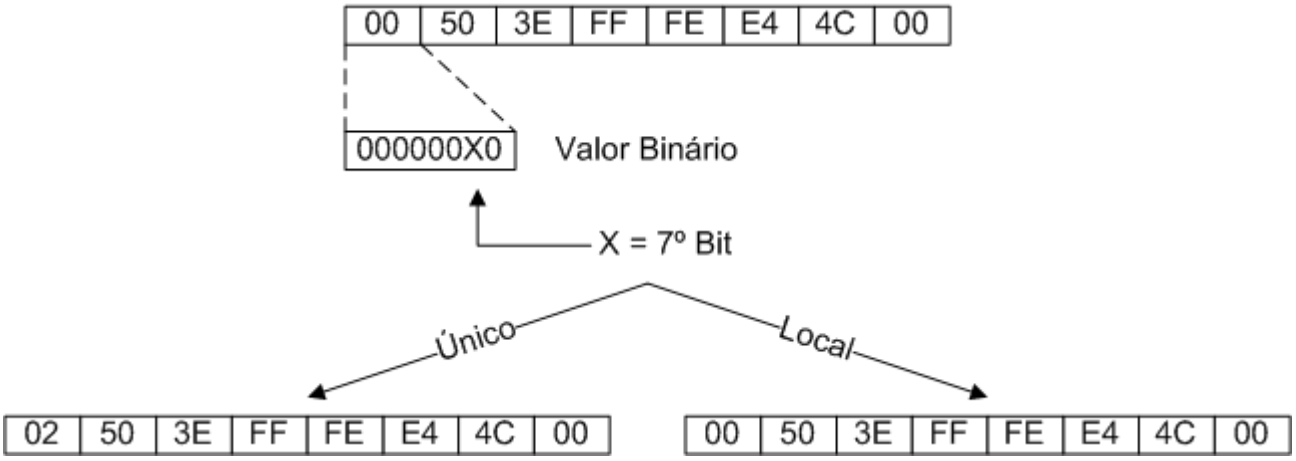
Ao atribuir endereços IPv6 aos hosts de uma rede temos duas opções:

- Utilizar endereços seqüenciais, como " 2001:bce4::1", "2001:bce4::2", "2001:bce4::3" e assim por diante;
- Seguir a sugestão do IEFT e usar os endereços MAC das placas de rede para preencher o campo de endereço de host.

Essa atribuição de IPs utilizando o MAC como identificador de host é uma ótima idéia pois além de tornar os IPs únicos haverá uma "coerência" entre MAC e IP. Se o endereço da rede é 2001:bce4:0:0: e o endereço MAC do micro é 00:50:3E:E4:4C:00 podemos atribuir o seguinte IP: 2001:bce4::0216:f2ff:fefe:34e1. "Espera ai, esse IP está diferente do MAC!". Sim, o endereço MAC contém apenas 12 dígitos hexadecimais (48 bits) enquanto a porção de host do IPV6 possui 16 dígitos hexadecimais (64 bits) por isso foi definido na RFC 2462 que seria utilizado a definição EUI-64, então temos que adicionar os dígitos "fffE" entre o sexto e sétimo dígito do endereço.



Após isso tem um último passo que é definir se o endereço terá uma importância global ou local. "Como assim?!", calma que eu explico. Quando digo global estou me referindo ao MAC propriamente dito, atribuído pelo fabricante, já o local se refere a um MAC alterado, atribuído manualmente, como é feito com frequência em máquinas SUN.



Está em estudo uma expansão dos endereços MAC das placas de rede, que passariam a ter 16 dígitos (64 bits), mas, enquanto isso não é colocado em prática, usamos essa regra.

Redes reservadas

No IPv6, da mesma forma que no IPv4, existem faixas de redes reservadas. Os endereços iniciados com 2001: são reservados para provedores de acesso e carriers. Os endereços iniciados com 3FFF:FFFF: e 2001:0DB8: são reservados para uso em documentação. exemplos e testes e por isso não são roteáveis.

No IPv4 existiam as faixas de IPs reservados para redes locais, ou redes privadas: 10.0.0.0/8, 172.16.0.0/20 e 192.168.0.0/16. Isso também existia no IPv6 mas foi derrubada pela RFC 3879, essas redes eram iniciadas com fec, fed, fee ou fef.

Outro endereço reservado no IPv4 muito famoso era o endereço 127.0.0.1 reservado para interface de loopback, também acessada pelo nome localhost. No IPv6 foi escolhido o IP ::1, que corresponde a 0:0:0:0:0:0:0:1.

Tipos de endereço

O IPv6 mantem os tipos de endereço unicast e multicast e cria um novo chamado Anycast. O tipo Broadcast foi eliminado, pois era ineficiente, além de muitas vezes gerar congestionamento na rede. O broadcast será substituido pelo multicast.

Unicast é endereçamento ponto-a-ponto tradicional. Um pacote destinado a um endereço unicast é entregue apenas àquela interface que possui o endereço especificado.

O Multicast é um tipo de endereçamento que foi acrescentado ao IPv4, mas que já é nativo no IPv6. Um pacote

destinado a um endereço multicast é entregue a todas as interfaces que fazem parte daquele grupo de endereços.

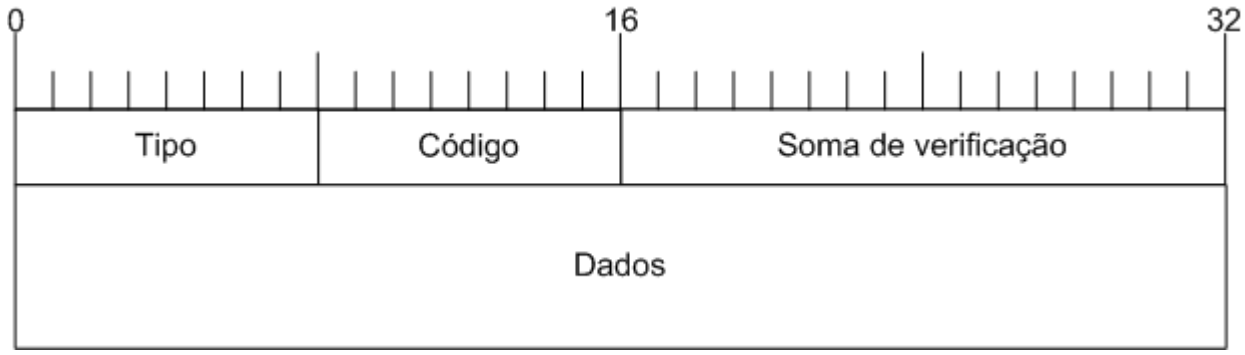
O Anycasting é um novo tipo de endereçamento, que surge com o IPv6 e permite que ao invés de enviar um pacote a um servidor específico, este seja enviado a um endereço anycast específico. O sistema de roteamento entregará então o pacote ao servidor mais próximo, de acordo com a sua medida de distância, dentre um grupo de servidores. Isto é interessante, por exemplo, para descobrir o servidor de nomes mais próximo.

ICMPv6

O ICMP do IPv6 é muito semelhante ao ICMP do IPv4, ele ainda tem a função de informar características da rede, realizar diagnóstico e relatar erros no processamento de pacotes. Estes processos são executados utilizando duas classes de mensagens: Erro e Informação

Quando falamos sobre cabeçalhos IPv6 vimos que existem os cabeçalhos de extensão e que eles são identificados no cabeçalho IPv6 pelo campo "Próximo cabeçalho" (Next Header). O ICMP é identificado no campo "próximo cabeçalho" pelo valor 58. Dessa forma teremos um encadeamento de cabeçalhos. É importante ressaltar que o cabeçalho ICMPv6 NÃO É UM cabeçalho de extensão, pois o ICMPv6 é um protocolo de camada 3 e não faz "parte" do IPv6. Os cabeçalhos de extensão são "partes" do IPv6, eles complementam as informações do cabeçalho IPv6, enquanto o ICMPv6 é um protocolo "paralelo".

O ICMPv6 possui o seguinte cabeçalho:



- **Tipo** (Type) - Indica o tipo de mensagem;
- **Código** (Code) - Oferece algumas informações adicionais para determinados tipos de mensagens.
- **Soma de verificação** (Checksum) - É utilizado para detectar dados corrompidos no cabeçalho IPv6.
- **Dados** - São apresentadas informações de diagnóstico e erro de acordo com o tipo de mensagem. Essa campo possui tamanho variável.

Apesar de o ICMPv6 ser muito semelhante ao ICMPv4, o ICMPv6 apresenta mais recursos, consequentemente possui mais mensagens, pois acabou "absorvendo" alguns outros protocolos como ARP, RARP e IGMP. Dessa forma o ICMPv6 torna-se responsável por serviços como:

- Descoberta de vizinhança;
- Gerenciamento de grupos multicast;
- Mobilidade IPv6;
- Descoberta do path MTU.

Vamos ver as mensagens de **erro**, definidas até o momento, no protocolo ICMPv6:

- **Destination Ureachable** (Tipo 1) - Indica falhas na entrega do pacote como endereço ou porta desconhecida ou problemas na comunicação;
- **Packet too big** (Tipo 2) - Indica que o tamanho do pacote é maior que a Unidade Máxima de Transmissão (MTU) de um enlace;
- **Time Exceeded** (Tipo 3) - Indica que o limite de roteamento ou o tempo de remontagem do pacote foi excedido;
- **Parameter problem** (Tipo 4) - Indica erro em algum campo de cabeçalho IPv6 ou que o tipo indicado no campo próximo cabeçalho não foi reconhecido.

Vamos ver as mensagens de **informação**, definidas até o momento, no protocolo ICMPv6:

- **Echo request** (Tipo 128) e **Echo reply** (Tipo 129) - utilizados pelo comando ping;
- **Multicast Listener Query** (Tipo 130)**/Report** (Tipo 131)**/Done** (Tipo 132) - Utilizadas no gerenciamento de grupos multicast;
- **Router Solicitation** (Tipo 133)**/advertisement** (Tipo 134), **Neighbour Solicitation** (Tipo 135)**/Advertisement** (Tipo 136), **Redirect Message** (Tipo 137) - Utilizadas com o protocolo Descoberta de Vizinhança;
- **Router Renumbering** (Tipo 138) - Utilizada no mecanismo de Re-endereçamento de roteadores;
- **ICMP Node Information Query** (Tipo 139)**/Response** (Tipo 140) - Utilizada para descobrir informações sobre nomes e endereços, são atualmente limitadas a ferramentas de diagnóstico, depuração e gestão de redes;
- **Inverse Neighbor Discovery Solicitation** (Tipo 141)**/Advertisement** (Tipo 142) - Utilizadas em uma extensão do protocolo de descoberta de vizinhança;
- **Version 2 Multicast Listener Report** (Tipo 143) - Utilizada no gerenciamento de grupos multicast;
- **Home Agent Addess Discovert Requset** (Tipo 144)**/Reply** (Tipo 145) e **Mobile prefix solicitation** (Tipo 146)**/advertisement** (Tipo 147) - Utilizadas no mecanismo de Mobilidade IPv6;
- **Certification Path Solicitation** (Tipo 148)**/Advertisement** (Tipo 149) - Utilizadas pelo protocolo SEND;
- **Multicast Router Advertisement** (Tipo 151)**/Solicitation** (Tipo 152)**/Termination** (Tipo 153) - Utilizada pelo mecanismo Multicast Router Discovery;
- **FMIPv6 Messages** (Tipo 154) - Utilizada pelo protocolo de mobilidade Fast Handovers.

Protocolo de Descoberta de Vizinhanças

O IPv6 utiliza o protocolo de Descoberta de Vizinhanças, que já era utilizado no IPv4, que foi replanejado, aprimorado e expandido. Esse protocolo é utilizado por hosts e roteadores para os seguintes fins:

- Divulgar o endereço MAC dos nós da rede;
- Encontrar roteadores vizinhos;
- Determinar prefixos e outras informações de configuração de rede;
- Detectar endereços duplicados;
- Determinar a acessibilidade dos roteadores;
- Redirecionamento de pacotes;
- Autoconfiguração de endereços.

Podemos ver que estão listadas ai em cima funções dos protocolos ARP, ICMP e DHCP. Para desempenhar essas funções o protocolo de descoberta de vizinhança se utiliza das mensagens ICMP. Abaixo está a lista das cinco mensagens utilizadas por ele:

- **Router Advertisement** - Enviadas periodicamente, ou em resposta a uma Router Solicitation, pelos roteadores da rede para anunciar sua presença em um enlace e na internet;
- **Router Solicitation** - Utilizada por hosts para requisitar aos roteadores mensagens Router Advertisements imediatamente;
- **Neighbor Solicitation** - Mensagem multicast enviada por um nó para determinar o endereço MAC e a acessibilidade de um vizinho, além de detectar a existência de endereços duplicados;
- **Neighbor Advertisement** - Enviada como resposta a uma Neighbor Solicitation, pode também ser enviada para anunciar a mudança de algum endereço MAC dentro do enlace;
- **Redirect** - Utilizada por roteadores para informar ao host um roteador mais indicado para se alcançar um destino.

Todas essas mensagens possuem o campo "Máximo de saltos" no cabeçalho IPv6 setado como 255. Dessa forma essa mensagem fica restrita a um único enlace pois, como está configurado o limite de saltos, esse pacote não será encaminhado pelos roteadores. Essas mensagens que possuam um valor máximo de saltos diferente de 255 serão descartadas pelos roteadores.

ARP no IPv6

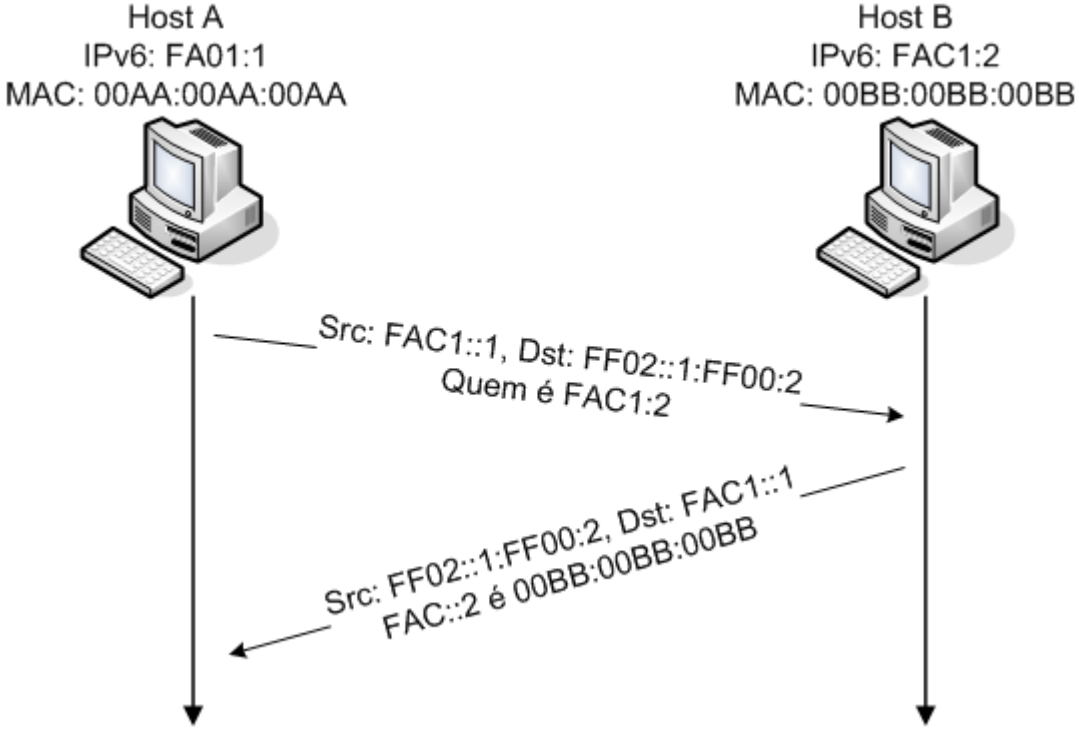
Como mostrado, a funcionalidade de "Descoberta de Endereços de Camada de Enlaçe", do protocolo ARP, foi incluída

no protocolo de Descoberta de Vizinhança. Podemos dar adeus ao ARP e seus broadcast que inundavam a rede, agora isso é feito pela "Descoberta de Vizinhança" utilizando multicast.

De um modo geral, se um Host A quer descobrir o MAC do HOST B ele envia uma mensagem "Neighbour Solicitation" para o endereço de multicast "Solicited-node Multicast " (que é composto pelo prefixo FF02::1:FF00:0/104 e os últimos 24 bits do endereço IPv6 que está sendo consultado) e recebe do HOST B uma resposta "Nighbor Advertisement" contendo o MAC solicitado.

Complicou?!?! Vamos a um exemplo:

- Você emite um ping, do HOST A, para o HOST B (IP FAC1::2) como você não conhece o MAC será realizada uma consulta;
- É enviada uam pacote mensagem "Neighbour Solicitation" para o IP de multicast FF02::1:FF00:2 do HOST B.
- O HOST B envia, em resposta para ao HOST A, um "Nighbor Advertisement" (em unicast) contendo o seu MAC.



Para quem não conhece multicast isso pode parecer um absurdo! "Como ele faz uma consulta MAC a vários destinos usando um endereço que não seja broadcast??" É muito estranho mesmo, mas depois que se entende como o multicast funciona, o broadcast se mostra extremamente obsoleto! Um dia posto algumas coisas aqui sobre multicast!

Descoberta de roteadores e prefixos

Autoconfiguração Stateless

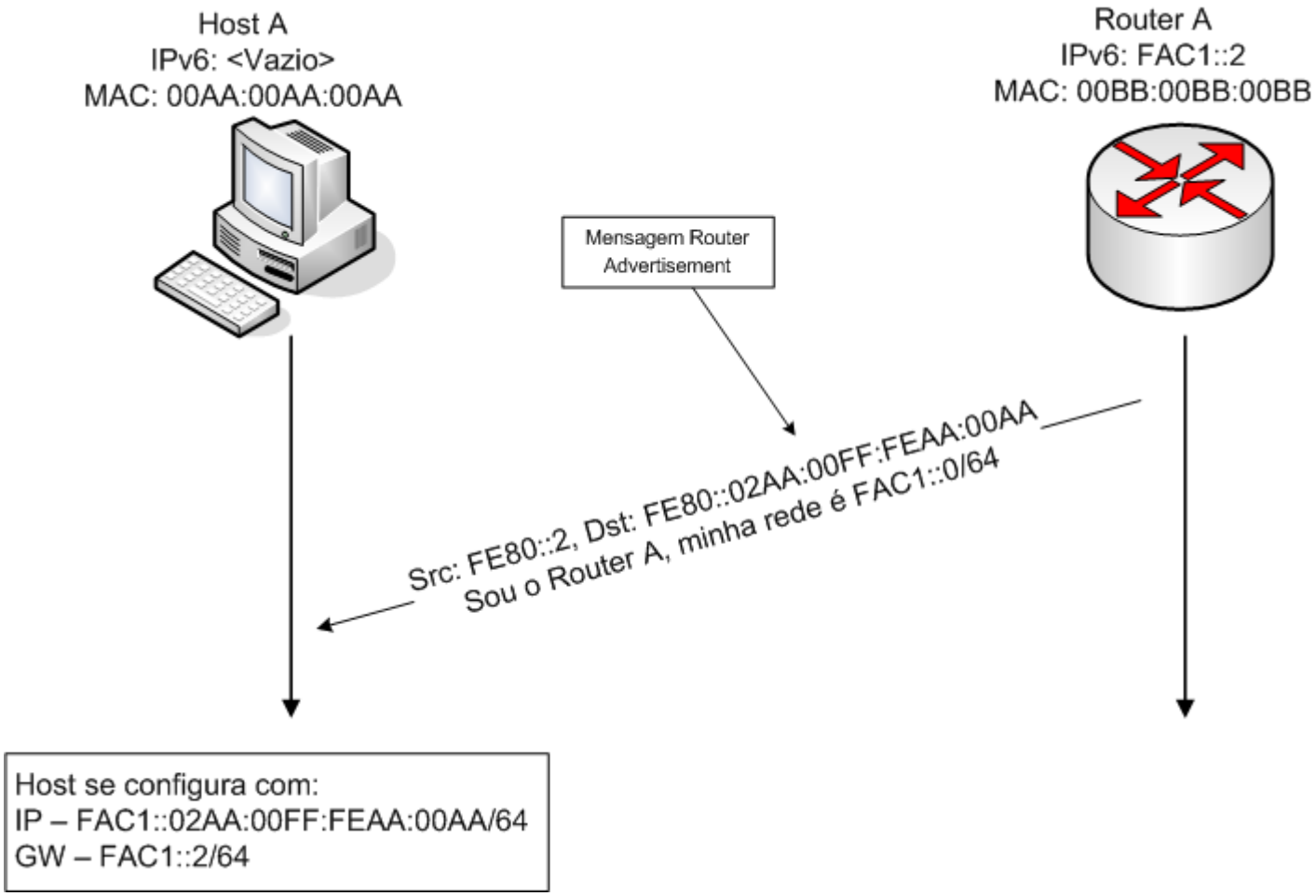
Esse mecanismo possibilita a atribuição automática de endereços unicast, a descoberta de roteadores no mesmo enlace de camada 2 e a atribuição automática de mais alguns parâmetros de rede. Essa atribuição é realizada sem a necessidade de servidores adicionais, apenas com configurações mínimas dos roteadores.

Como no inicio desse processo os hosts não possuem endereços eles utilizam o endereço reservado "link-local" que são gerados com o prefixo FE80::/64 com o MAC 'expandido' da interface. A rede "link-local" está presente em todas as interfaces de qualquer dispositivo que fale IPv6. Essa rede não é roteavel e nem é divulgada por protocolos de roteamento dinâmico mas pode ser utilizada para testes de conectividade normalmente.

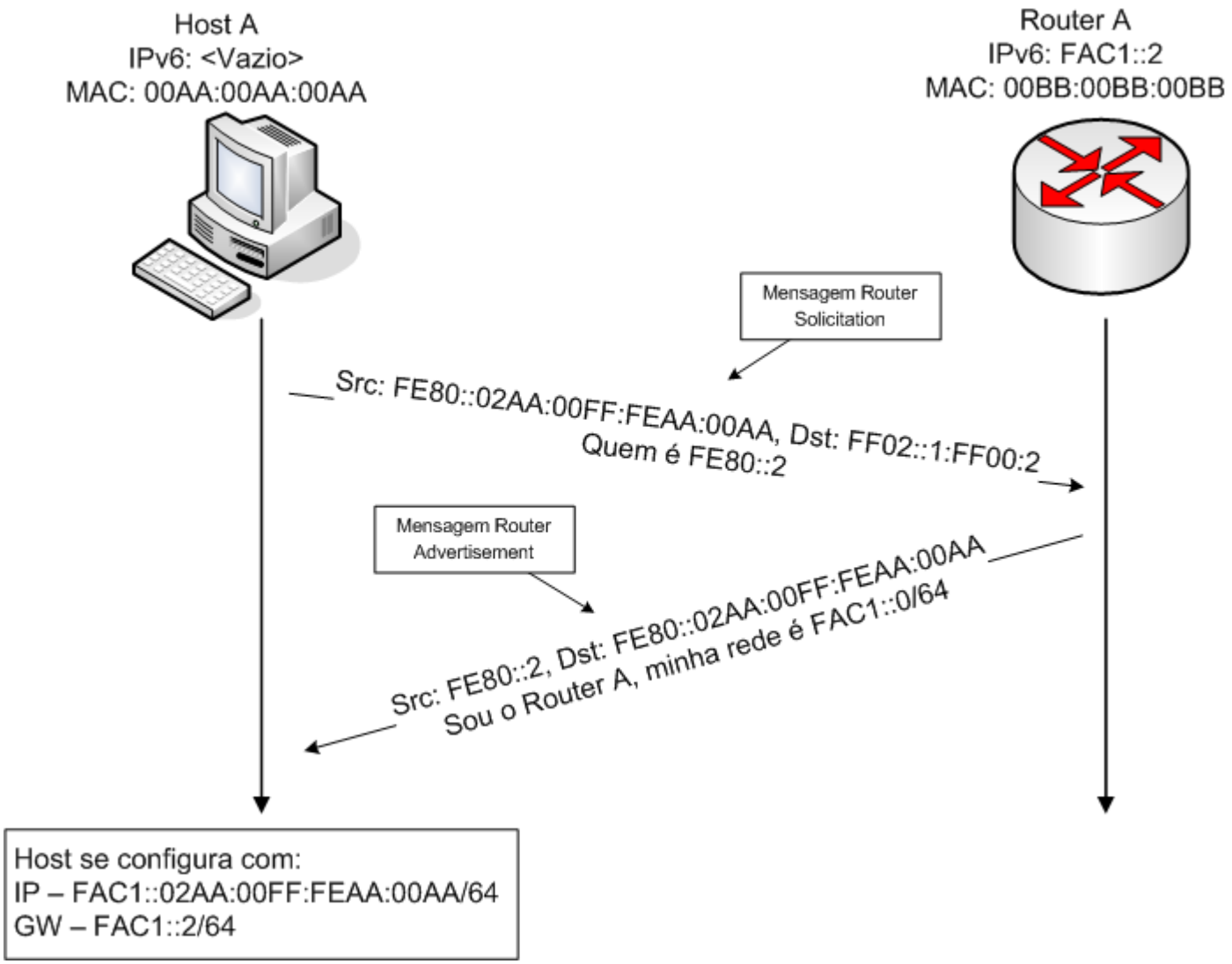
Essa descoberta pode ocorrer de duas formas: **Consulta por parte de um Host** ou **Anuncio gratúito de um roteador**.

Anúncio Gratuito
Quando há um anuncio gratuito de um roteador, ele envia uma mensagem "Router Advertisement" para o endereço "All-nodes Multicast" (endereço FF02::1) contendo informações necessárias para a autoconfiguração. Com o conteúdo

dessa mensagem o host pode auto configurar os seguintes atributos: Gateway padrão; lista de endereços de rede; MTU; o "Máximo de Saltos" (para o cabeçalho IPv6); e as flags "Managed Addess Configuration" e "Stateful Configuration".



Consulta de um Host
Quando há uma consulta por parte do host, é enviado uma mensagem "Router Solicitation" para o endereço "All-routers Multicast" (endereço FF02::2) e os roteadores do segmento respondem com uma mensagem "Router Advertisement".



Vamos explicitar a função das duas flags citadas:

- **Managed Addess Configuration** - Indica se os hosts devem ou não utilizar a autoconfiguração stateful para obter endereços;
- **Stateful configuration** - Indica se o hosts devem utilizar a auto configuração stateful para obter informações adicionais, como endereços de servidores DNS e outros dados sobre a configuração de rede.

"Então quer dizer que o DHCP morreu?!?!". Não, o DHCP ainda existe...

Autoconfiguração Stateful

A auto configuração statuful é uma alternativa à stateless. Nela é utilizado um servidor DHCPv6 para atribuir os endereços. A autoconfiguração stateful é utilizada quando não há um roteador na rede ou quando o roteador indica, por meio das flags "Stateful configuration" e "Managed Addess Configuration" da mensagem "Router Advertisement".

Qual a real necessidade de utilizar uma autoconfiguração stateful? Atualmente a autoconfiguração stateless não é capaz de distribuir endereços de servidores de DNS, NTP, proxy entre outros, além de possibilitar a definição de políticas de controle de acesso.

Está sendo prevista uma expansão para que a autoconfiguração stateless distribua também endereços de servidores DNS.

No DHCPv6 são utilizadas mensagens UDP para a solicitação/divulgação dos IPv6s. Os clientes utilizam o endereço

"link-local" para enviar ou receber mensagens DHCPv6, enquanto os servidores utilizam os endereços Multicast "All-dhcp-agents" (FF02::1:2) ou "All-dhcp-servers" (FF05::1:3) - Estes endereços foram definidos na RFC RFC3315 - para receber as mensagens dos clientes. Como esses endereços Multicast possuem apenas importância local, para receber mensagens de uma rede remoto é necessário a configuração de um Relay DHCP.

Detecção de endereços duplicados.

No IPv4 essa função era feito pelo protocolo ARP utilizando ARPs gratuitos, já no IPv6 isso será feito utilizando mensagens "Neighbor Solicitation" para o endereço "All-nodes Multicast". O detentor de um endereço IPv6 envia uma mensagem "Neighbor Solicitation" e aguarda uma resposta. Caso haja uma resposta ele sabe que o IP que ele utiliza está duplicado

Detecção de vizinhos inacessíveis.

Esse mecanismo é utilizado para comunicação host-a-host, roteador-a-host ou host-a-roteador. Isso é feito monitorando o recebimento de pacotes de confirmação de entrega a um certo vizinho. Essa confirmação pode ser uma resposta a uma mensagem de "Descoberta de Vizinhança" ou algum pacote de camadas superiores a 3. É importante ressaltar que esse monitoramento só é realizado para comunicações unicast. Para esse rastreamento são utilizadas duas tabelas:

- **Neighbor Cache** - Mantem uma lista de vizinhos locais para os quais foi enviado tráfego recentemente. Essa lista contem o endereço IP, o endereço MAC, uma flag que identifica se esse IP é um Host ou um Router, se há pacotes na fila para serem enviados a esse destino, a sua acessibilidade e a próxima vez que um evento de detecção de vizinhos está agendado. Essa tabela é "semelhante" a tabela ARP do IPv4.
- **Destination Cache** - Mantem informações sobre destinos, locais e/ou remotos, para os quais foi enviado tráfego recentemente. As entradas dessa tabela são atualizadas com informações recebidas por mensagens "Redirect". A tabela Neighbour Cache pode ser considerada como um subconjunto dessa tabela.

Redirecionamento

As mensagens de redirecionamento são quase idênticas as mensagens de redirecionamento no IPv4. Ela são enviadas por roteadores e tem como função redirecionar um host automaticamente para um outro roteador mais apropriado ou para informar ao host que o destino encontra-se no mesmo enlace.