Leonardo Ferreira Carneiro
leo@cat.cbpf.br

Nilton Costa Braga
ncb@cat.cbpf.br

Nilton Alves Júnior
naj@cat.cbpf.br

Resumo

Essa nota técnica tem o objetivo de servir como fonte de consulta a todos aqueles profissionais da área técnica, ou àqueles que se interessarem, que queiram incrementar, ou adquirir, conhecimentos relativos à redes de computadores.

Esse trabalho visa englobar todos os aspectos técnicos relacionados à rede de computadores em geral, como os seus objetivos, classificação e estruturação. Falaremos também à respeito das camadas e protocolos que estruturam uma rede, nos fixando no modelo de referência OSI e nos protocolos utilizados na rede do CBPF, além de abordarmos os componentes constituintes de uma rede local, explicando as suas funções. Por último, utilizaremos a própria rede do CBPF como exemplo, explicando a sua estruturação e as funções executadas pelos equipamentos que a compõem.

Índice

Redes de Computadores

Introdução

Cada um dos três últimos séculos foi dominado por uma tecnologia principal. O século XVIII foi a época dos grandes sistemas mecânicos que acompanhavam a Revolução Industrial. O século XIX foi a idade da máquina a vapor. Ao longo do século XX, a tecnologia-chave tem sido a coleta, o processamento e a distribuição da informação. Entre outros desenvolvimentos, assistimos à instalação de redes telefônicas mundiais, à invenção do rádio e da televisão, ao nascimento de computadores e ao lançamento de satélites de comunicação.

À proporção que nos aproximamos do final desse século, essas áreas estão convergindo rapidamente, e as diferenças entre coletar, transportar, armazenar e processar informações estão rapidamente desaparecendo. Organizações em geral, com centenas de escritórios espalhados em uma vasta área geográfica esperam poder verificar a situação até do seu escritório mais remoto com um simples apertar de botão. À medida que aumenta a nossa habilidade de coletar, processar e distribuir informações, aumenta mais rapidamente a demanda por aplicações ainda mais sofisticadas.

Embora a indústria de computadores seja jovem quando comparada com indústrias como a automotiva e a de transportes aéreos, os computadores têm feito um fantástico progresso em um curto espaço de tempo. Durante as suas duas primeiras décadas de existência, os sistemas de computadores eram altamente centralizados, em geral, em uma única sala grande. Uma empresa de porte médio ou uma universidade pública poderia ter um ou dois computadores, enquanto as grandes instituições tinham no máximo uma dúzia. A noção de que dentro de vinte anos computadores igualmente poderosos, menores do que um selo postal, pudessem ser produzidos em massa era considerada pura ficção científica.

A fusão dos computadores e das comunicações teve uma profunda influência sobre a forma como os computadores são organizados. O conceito de "centro de computação" como sendo uma sala com um grande computador, ao qual os usuários levam as suas tarefas para serem processadas, está obsoleto. Esse modelo não tem uma, mas duas falhas: o conceito de um único grande computador fazendo todo o trabalho, e a noção dos usuários levando as suas tarefas para o computador, ao invés de levar o computador até os usuários.

O velho modelo de um único computador servindo a todas as necessidades computacionais da organização está rapidamente sendo substituído por outro no qual um grande número de computadores separados, mas interconectados, executam essa tarefa. Essas são as chamadas redes de computadores. Atualmente, a maioria das organizações que usam computadores já tem, ou estão instalando, uma ou mais redes locais de computadores. Um exemplo típico dessa expansão pode ser visto no fato de que o correio eletrônico em âmbito mundial é uma realidade diária para milhões de pessoas. Podemos

perceber com isso que as redes de computadores tornam-se uma ferramenta de vital importância aos usuários de empresas, governos e universidades.

Há somente alguns anos atrás, o projeto de uma rede era considerado obra de um mágico, uma vez que cada fabricante de computadores tinha a sua própria arquitetura de rede, e não se encontrava um par de arquiteturas que fossem iguais. Felizmente, esse quadro mudou. Uma série de Padrões Internacionais para a descrição de arquiteturas de redes de computadores foi aceita por toda a indústria de computadores, sendo esses padrões conhecidos como o Modelo de Referência OSI, que será estudado posteriormente nessa Nota Técnica. Com essa padronização, tem-se que, em um futuro próximo, praticamente todas as arquiteturas de rede desaparecerão, capacitando os computadores de um fabricante a se comunicarem com computadores de outros fornecedores, sem quaisquer problemas de compatibilidade, estimulando ainda mais o uso de redes de computadores [1].

2. Estrutura de uma Rede

Uma rede de computadores é formada por um conjunto de computadores autônomos interconectados. Dizemos que computadores são autônomos quando não há relação de mestre/escravo entre eles, ou seja, um computador não pode controlar, ligar ou desligar um outro computador qualquer à sua revelia. Por interconectados, entende-se que eles são capazes de trocar informações entre si, sendo que essa conexão pode ser feita por meio de fios de cobre, por *lasers*, microondas ou até por satélites de comunicação [8]. É importante ressaltar que uma rede não precisa ser constituída unicamente por computadores, sendo comum a presença de impressoras, *scanners* e outros dispositivos de rede.

2.1. Objetivos de uma Rede

Quando uma rede de computadores qualquer é construída, existem alguns objetivos a serem alcançados. Nesse item, abordaremos alguns desses objetivos, para assim mostrarmos para que as redes de computadores podem ser usadas.

Podemos citar como primeiro objetivo o compartilhamento de recursos; isto é, todos os programas, dados e equipamentos devem estar disponíveis para qualquer um na rede, independentemente da localização física do recurso e do usuário, esse último sendo chamado também de *host*.

Como segundo objetivo, temos a necessidade de uma alta confiabilidade, tendo-se fontes de suprimento alternativas. Isto significa, por exemplo, que todos os arquivos existentes em uma rede

poderiam ser vistos em quantas máquinas quiséssemos, de maneira que, se uma delas apresentasse algum problema de hardware, as outras cópias poderiam ser usadas.

Um terceiro objetivo é a economia, uma vez que computadores de pequeno porte, os mais utilizados atualmente na construção de modelos de redes, têm uma relação custo/desempenho muito melhor do que os computadores de grande porte. Isto se explica pelo fato de que um *mainframe*, apesar de ser aproximadamente dez vezes mais rápido do que o mais rápido microprocessador de um *chip*, custa muitas vezes mais. Esse desequilíbrio levou muitos projetistas de sistemas a construírem redes constituídas de computadores pessoais potentes, havendo um por usuário, com os dados guardados em uma ou mais máquinas servidoras de arquivos. Esse último objetivo leva à existência de redes com muitos computadores localizados em um mesmo prédio, sendo esse tipo de estrutura conhecida como rede local [1].

De acordo com a localização dos módulos processadores, isto é, dos computadores que a compõem, uma rede pode ser classificada em diferentes tipos. No item seguinte, temos alguns exemplos desses modelos de rede.

2.2. Classificação

- Redes Locais:

São redes em que os computadores localizam-se em uma faixa que varia de poucos metros até alguns quilômetros. É conhecida como *Local Area Network* – LAN.

- Redes Metropolitanas:

São redes de computadores onde a distância entre as máquinas começa a atingir distâncias metropolitanas, sendo conhecida como *Metropolitan Area Network* – MAN.

- Redes Geograficamente Distribuídas:

Também conhecida como *Wide Area Network* – WAN, esse tipo de rede apareceu devido à necessidade de compartilhamento de recursos entre usuários geograficamente dispersos, sendo que seu custo de comunicação é elevado, uma vez que ela trabalha com enlaces de microondas, satélites, etc. Devido a isto, elas são geralmente de propriedade pública [8].

Abaixo, temos uma tabela em que são mostrados os tipos de rede explicados, seguidos dos valores das distâncias entre os seus módulos processadores e a respectiva localização entre eles.

Distância entre Módulos Processadores	Localização entre Módulos Processadores	Tipo de Rede
10 m	Sala	Local
100 m	Prédio	Local
1 km	Campus	Local
10 km	Cidade	Metropolitana
100 km	País	Metropolitana
1000 km	Continente	Geograficamente Distribuída
10000 km	Planeta	Geograficamente Distribuída

Hoje em dia, as modernas redes de computadores são projetadas de forma altamente estruturada. A seguir, serão mostradas as técnicas de utilização de camadas e protocolos.

2.3. Camadas

Para reduzir a complexidade de seu projeto, as redes de computadores são, em sua maioria, organizadas em camadas ou níveis, que representam diferentes níveis de abstração com funções definidas. Temos que cada camada é construída sobre aquela que a antecede. O número de camadas, o nome, o conteúdo e a função de cada camada diferem de uma rede para outra. Entretanto, em qualquer rede, o objetivo de cada camada é oferecer determinados serviços às camadas superiores, protegendo essas camadas dos detalhes de como os serviços oferecidos são de fato implementados, além de também receberem serviços das camadas inferiores.

Imaginemos, como exemplo, uma camada *n* qualquer em um computador. Essa camada estabelece comunicação com a camada *n* em outro computador, utilizando o devido protocolo, que será explicado posteriormente. Na verdade, nenhum dado é transferido diretamente da camada *n* de uma máquina para a outra. O que ocorre de fato é uma transferência de dados e informações de controle dessa camada para a camada imediatamente abaixo, até que o nível mais baixo tenha sido alcançado. Esse processo ocorre de camada para camada, até que a última camada (no caso, o nível mais baixo), seja alcançada, sendo que abaixo dela encontra-se o meio físico de comunicação, através do qual a comunicação entre os computadores de fato ocorre. Temos ainda que ao dizermos que houve uma comunicação da camada *n* com a camada *n*, essa comunicação é denominada de virtual, enquanto que a comunicação no meio físico é denominada de real [1].

2.4. Protocolos

Em uma rede de computadores, as regras e convenções utilizadas na conversação de uma camada *n* em uma máquina com a camada *n* em outra são usualmente chamadas de protocolos [1]. Um

protocolo é um sistema de comunicação de dados que permite que vários dispositivos de uma rede interajam entre si, sendo a sua principal característica a capacidade de permitir a comunicação entre computadores que diferem, entre outras coisas, nos seus sistemas operacionais, nas suas CPU's, nas interfaces de rede, etc [10].

Agora que já discutimos superficialmente como funcionam redes organizadas em camadas, vamos examinar o conjunto de camadas que é utilizado como referência na estruturação de uma rede.

3. O Modelo de Referência OSI

Esse item abordará o modelo de referência que é mais difundido na área de redes de computadores: o modelo OSI. Serão vistas as suas características básicas, como por exemplo o tipo de comunicação adotado por ele, o formato da comunicação dos dados transmitidos pela rede, entre outras.

3.1. Introdução

O transporte de informações entre computadores de diferentes tipos é uma capacidade que mostra-se extremamente importante. No início dos anos 80, a ISO (*International Organization of Standardization*) reconheceu a necessidade de um modelo de rede que auxiliasse na criação de implementações de redes interoperacionais. Em conseqüência dessa necessidade, surgiu o Modelo de Referência OSI ISO para Interconexão de Sistemas Abertos, que é chamado, por brevidade, de modelo OSI. Esse modelo foi um primeiro passo para a padronização internacional dos diversos protocolos existentes hoje em dia [1].

3.2. Comunicação Hierárquica

O modelo de referência OSI divide o problema de transmissão de informações entre computadores de uma rede em 7 problemas menores e melhor gerenciáveis. Cada uma dessas 7 áreas de problemas é resolvida por uma camada do modelo OSI. Os princípios utilizados para chegar-se às 7 camadas são:

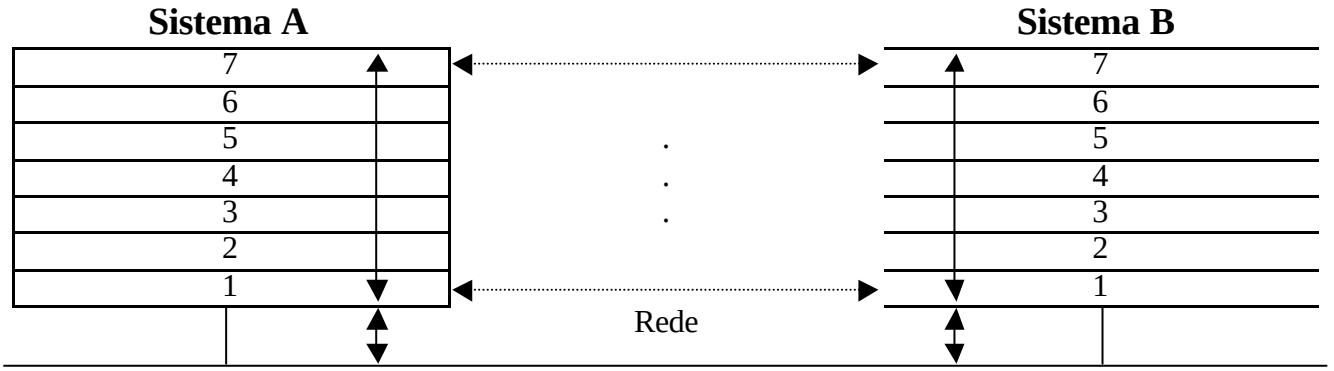
- Uma camada deve ser criada onde é necessário um nível de abstração diferente.
- Cada camada deve desempenhar uma função bem definida.

- A função de cada camada deve ser definida tendo em vista a definição de protocolos padrões internacionais.
- As fronteiras entre as camadas devem ser escolhidas de forma a minimizar o fluxo de informações através das interfaces.

O número de camadas deve ser grande o suficiente para que não seja preciso agrupar funções em uma mesma camada por necessidade, e pequeno o suficiente para que a arquitetura fique manejável.

A maioria dos dispositivos de uma rede implementam todas as 7 camadas. Entretanto, para tornar mais eficiente as operações, algumas implementações de rede omitem uma ou mais camadas. As 2 últimas camadas do modelo OSI são implementadas com software e hardware; as 5 camadas superiores são geralmente implementadas em software.

Como exemplo do tipo de comunicação realizado pelo modelo OSI, vejamos a figura abaixo. Considere que o sistema A tem informações a serem transmitidas para o sistema B. O programa de aplicação do sistema A comunica-se com a camada 7 (a camada de topo) do sistema A, que se comunica com a camada 6 do sistema A, que se comunica com a camada 5, até que a camada 1 desse sistema seja carregada. A camada 1 preocupa-se em colocar e retirar informações do meio físico da rede. Após as informações terem atravessado esse meio, elas ascendem pelas camadas do sistema B na ordem inversa (primeiro a camada 1, depois a camada 2, etc.), até que finalmente carreguem o programa de aplicação desse sistema B. Esse processo também é válido para o caso em que as informações são transmitidas do sistema B para o sistema A.



Apesar de cada uma das camadas do sistema A comunicarem-se com a camada adjacente desse mesmo sistema, o verdadeiro objetivo delas é a comunicação com as suas camadas iguais no sistema B (ver setas tracejadas na figura). Isto é, o objetivo primário da camada 1 do sistema A é comunicar-se com a camada 1 do sistema B; a camada 2 do sistema A se comunica com a camada 2 do sistema B e assim por diante. Isto é necessário porque cada camada em um sistema tem certas tarefas que devem ser executadas, e para ocorrer essa execução, a camada precisa comunicar-se com a sua camada igual do outro sistema [7].

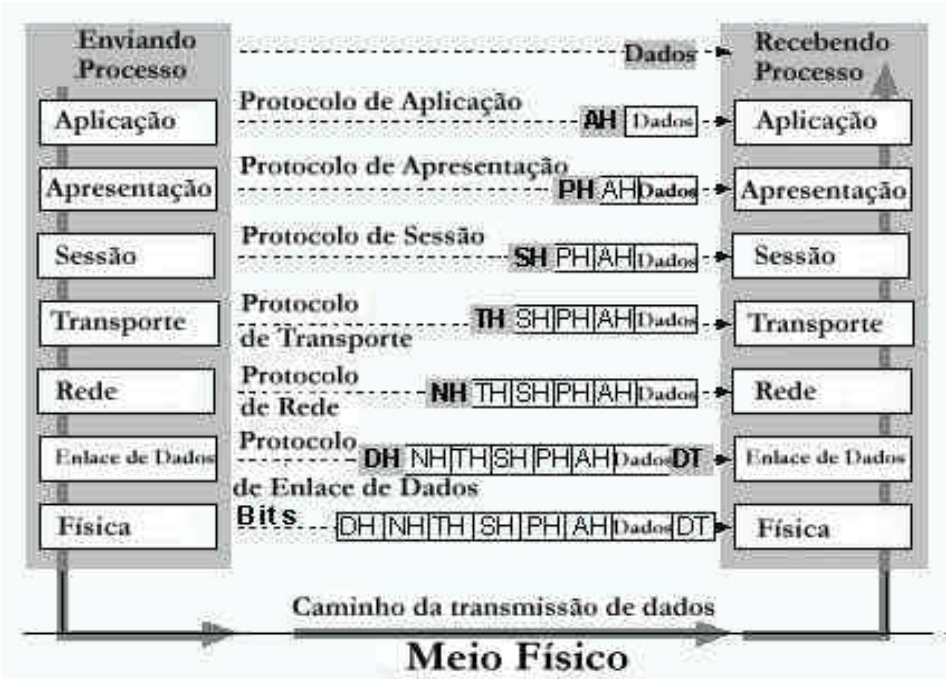
3.3. Formato das Informações

Como a camada 4 no sistema B sabe o que a camada 4 no sistema A quer? As especificações da camada 4 requeridas são carregadas como uma informação de controle, que é transmitida entre camadas iguais em um bloco chamado de cabeçalho (*header*), que é anexado na informação de aplicação atual. Consideremos, por exemplo, que o sistema A deseja enviar o seguinte texto, no caso chamado de dado ou informação, para o sistema B.

"O pequeno gato cinza subiu no muro para tentar apanhar o pássaro vermelho."

Esse texto é transmitido do programa de aplicações do sistema A para a sua camada de topo. A camada de aplicações do sistema A deve comunicar certas informações para a mesma camada no sistema B. Sendo assim, anexa-se essa informação de controle, na forma de um cabeçalho codificado, no texto atual, para então esse ser transmitido. Essa unidade de informação é transmitida da camada 6 no sistema A, que pode anexar sua própria informação de controle. Com isso, temos que a unidade de informação cresce em tamanho a medida que é transmitida através das camadas até alcançar a rede, onde o texto original, e todas as suas informações de controle associadas, trafegam até o sistema B, onde é absorvido pela camada 1 desse mesmo sistema. A camada 1 no sistema B abre o cabeçalho de camada 1, o lê, e então sabe como processar a unidade de informação. A reduzida unidade de informação é então transmitida para a camada 2, que abre o cabeçalho de camada 2, analisa esse cabeçalho para saber as ações que a camada 2 deve tomar, e passa adiante. Quando a unidade de informação finalmente alcança o programa de aplicações no sistema B, ela simplesmente contém o texto original.

É importante saber que o conceito de cabeçalho e dados é relativo. Ele depende da perspectiva da camada analisando a unidade de informação. Por exemplo, para a camada 3, uma unidade de informação consiste de um cabeçalho de camada 3 e os dados que o seguem. Entretanto, os dados da camada 3 podem potencialmente conter cabeçalhos das camadas 4, 5, 6 e 7. Além disso, o cabeçalho de camada 3 é simplesmente um dado para a camada 2. Esse conceito fica melhor ilustrado através da figura representada abaixo. Finalmente, não são todas as camadas que precisam de cabeçalhos anexados. Algumas camadas simplesmente realizam uma transformação no dado atual que elas recebem, para então torná-lo legível para as suas camadas adjacentes [7].



3.4. Questões de Compatibilidade

O modelo de referência OSI não é uma implementação de rede. Ao invés disto, ele especifica as funções de cada camada. Nós temos que um projetista de rede qualquer pode construir uma implementação de protocolo, baseando-se em uma dada especificação, sendo que, a não ser que essa especificação seja extremamente compreensível, as diferentes implementações que forem construídas baseando-se nela serão diferentes, pelo menos, em pequenos detalhes.

Em parte, essas diferenças existem devido à incapacidade de qualquer especificação de considerar todos os possíveis detalhes de implementação. Além disso, diferentes implementadores irão interpretar, sem dúvida, os protocolos de diferentes maneiras e, devido à isto, ocorrerão inevitáveis erros de implementação, que resultarão em implementações diferentes e, consequentemente, em diferenças nas execuções de tarefas [7].

3.5. As Camadas do Modelo OSI

Nesse item, falaremos à respeito das camadas individuais do modelo OSI e suas funções. Cada uma dessas camadas tem um grupo predeterminado de funções que devem ser executadas para a comunicação ocorrer. Abaixo, temos uma tabela que mostra a disposição das camadas em questão.

Número das Camadas	**Funcionalidade**
7	Aplicações
6	Apresentação
5	Sessão
4	Transporte
3	Rede
2	*Link* de Dados
1	Física

Devemos observar que o modelo OSI não é uma arquitetura de rede, já que uma arquitetura de rede é um conjunto de camadas e protocolos e o modelo OSI não especifica exatamente os serviços e protocolos a serem usados em cada camada. No entanto, a ISO também já produziu padrões para todas as camadas, embora estritamente falando eles não façam parte do modelo OSI. Cada um foi publicado como um padrão internacional separado [1].

Agora, serão explicadas com algum detalhe as funções de cada uma das 7 camadas do modelo OSI.

3.5.1. A Camada Física

A camada física define as especificações elétricas, mecânicas, procedimentais e funcionais para ativação, manutenção e desativação do *link* físico entre sistemas locais [7], isto é, ela lida com a transmissão pura de bits através de um canal de comunicação. As questões de projeto são concernentes à garantia de que, quando um dado transmite um bit 1, esse seja recebido como um bit 1 do outro lado, e não como um bit 0. As funções típicas aqui são: os níveis de voltagem, ou seja, quantos volts devem ser usados para representar um 1 e quantos um 0; o tempo de mudança de voltagens; a taxa de velocidade do dado físico, que significa quantos microssegundos dura um bit; as distâncias máximas de transmissão e se essa transmissão pode ocorrer em ambos os sentidos simultaneamente; como a conexão inicial é estabelecida e como é desfeita quando os dois computadores terminam a transmissão de dados; quantos pinos o conector de rede possui e para que serve cada pino e outras similares. Esses atributos são definidos pelas especificações da camada física, sendo o seu projeto considerado, apropriadamente, como parte dos domínios do engenheiro eletrônico [1].

3.5.2. Camada de *Link* de Dados

A camada de *link* (formalmente conhecida como sendo a camada de *link* de dados) proporciona o tráfego confiável de dados através de um *link* físico. Temos então que a camada de *link* está preocupada com o endereçamento físico (endereço de MAC), com a topologia de rede, com a forma com que os sistemas locais irão usar o *link* da rede, a notificação de erros, a entrega ordenada de quadros (*frames*) e o controle de fluxo [7]. Um quadro, ou *frame*, é uma unidade de informação lógica que representa a estrutura exata de dados transmitidos fisicamente através do fio de conexão ou através de outro meio [3].

A tarefa principal dessa camada é utilizar-se da facilidade de transmissão de dados brutos, transformando-a em uma linha que pareça à camada de rede, a camada superior a essa, ser livre de erros de transmissão. Ela realiza essa tarefa fazendo com que o transmissor fragmente os dados de entrada em quadros, em geral com algumas centenas de bytes, transmita-os seqüencialmente e processe os quadros de confirmação mandados de volta pelo receptor.

Uma vez que a camada física meramente aceita uma seqüência de bits sem se importar com o significado ou a estrutura, cabe à camada de *link* de dados criar e reconhecer os limites dos quadros. Isto pode ser conseguido anexando-se padrões de bits especiais ao começo e ao fim do quadro. Como esses padrões de bits podem ocorrer acidentalmente nos dados, certas precauções especiais devem ser tomadas para evitar problemas.

Uma ruído qualquer na linha pode destruir completamente um quadro. Nesse caso, o software da camada de *link* de dados do transmissor deve retransmitir esse quadro. Entretanto, múltiplas transmissões de um mesmo quadro possibilitam a ocorrência de duplicação. Temos que um quadro duplicado poderia ser transmitido, por exemplo, se o quadro de confirmação do receptor para o transmissor fosse destruído. É tarefa então da camada de *link* resolver esse tipo de problema, causado por quadros danificados, perdidos ou duplicados. Ela pode oferecer várias classes de serviço diferentes à camada de rede.

Uma outra questão relacionada com a camada de *link* de dados, e com as demais camadas superiores, é de que forma pode-se impedir que um transmissor rápido afogue com dados transmitidos um receptor lento. A solução utilizada para isso é o emprego de algum mecanismo regulador de tráfego, a fim de permitir ao transmissor saber quanto espaço em *buffer* o receptor tem no momento. Freqüentemente, por conveniência, essa regulação do fluxo e o tratamento de erros são integrados [1].

3.5.3. Camada de Rede

A camada de rede é uma camada complexa que possibilita conectividade e seleção de caminhos entre dois sistemas locais que podem ser localizados geograficamente dispersos em sub-redes, sendo que o controle da operação dessas sub-redes é responsabilidade dessa camada. Uma sub-rede é um único cabo de rede, no caso de um cabeamento coaxial, sendo nesse caso também chamado de segmento; ou é um grupo de dispositivos agrupados por meio de software, no caso de um cabeamento de par trançado [7].

Uma questão de projeto fundamental da camada de rede é determinar como os pacotes são roteados da origem para o destino. As rotas podem ser baseadas em tabelas estáticas embutidas e raramente modificadas na rede. Essas rotas também poderiam ser determinadas no início de cada conversa, como, por exemplo, uma sessão de terminal, ou, finalmente, altamente dinâmicas, sendo então determinadas novamente para cada pacote e refletindo a carga atual da rede.

Temos que se vários pacotes estiverem presentes na sub-rede ao mesmo tempo, eles ficarão uns nos caminhos dos outros, criando congestionamentos, sendo a camada de rede responsável então pelo controle desses congestionamentos.

Podem aparecer vários problemas quando um pacote deve viajar de uma rede a outra para chegar ao seu destino, como, por exemplo, o endereçamento utilizado na segunda rede ser diferente do na primeira. Nesse exemplo, temos que a segunda rede pode até mesmo não aceitar o pacote por ser muito grande, ou ainda, os protocolos são diferentes, etc. É tarefa da camada de rede superar esses tipos de problemas para permitir então a interconexão entre redes heterogêneas [1].

3.5.4. Camada de Transporte

A função básica da camada de transporte é aceitar dados da camada de sessão, dividi-los se necessário em unidades menores, passá-las à camada de rede e garantir que os pedaços cheguem corretamente ao outro lado [1]. O limite entre a camada de sessão e a camada de transporte pode ser visto como o limite entre os protocolos da camada de aplicações e os protocolos das camadas inferiores. Enquanto as camadas de aplicação, de apresentação e de sessão estão preocupadas com questões de aplicação, as quatro camadas inferiores estão preocupadas com questões de transporte de dados. Esse serviço é executado pela camada de transporte, que protege as camadas superiores dos detalhes de implementação dos dados. Especificamente, questões como, por exemplo, como o transporte de dados de confiança através da rede é executado são de interesse dessa camada. Em relação à confiança nos serviços, temos que a camada de transporte oferece mecanismos para estabelecimento, manutenção e terminação ordenada de circuitos virtuais, para detecção e correção de falhas no transporte e controle do fluxo de informações, prevenindo assim que um sistema envie mensagens em uma taxa maior do que a capacidade que o receptor tem de recebê-las [7].

A camada de transporte também determina que tipo de serviço é oferecido à camada de sessão e, em última análise, aos usuários da rede. Dentre os tipos de conexão de transporte, o mais popular é um canal ponto a ponto livre de erros, que entrega as mensagens na ordem em que as recebeu. Outros tipos possíveis de transporte são a difusão de mensagens para múltiplos destinos e o transporte de mensagens isoladas sem garantias da ordem de entrega. Temos que o tipo de serviço é determinado quando a conexão é estabelecida.

Podemos dizer que a camada de transporte é uma camada origem-destino ou camada fim a fim, isto é, um programa no computador de origem conversa com um programa similar no computador de destino, usando os cabeçalhos das mensagens e mensagens de controle. Nas camadas inferiores, os

protocolos são entre cada computador e seus vizinhos imediatos, e não entre os computadores finais de origem e de destino, que podem estar separados por vários IMP's. Podemos então dizer que as camadas de transporte, de sessão, de apresentação e de aplicações são fim a fim, enquanto que as camadas física, de *link* de dados e de rede são encadeadas.

Muitos *hosts* são multiprogramados, o que implica que múltiplas conexões estarão entrando e saindo de cada *host*. Deve haver algum modo de determinar que mensagem pertence a que conexão. O cabeçalho de transporte é um lugar onde essa informação poderia ser colocada.

Além de multiplexar vários fluxos de mensagens em um canal, a camada de transporte deve cuidar do estabelecimento e encerramento de conexões através da rede. Para isto, é necessário um mecanismo de nomeação para que um processo em um computador tenha como descrever com quem deseja conversar. Além disso, deve haver também algum mecanismo para regular o fluxo de informações de forma que um *host* rápido não atropele um outro mais lento. Por último, temos ainda que o controle de fluxo entre *hosts* é diferente do controle de fluxo entre IMP's [1].

3.5.5. Camada de Sessão

Como o próprio nome indica, a camada de sessão estabelece, gerencia e termina sessões entre aplicativos [7]. Em outras palavras, ela permite a usuários em computadores diferentes estabelecerem sessões entre eles. Uma sessão permite o transporte ordinário de dados, tal como a camada de transporte, mas também provê serviços aperfeiçoados que podem ser úteis para algumas aplicações.

A camada de sessão tem, como um dos seus serviços, a função de gerenciar o controle de diálogos. As sessões podem permitir o tráfego fluindo em ambos os sentidos ao mesmo tempo, ou em apenas um sentido de cada vez. Se o tráfego só pode ir em um sentido por vez, a camada de sessão pode ajudar a acompanhar de quem é a vez de transmitir.

Um outro serviço executado por essa camada é o gerenciamento de *tokens*, isto é, sinais. Em alguns protocolos é essencial que ambos os lados não tentem fazer uma operação ao mesmo tempo. Para controlar essa situação, a camada de sessão provê sinais que podem ser trocados. Somente o proprietário do *token* poderá transmitir dados.

Como terceiro exemplo de serviço da camada de sessão, temos a sincronização. Consideremos como exemplo os problemas que podem ocorrer quando tenta-se fazer uma transferência de arquivos que leva duas horas entre dois computadores em uma rede com tempo médio entre falhas de uma hora. Teríamos que, depois que cada transferência fosse abortada, o processo inteiro teria de recomeçar e provavelmente falharia novamente após uma hora, não permitindo que a transferência seja completada. Para eliminar esse problema, a camada de sessão fornece meio de inserir *checkpoints* no fluxo de dados, isto é, existem pontos no decorrer desse fluxo onde os dados são verificados. Com isso, após uma falha, somente os dados após o último *checkpoint* devem ser repetidos [1].

3.5.6. Camada de Apresentação

A camada de apresentação assegura que as informações enviadas pela camada de aplicações de um sistema sejam reconhecidas pela camada de aplicações de outro sistema. Se for necessário, essa camada traduz formatos de representação entre múltiplos dados utilizando um único formato de representação de dados.

A camada de apresentação preocupa-se não somente com o formato e a representação dos dados, mas também com as suas estruturas usadas por programas. Portanto, além de modificar o formato dos dados, quando necessário, ela também negocia a sintaxe da transferência de dados da camada de aplicações [7].

Um típico exemplo de um serviço executado pela camada de apresentação é a codificação de dados, em alguma forma padrão estabelecida previamente. A maioria dos programas do usuário não troca cadeias aleatórias de bits. Eles trocam itens como, por exemplo, nomes de pessoas, datas, quantias em dinheiro e faturas, que são representados como cadeias de caracteres, inteiros, números em ponto flutuante e estruturas de dados compostas de vários itens mais simples. Computadores diferentes podem ter codificações diferentes para representar caracteres como, por exemplo, ASCII e EBCDIC; inteiros, como complemento de um e complemento de dois, e assim por diante. Para possibilitar a comunicação entre computadores com representações diferentes, as estruturas de dados que devem ser trocadas podem ser definidas de forma abstrata, junto com uma codificação padrão, para ser utilizada então "na linha". Temos com isso que a camada de apresentação tem a tarefa de gerenciar essas estruturas abstratas de dados e convertê-las da representação utilizada dentro do computador para a representação padrão da rede, além de também se preocupar com outros aspectos da representação da informação como, por exemplo, a compressão de dados, que pode ser utilizada para reduzir o número de bits que devem ser transmitidos, e a criptografia, que é freqüentemente requerida para privacidade e autenticação [1].

3.5.7. Camada de Aplicações

A camada de aplicações é a camada do modelo OSI que mais aproxima-se do usuário. Ela é diferente das camadas anteriores, uma vez que não pressa serviços para qualquer uma dessas, mas preferivelmente para processos de aplicação como, por exemplo, o correio eletrônico e a consulta a diretórios. Temos que ela identifica e estabelece a disponibilidade de pares desejados de comunicação, sincroniza aplicações cooperadas e estabelece permissão nos procedimentos de recuperação de erros e controle da integridade de dados. Adicionalmente, essa camada determina se existem recursos suficientes para a comunicação desejada [7].

Essa camada contém uma variedade de protocolos comumente necessários. Como exemplo, temos que existem vários tipos de terminais diferentes no mercado, o que causa grandes problemas de

compatibilidade. Considere a dificuldade pela qual um editor de tela passa para funcionar através de uma rede com muitos tipos diferentes de terminais, cada qual com diferentes *layouts* de tela, seqüências de escape para inserção e deleção de texto, movimentos do cursor, etc.

Uma forma de resolver esse problema é definir um terminal visual de rede abstrato, de tal forma que editores e outros programas capazes de lidar com ele possam ser escritos. Para manipular cada tipo de terminal, deve-se escrever um trecho de software para mapear as funções do terminal virtual de rede para o terminal real. Por exemplo, quando o editor move o cursor do terminal visual para o canto superior esquerdo da tela, esse software deve dar a seqüência de comandos apropriada para que o terminal real desloque o cursor para lá. Todo o software para o terminal visual está na camada de aplicações.

Uma outra função da camada de aplicações é a transferência de arquivos. Sistemas de arquivos diferentes têm convenções de nomenclatura diferentes, formas diferentes de representar linhas de texto, e assim por diante, sendo que essas e outras incompatibilidades apresentam-se na transferência de arquivos entre sistemas diferentes, ficando então a cargo da camada de aplicações solucioná-las [1].

4. Os protocolos

Agora que já estudamos as diferentes funções de cada uma das 7 camadas do modelo OSI, iniciaremos um estudo mais detalhado dos protocolos constituintes dessas camadas. Tendo como base a rede do CBPF, iremos estudar os principais protocolos utilizados na Internet, em uma rede *Novell* e em uma rede NT.

Como já sabemos, um protocolo é um sistema de comunicação de dados que permite que 2 computadores diferentes troquem informações. Começaremos falando à respeito dos protocolos utilizados na Internet, explicando em detalhes quais as funções de cada um deles. Inicialmente, mostramos uma tabela que mostra a relação entre as camadas do modelo OSI com os protocolos utilizados na Internet.

Modelo OSI	**INTERNET**	
Camada de Aplicações	*TELNET* / FTP / SMTP	NFS / SNMP / DNS
Camada de Apresentação		
Camada de Sessão		

Camada de Transporte	TCP	UDP
Camada de Rede	IP	
Camada de *Link* de Dados		
Camada Física		

4.1. *Internet Protocol* (IP)

O protocolo IP é o protocolo da camada de rede responsável pelo transporte de datagramas (blocos de dados) através da Internet [3]. Isto é, ele é um protocolo de datagrama orientado, sendo que cada pacote é tratado independentemente. Isto significa que cada pacote deve conter informações completas de endereçamento.

O protocolo IP não verifica se um pacote alcançou o seu destino, além de não executar nenhuma ação de correção caso ele não o tenha alcançado. Temos ainda que esse protocolo também não verifica o conteúdo de um pacote, mas somente o seu cabeçalho.

O protocolo IP oferece diferentes serviços, que estão listados abaixo:

- Endereçamento:

Os cabeçalhos IP contém endereços de 32 bits, que identificam os *hosts* transmissores e receptores de informações. Esses endereços são utilizados por roteadores intermediários para selecionar um caminho apropriado para o pacote através da rede.

- Fragmentação:

Datagramas IP podem ser divididos, ou fragmentados, em pequenos pacotes. Isto permite que um pacote muito grande viaje através de uma rede que suporte apenas pequenos pacotes. O protocolo IP fragmenta e remonta esses pacotes de forma transparente.

- Término de Pacotes:

Cada pacote IP contém um campo TTL (*Time To Live*), que é um campo que especifica quantos *hops* a mais um pacote pode dar antes de ser descartado ou retornado. Um *hop* é uma conexão intermediária em uma seqüência de conexões que une 2 dispositivos de rede. Temos que esse campo diminui toda vez que um roteador lida com o pacote, sendo que o pacote é descartado quando o TTL alcança o zero, prevenindo assim os pacotes de circularem eternamente e, consequentemente, inundarem a rede.

- Tipo de Serviço:

O IP suporta priorização de tráfego, permitindo que pacotes sejam classificados com algum tipo de serviço abstrato.

- Opções:

O protocolo IP possui diversas características opcionais, como por exemplo, permitir que alguém que esteja enviando um pacote qualquer determine condições no caminho que esse pacote utiliza através da rede, traçar a rota utilizada pelo pacote e classificar os pacotes com características seguras [11].

4.1.1. Endereço IP

O endereço IP é uma identificação para um computador ou um dispositivo qualquer de uma rede TCP/IP [12]. O TCP/IP (*Transmission Control Protocol / Internet Protocol*) é um conjunto de protocolos de comunicação. As informações enviadas pela Internet são dependentes do TCP/IP, fazendo com que ele seja utilizado como um protocolo primário de rede na Internet [10].

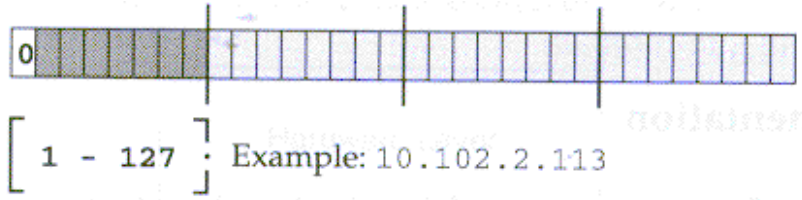
Temos que as redes que utilizam o protocolo TCP/IP roteam mensagens baseadas no endereço IP de destino. O formato de um endereço IP é o de um endereço numérico de 32 bits escritos como 4 números, também conhecidos como octetos, que são separados por pontos. Temos que cada um desses quatro octetos representam campos de 8 bits.

Com uma rede isolada, pode-se determinar um endereço IP qualquer, respeitando o fato de que cada endereço deve ser único. Contudo, registrar uma rede privada à Internet requer endereços IP registrados, chamados endereços da Internet, para evitar possíveis duplicações.

Os 4 números ou octetos de um endereço IP são usados de maneiras diferentes para identificar uma rede particular e um *host* qualquer nessa rede [12]. Classifica-se endereços da Internet registrados em 4 classes:

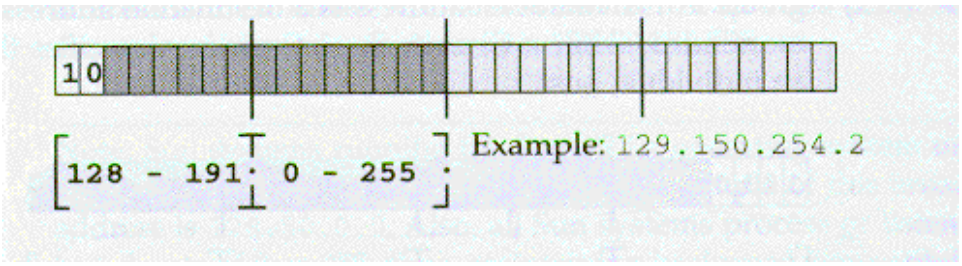
- Classe A:

Suporta 16 milhões de *hosts* em cada uma das suas 127 redes. Nessa classe de rede, temos que se o primeiro bit do seu endereço IP for 0, então os próximos 7 bits serão destinados ao número de rede e os 24 bits (3 octetos) restantes, aos números de dispositivo. Abaixo, temos uma representação da divisão em octetos dessa classe.



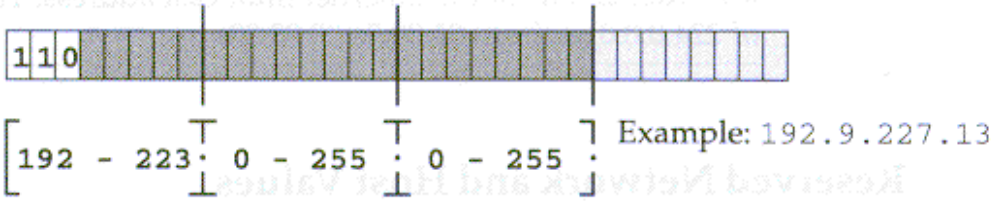
- Classe B:

Suporta 65.000 *hosts* em cada uma das suas 16.000 redes. Aqui, temos que se os 2 primeiros bits forem 1 e 0, respectivamente, então os próximos 14 bits serão destinados ao número da rede e os 16 bits (2 octetos) restantes aos números de dispositivos. A representação relativa a essa classe encontra-se abaixo.



- Classe C:

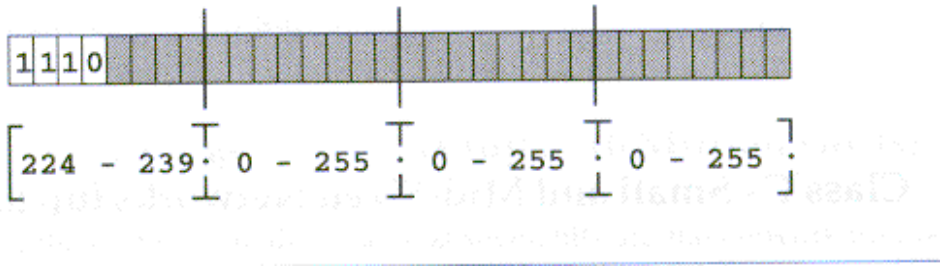
Suporta 254 *hosts* em cada um dos seus 2 milhões de redes. Se os seus 3 primeiros bits forem 1, 1 e 0, respectivamente, então os próximos 21 bits serão destinados ao número de rede e os 8 bits (1 octeto) restantes aos números de dispositivos. Abaixo, podemos ver a sua representação.



- Classe D:

Se os quatro primeiros bits forem 1, 1, 1 e 0, respectivamente, então o valor do primeiro octeto pode variar entre 224 e 239 e dizemos que esse número é um endereço *multicast*. Os próximos 28 bits compõem um número de identificação de grupo para um específico grupo *multicast*. Podemos concluir então que um endereço IP *multicast* é um endereço destinado a um ou mais *hosts* ou dispositivos, ao

contrário dos endereços classe A, B e C, que especificam o endereço de um *host* ou dispositivo individual. A divisão de octetos da classe D está representada abaixo [3].



4.2. *Transmission Control Protocol* (TCP)

O protocolo TCP é o protocolo da camada de transporte, entre um par de aplicações, responsável por uma transmissão de dados, com conexão orientada, de confiança [3]. Esse protocolo provê serviços de *full-duplex*, que será definido posteriormente, reconhecimento e controle de fluxo para os protocolos das camadas superiores [7].

O TCP adiciona uma grande quantidade de funcionalidade ao serviço IP, como podemos verificar abaixo:

- Canais:

Os dados transmitidos pelo TCP são organizados como um canal de bytes, muito parecido com um arquivo. A natureza do datagrama da rede é escondida. Um mecanismo, conhecido como *Urgent Pointer*, existe para apontar na direção do primeiro dado de bytes no pacote.

- Transmissão de Segurança:

Seqüências de números são utilizadas para coordenar quais dados são transmitidos e recebidos. O TCP organiza-se para retransmitir os dados se for determinado que um dado qualquer foi perdido.

- Adaptação de Rede:

O TCP aprende dinamicamente as características de atraso de uma rede e ajusta a sua operação para maximizar a saída (*throughput*) da rede sem sobrecarregá-la.

- Controle de Fluxo:

O TCP gerencia os dados do *buffer* e coordena o tráfego de forma que os seus *buffers* nunca inundem. Isto significa que remetentes muito rápidos serão travados periodicamente para conservar receptores lentos [13].

4.2.1. Operação *Full-Duplex*

Independentemente da aplicação particular, o TCP quase sempre opera em *full-duplex*, isto é, os dados são transmitidos em ambos os sentidos simultaneamente. Algumas vezes, é útil pensar em uma sessão TCP como 2 canais de bytes independentes, viajando em sentidos opostos. Não existe nenhum mecanismo TCP que associe os dados nos canais de bytes direto (para dados) e inverso.(para confirmações). O TCP só pode exibir um comportamento assimétrico durante conexões de início e fechamento de seqüências, como por exemplo, uma transferência de dados no sentido direto mas não no inverso, e vice-versa [13].

4.2.2. Seqüência de Números

O protocolo TCP utiliza uma seqüência de números de 32 bits que contabiliza os bytes em um canal de bytes. Cada pacote TCP contém a seqüência inicial de números do dado naquele pacote, e a seqüência de números, chamada de número de reconhecimento (*acknowledgement number*), do último byte recebido do par remoto. As seqüências de números dianteira e invertida são completamente independentes, e cada par TCP deve localizar em ambos sua própria seqüência de numeração, sendo essa numeração usada pelo par remoto.

Temos que o TCP utiliza bandeiras de controle para gerenciar uma conexão [13]. Uma bandeira é uma marcação especial que indica se um pedaço do dado transmitido está danificado [26]. Algumas dessas bandeiras pertencem a um único pacote, como a bandeira URG indicando os dados válidos no campo do *Urgent Pointer*. Já outros dois tipos de bandeiras, SYN e FIN, requerem transmissão de confiança, pois elas marcam o início e o fim dos canais de bytes. Com o intuito de assegurar a transmissão de confiança dessas 2 bandeiras, elas são apontadas como pontos no espaço da seqüência de números. Cada bandeira ocupa um único byte [13].

4.2.3. *Window Size* e *Buffering*

Cada computador encontrado em um extremo de uma conexão TCP terá uma área de *buffer* para carregar dados que são transmitidos pela rede antes que a aplicação esteja pronta para ler os dados. Isto permite que ocorram transferências na rede enquanto as aplicações estão ocupadas com outros processamentos, melhoramentos, da performance total.

Para evitar inundamentos do *buffer*, o TCP estabelece um campo de *Window Size* em cada um dos pacotes que são transmitidos. Esse campo contém a quantidade de bytes que podem ser transmitidos para a área de *buffer*. Se esse número cair até o zero, o TCP remoto pode não mais enviar dados. Quando isto ocorre, deve-se esperar até que a área de *buffer* esteja abilitada novamente, recebendo então um pacote anunciando um campo de *Window Size* diferente de zero.

Algumas vezes, temos que a área de *buffer* é muito pequena. Isto ocorre quando o produto da largura de banda pelo atraso da rede (*bandwidth-delay product*) ultrapassa o tamanho do *buffer*. A solução mais simples é aumentar a área de *buffer*. Porém, em casos extremos, o próprio protocolo cria o congestionamento, uma vez que ele não suporta um *Window Size* grande o suficiente. Nessas condições, a rede é chamada de LFN (*Long Fat Network*) [13].

4.2.4. Estimativa de tempo *Round-Trip*

Quando um usuário transmite um pacote TCP, deve-se esperar um período de tempo para o reconhecimento desse pacote. Se a resposta não chegar no tempo esperado, o pacote é considerado como tendo sido perdido, e então os dados são retransmitidos pelo computador. Porém, quanto tempo espera-se pela resposta? Temos que, através de uma rede *Ethernet*, não mais que alguns microssegundos são necessários para uma resposta. Já no caso do tráfego fluindo através da extensa área da Internet, um segundo ou mais é um tempo razoável durante picos de tempo de utilização, e assim por diante; isto é, o tempo de espera aumenta de acordo com as circunstâncias, demostrando então que não existe uma resposta exata para essa questão.

Todas as implementações TCP modernas procuram responder a essa questão através do monitoramento da troca normal de pacotes de dados e do desenvolvimento de uma estimativa de quanto tempo deve ser considerado "muito longo". Esse processo é chamado de estimação de RTT (*Round Trip Time*). As estimativas do RTT são um dos mais importantes parâmetros de performance em uma troca TCP, especialmente quando você considera que em uma transmissão indefinidamente longa, todas as implementações TCP eventualmente perdem pacotes e os retransmitem, não importando a qualidade do *link*. Se a estimativa RTT for muita baixa, pacotes serão retransmitidos desnecessariamente; se for alta, a conexão pode ficar inativa enquanto o usuário espera o fim do processo [13].

4.3. *User Datagram Protocol* (UDP)

Assim como o TCP, o protocolo UDP é um protocolo da camada de transporte, que é usado para transferir dados entre agentes [3], em que um usuário pode enviar uma mensagem sem estabelecer uma conexão com o receptor, isto é, o usuário simplesmente põe a mensagem na rede com o endereço de destino e espera que essa chegue. Define-se agente como sendo um programa que executa informações agrupando ou processando tarefas no modo *background*, isto é, que executa diversas tarefas, ou programas, simultaneamente.

Os pacotes UDP são transmitidos da mesma forma que os pacotes IP, isto é, temos datagramas não-conectados que podem ser descartados antes de alcançarem seus respectivos alvos. O UDP mostra-se útil quando o protocolo TCP for muito complexo, muito lento ou simplesmente desnecessário. Temos ainda que o UDP apresenta algumas funções além das do IP, que estão listadas abaixo:

- Números de Porta (*Port Numbers*):

O UDP provê números de porta de 16 bits para permitir que vários processos utilizem os serviços do próprio UDP no mesmo *host*. O endereço UDP é a combinação de um endereço IP de 32 bits com os 16 bits de um número de porta.

- *Checksumming*:

Ao contrário do IP, o UDP verifica se os seus dados foram transmitidos corretamente, através do processo chamado de *checksum*, que consiste em um simples processo de verificação de erros, em que cada mensagem transmitida é acompanhada de um valor numérico baseado no número de bits contidos na mensagem. A estação receptora então aplica a mesma fórmula para a mensagem recebida, verificando se o valor numérico que acompanha o pacote é o mesmo. Se não for, o receptor pode assumir que a mensagem foi corrompida. Dessa forma, esse protocolo assegura a integridade dos dados. Um pacote que não passa pelo *checksum* simplesmente é descartado, sem que nenhuma ação adicional seja executada [14].

4.4. *Telnet*

O *Telnet* é um protocolo de terminal visual encontrado nas camadas superiores, que consiste em um programa utilizado em redes TCP/IP da mesma forma que a Internet. O programa *Telnet*, ao rodar em um computador, conecta-o em um servidor qualquer na rede. Pode-se então executar comandos através do programa *Telnet* como se o usuário estivesse trabalhando diretamente no console do servidor remoto. Isto permite que o usuário controle então esse servidor remoto e comunique-se com outros servidores da rede. Para iniciar uma sessão *Telnet*, o usuário deve conectar-se em um servidor utilizando um nome de usuário (*username*) e senha (*password*) válidos. O *Telnet* é o meio mais comum de controlar remotamente servidores *web* [15].

Para que o programa *Telnet* execute essa tarefa, temos que os equipamentos remotos devem possuir um sistema operacional multitarefa (executa mais de uma aplicação simultaneamente, compartilhando o tempo de CPU), que contenha mecanismos de autorização de acesso via sistema de contas, justificando assim a classificação do serviço *Telnet* como um serviço tipo *Remote Login* da Internet.

A execução do *logon* em um computador qualquer conectado na Internet via *Telnet* pode ser feita, como já comentado, conhecendo-se um nome de usuário e uma senha válidos na máquina remota. Isto é feito através de um programa cliente *Telnet*, que permite que o usuário interaja com o serviço Internet. Essa interação ocorre selecionado-se o equipamento onde se deseja executar uma dada aplicação. O servidor *Telnet* é acionado, enviando então um *prompt* para o estabelecimento da sessão, pedindo o nome do usuário e a senha necessários. Uma vez iniciada a sessão, o usuário poderá utilizar qualquer aplicação desse equipamento autorizado para essa sessão.

Existem sistemas que oferecem a variante de um *logon* como *guest*. Contudo, ao conectar-se dessa forma, a sua área de rede pode ser vista por qualquer um que venha a se conectar também como *guest*, enquanto que um *logon* específico, com um *username* e uma *password* válidos garante a segurança necessária para você trabalhar na sua área própria.

O *Telnet* é um serviço que pode ser muito útil, como por exemplo, ao disponibilizar serviços da Internet que não estejam disponíveis localmente, através da execução do lado cliente desses serviços em outros equipamentos. No entanto, a sua aplicação mais útil é a de permitir acesso remoto a qualquer *host* da Internet que disponibilize esse serviço. Acesso remoto é talvez a maior facilidade da Internet.

Abaixo, mostramos uma pequena tabela com alguns dos comandos mais comuns utilizados pelo programa *Telnet*.

Comandos	**Funções**
Open	Estabelece uma conexão *Telnet* com um *host* remoto.
Close	Termina uma conexão *Telnet*.
Quit	Fecha o *Telnet* corrente (se houver) e termina o *Telnet*.
z	Suspende o *Telnet* de forma que comandos possam ser executados no computador local.
?	Obtém ajuda no uso dos comandos *Telnet*.

Obs.: O comando z não é reconhecido em todas as implementações do serviço *Telnet* – nem mesmo em todas as implementações UNIX. Você deve consultar a documentação da sua versão particular do *Telnet* para determinar se o comando z é aceito e, se for, como reiniciar o *Telnet* após o comando z [10].

4.5. *File Transfer Protocol* (FTP)

O FTP é um protocolo que, assim como o *Telnet*, é encontrado nas camadas superiores. Sua função é permitir a transferência de arquivos, tanto ASCII (texto) quanto binários (codificados), entre computadores de uma rede TCP/IP, oferecendo aquele que é considerado o serviço padrão da Internet.

Esse serviço se baseia no estabelecimento de uma sessão padrão limitada entre o cliente FTP local e o servidor FTP do equipamento remoto. Durante uma sessão FTP, o usuário conecta-se com o outro computador usando o cliente FTP. A partir desse ponto, ele, o cliente, pode mover-se ao longo da árvore de diretórios, do conteúdo da lista de diretórios, copiar arquivos do computador remoto para o seu computador e transferir arquivos do seu computador para o sistema remoto.

Essa sessão é autenticada de forma parecida à do serviço *Telnet*, possuindo apenas comandos referentes à manipulação de diretórios e arquivos, permitindo ao usuário pesquisar a estrutura de arquivos do equipamento remoto antes de fazer a transferência de arquivos propriamente dita. É importante ressaltar que clientes FTP possuem grupos de comandos diferentes dependendo do sistema operacional utilizado. Entretanto, esses grupos variam pouco de sistema operacional para sistema operacional, evitando assim maiores problemas de compatibilidade.

O uso mais comum do serviço FTP na Internet é a obtenção de programas ou informações partindo de servidores de domínio público ou comercial. Esse serviço é conhecido como FTP Anônimo (*Anonymous FTP*). Para utilizá-lo, o usuário deve iniciar uma sessão FTP para o sistema remoto e usar como *username* a palavra *anonymous*, seguida da *password*, que será, em geral, o seu endereço de *e-mail*. Essa conta é especial, pois possui uma autenticação flexível do correio eletrônico do usuário somente para controle estatístico ou posterior comunicação. A sessão assim estabelecida tem acesso somente aos arquivos que puderem ser consultados ou transferidos para o computador do usuário e isto é definido pelo *host* servidor.

Abaixo, temos uma tabela com alguns dos comandos do programa FTP.

Comandos	**Funções**
ascii	Coloca o FTP em modo ASCII (quando for transferir arquivos de tipo texto).
binary	Coloca o FTP em modo binário (geralmente quando os arquivos a serem transferidos não forem de tipo texto. Ex.: .zip, .com, etc.).
cd	Esse comando permite a mudança de diretório na estação remota.
delete	Apaga um arquivo remoto.
dir	Mostra o conteúdo do diretório corrente.
disconnect	Termina uma sessão FTP.
get	Transfere um arquivo para o computador do usuário.
help	Lista os comandos do FTP.
mkdir	Cria um diretório na máquina remota.
open/close	Abre/Fecha uma sessão FTP.
quit/bye	Encerra o FTP.

Obs.: Muitos *sites* não concedem FTP Anônimo. Conceder a usuários do tipo *guest* permissão para se conectarem ao seu computador envolve alguns riscos. Nos casos em que o FTP Anônimo não é aceito, o comando FTP envia uma mensagem similar àquela quando o *login* falha – *User "anonymous" unknown*. Os *sites* que permitem FTP Anônimo geralmente colocam o usuário em uma árvore de

diretórios restrita que tem apenas acesso à leitura. Se você tem permissão para colocar arquivos no computador remoto, você usualmente pode apenas colocá-los em um diretório [10].

4.6. *Simple Mail Transfer Protocol* (SMTP)

O protocolo SMTP é outro protocolo encontrado nas camadas superiores e tem como função transferir correio eletrônico entre sistemas [3]. A maioria dos sistemas de *e-mail* que enviam *mails* pela Internet utilizam esse protocolo para enviar mensagens de um servidor até outro. As mensagens podem ser restabelecidas com um cliente de *e-mail*, que é um aplicativo que roda em um computador qualquer, permitindo ao usuário enviar, receber e organizar *e-mails* (usa-se a palavra cliente pois um sistema de *e-mail* baseia-se em uma arquitetura cliente-servidor). Utiliza-se para isto ou o POP (*Post Office Protocol*), um protocolo cuja função é restabelecer *e-mails* de um servidor de *mail*, ou o IMAP (*Internet Message Access Protocol*), um protocolo diferente, mas de função similar ao POP. Temos com isso que o SMTP é geralmente usado para enviar mensagens de um cliente de *mail* para um servidor de *mail*. Esse é o motivo que leva à necessidade de especificar ou o servidor POP ou o IMAP, além do servidor SMTP, quando o aplicativo de *e-mail* é configurado.

O *design* desse protocolo é baseado no seguinte processo de comunicação: um remetente SMTP estabelece um canal de transmissão *two-way* com um receptor SMTP, que pode ser tanto o destino final do *mail* enviado quanto um destino intermediário. Os comandos do SMTP são produzidos pelo remetente SMTP e enviados para o receptor. O receptor então responde aos comandos enviados pelo remetente.

Uma vez que o canal de comunicação é estabelecido, o remetente SMTP envia um comando MAIL indicando assim o remetente do *mail* em questão. Se o receptor SMTP puder receber essa mensagem, ele envia um OK para o remetente. Feito isto, o remetente SMTP envia então um comando RCPT, identificando um recipiente do *mail*. Mais uma vez, se o receptor SMTP puder receber o *mail* para esse recipiente, ele envia um OK como resposta; se não puder receber, ele rejeita esse recipiente, mas não toda a operação de *mail*. Ocorrendo essa rejeição, o remetente SMTP e o receptor SMTP podem "negociar" diversos recipientes. Quando esses recipientes tiverem sido "negociados", o remetente envia os dados de *mail*, terminados com uma seqüência especial. Se o receptor processar com sucesso os dados, é dado então o OK como resposta [16].

4.7. *Network File System* (NFS)

O NFS é um sistema operacional aberto, projetado pela *Sun Microsystems*, que permite o acesso por parte de todos os usuários de uma rede a arquivos compartilhados que estejam carregados em computadores de diferentes tipos, ou ainda, é um protocolo das camadas superiores que permite acesso remoto transparente de arquivos compartilhados entre redes. O NFS provê acesso a arquivos compartilhados através de uma interface chamada de *Virtual File System* (VFS), que roda no topo do TCP/IP. Os usuários podem então manipular esses arquivos como se estivessem carregados localmente em seus próprios discos rígidos. Com o NFS, computadores conectados a uma rede operam como se fossem clientes enquanto acessam arquivos remotos, e como servidores enquanto provêem acesso de arquivos compartilhados locais a usuários remotos.

Temos ainda que o protocolo NFS deve ser o menos estável possível, isto é, um servidor não deve precisar de qualquer tipo de informação de estado do protocolo em qualquer um dos seus clientes para garantir um bom funcionamento. Servidores não-estáveis (*stateless servers*) apresentam uma vantagem sobre servidores estáveis (*stateful servers*) no momento de uma falha. Com um servidor não-estável, um cliente somente precisa repetir o requerimento de um serviço qualquer até receber a resposta do servidor; não é necessário saber se o servidor ou a rede caíram. Por outro lado, um servidor estável deve ou detectar a falha no servidor e então reconstruir o estado do servidor quando esse volta a funcionar, ou produzir uma falha na operação executada pelo cliente [17].

4.8. *Simple Network Management Protocol* (SNMP)

O SNMP é um protocolo da camada de aplicações que tem a função de facilitar a troca de informações de gerenciamento entre dispositivos de rede. Ao usar-se esse protocolo para acessar esses dados, como por exemplo a quantidade de pacotes por segundo que estão sendo transmitidos, ou ainda as taxas de erros na rede, os administradores podem mais facilmente gerenciar a performance da rede, além de encontrar e solucionar problemas nessa.

Um dispositivo gerenciado pode ser qualquer tipo de nó residindo em uma rede, incluindo-se usuários, servidores de comunicação, impressoras, roteadores, *bridges* e *hubs*. Como alguns desses dispositivos podem ter limitações para rodar um software de gerenciamento, como ter CPU's lentas ou memórias limitadas por exemplo, esse software deve ser construído de forma a minimizar a sua própria performance de impacto no dispositivo que estiver sendo gerenciado [7].

4.8.1. Tipos de Comando

Se um NMS (*Network Management System*), que é uma das partes que constituem o modelo do SNMP, quiser controlar um dispositivo gerenciado, envia-se uma mensagem pedindo que o dispositivo altere uma ou mais de suas variáveis. Temos que dispositivos gerenciados iniciam ou respondem a quatro tipos de comandos diferentes:

- *Reads*: Usado pelos NMS's para monitorar dispositivos gerenciados. Os NMS's lêem as variáveis mantidas pelos dispositivos.

- *Writes*: Usado pelos NMS's para controlar dispositivos gerenciados. Os NMS's escrevem variáveis carregadas com os dispositivos gerenciados.

- *Traversal Operations*: Usado pelos NMS's para determinar quais variáveis um dispositivo suporta e para reunir seqüencialmente informações em tabelas de variáveis, como por exemplo em uma tabela de roteamento IP.

- *Traps:* Usado por dispositivos gerenciados para informar, sem qualquer sincronismo, certos eventos aos NMS's [7].

4.8.2. Diferenças entre Representação de Dados

A troca de informações em uma rede gerenciada está compromissada com as diferenças nas técnicas de representação de dados utilizadas pelos dispositivos gerenciados; isto é, vários computadores representam informações diferentemente. Essas incompatibilidades devem ser racionalizadas para permitir que sistemas diversos comuniquem-se. Essa função é executada ao reunir-se uma sintaxe de transferência com uma sintaxe abstrata. O SNMP usa a Sintaxe Abstrata de Notação Um (ASN.1 – *Abstract Sintax Notation One*), uma sintaxe abstrata definida como parte do OSI. Essa sintaxe é utilizada para definir tamanhos de pacotes e dispositivos gerenciados [7].

4.8.3. Base de Informações de Gerenciamento

Todos os dispositivos (ou objetos) gerenciados estão contidos na Base de Informações de Gerenciamento (MIB – *Management Information Base*), que é uma base de dados de objetos. Uma MIB é descrita como uma árvore, com os itens de dados individuais representados como licenças. Identificadores de objetos reconhecem tão somente objetos MIB em uma árvore. Podemos dizer que esses identificadores são como números de telefone, em que existe uma organização hierárquica, sendo as partes dos objetos apontadas por diferentes organizações. Os identificadores de objetos MIB das camadas superiores são apontados pela ISO/IEC (*International Organization of*

Standardization/International Electrotechnical Comission), enquanto que os objetos ID das camadas inferiores são alocados pelas organizações associadas [7].

4.8.4. Operações

Abaixo, temos os comandos definidos pelo protocolo SNMP, tendo sido usado como base a versão mais atual desse protocolo, que é o SNMP 2.0.

- *Get*: Recupera um exemplo de objeto de um agente. Um agente aqui é definido como sendo um módulo de software que é rodado em dispositivos gerenciados.
- *Get-Next*: Recupera o exemplo de objeto seguinte em uma tabela ou lista com um agente.
- *Set*: Estabelece exemplos de objetos com um agente.
- *Trap*: Informa ao NMS, de forma dessincronizada, sobre algum evento ocorrido. Ao contrário dos comandos anteriores, o *trap* não deduz uma resposta do receptor.
- *Inform*: Permite que um gerente envie informações do tipo *trap* para um outro gerente e peça então uma resposta.
- *Get-bulk*: Permite que um gerente recupere, de forma eficiente, um grande bloco de dados, assim como múltiplas colunas em uma tabela, que poderia por outro lado requerer a transmissão de vários blocos pequenos de dados [7].

4.8.5. Formato das Informações

Os pacotes SNMP são constituídos de duas partes: o cabeçalho do pacote e a unidade de dado do protocolo (PDU – *Protocol Data Unit*). O cabeçalho do pacote consiste de um número de versão e de um nome comunitário, sendo que esse último serve para 2 funções: a primeira é definir um meio de acesso para um grupo de NMS's usando então o próprio nome comunitário. A segunda é utilizá-lo como forma de autenticação, uma vez que os dispositivos que não conhecem o nome comunitário apropriado são excluídos das operações do SNMP. Já em relação a PDU, os seus campos para os comandos *get*, *get-next*, *set*, *response* e *trap* são os seguintes:

- Tipo de PDU: Identifica o tipo de PDU (*get*, *get-next*, *set*, *response* ou *trap*)
- ID requerido: Associa os requerimentos com as respostas.

- Erro de *Status*: Indica quando um erro ocorre e qual o tipo do erro.
- Índice de Erro: Associa o erro com uma variável particular nas variáveis de ligação.
- Variáveis de Ligação: Associam variáveis particulares com os seus valores correntes, com exceção para os requerimentos *get* e *get-next*, para os quais os valores são ignorados.

Temos que os campos de erro de *status* e índice de erros são estabelecidos como sendo zero quando são usados pelas operações *get, get-next, set ,trap* e *inform*. Somente a operação *response* estabelece esses dois campos.

No caso particular da operação *get-bulk*, os campos da PDU passam a ser os seguintes:

- Tipo de PDU, ID requerido e variáveis de ligação: análogos as funções da PDU para as operações anteriores já explicadas.

- *Nonrepeaters*: Especifica o número de variáveis na lista de variáveis de ligação para onde um sucessor lexicográfico retorna.

- *Max-Repetitions*: Especifica o número de sucessores lexicográficos que são retornados das variáveis remanescentes na lista de variáveis de ligação [7].

4.9. *Domain Name Service* (DNS)

O DNS é um protocolo de camada superior que cria um banco de dados, onde são armazenados os nomes dos dispositivos da rede, sendo esses nomes associados à endereços IP. Assim, define-se a sintaxe dos nomes usados na Internet, permite-se o gerenciamento na escolha de nomes e define-se um algoritmo de mapeamento de nomes em endereços. Isto se justifica pelo fato de que os usuários preferem identificar os dispositivos por meio de nomes ao invés de números.

Um conjunto de servidores de nomes interconectados (servidores DNS) mantém um banco de dados com os nomes e endereços das máquinas conectadas à Internet. Os servidores são consultados da seguinte forma: quando um pacote é enviado, por exemplo, para a máquina *Apollo.cat.cbpf.br* (computador da CAT), um servidor central (servidor raiz) procura onde está o servidor *br*. Como resultado, o servidor raiz mostra os endereços IP de vários servidores de nível *br*. Feito isso, um servidor desse nível informa o endereço IP associado ao domínio *cbpf.br*, e dessa forma a procura é feita até ser encontrada a máquina de nome *Apollo*, no domínio *cat.cbpf.br*.

Adicionalmente, o DNS, além de armazenar endereços Internet, mantém informações referentes às características dos dispositivos, sobre os usuários e outros objetos [2].

A seguir, verificaremos os protocolos constituintes de uma rede *Novell*. Abaixo, similarmente aos protocolos da Internet, temos uma tabela comparativa entre os protocolos *Novell* e o modelo de referência OSI.

<table>
<tr><th>Modelo OSI</th><th colspan="3">Novell</th></tr>
<tr><td>Camada de Aplicações</td><td colspan="3" rowspan="3"></td></tr>
<tr><td>Camada de Apresentação</td><td rowspan="3">NCP</td></tr>
<tr><td>Camada de Sessão</td><td colspan="2" rowspan="2">Aplicações do Netware</td></tr>
<tr><td>Camada de Transporte</td><td>SPX</td></tr>
<tr><td>Camada de Rede</td><td colspan="3">IPX</td></tr>
<tr><td>Camada de Link de Dados</td><td colspan="3" rowspan="2">Ethernet, Token Ring, ARCnet</td></tr>
<tr><td>Camada Física</td></tr>
</table>

4.10. *Ethernet*

O *Ethernet* é um protocolo LAN (*Local Area Network*), desenvolvido pela *Xerox Corporation* em parceira com a DEC e a Intel em 1976, que utiliza uma topologia em barra ou em estrela que suporta taxas de transferência de dados de até 10 Mbps. A especificação *Ethernet* serve como base para o modelo IEEE 802.3, um modelo de rede criado pela IEEE (*Institute of Electrical and Electronics Engineers*), que define a camada MAC para redes de topologia em barra que utilizam CSMA/CD (*Carrier Sense Multiple Acess / Collision Detection*), um método de acesso usado para manejar demandas simultâneas. Temos que o *Ethernet* é um dos modelos LAN mais implementados. Uma versão mais nova do *Ethernet*, chamada de *100Base-T*, ou *Fast Ethernet*, suporta taxas de transferência de arquivos de 100 Mbps, e a mais nova versão, chamada de *Gigabit Ethernet*, suporta até 1 *gigabit* (1000 *megabits*) por segundo [21].

4.11. *Token Ring*

O *Token Ring* é um tipo de rede em que todos os computadores são arrumados esquematicamente em um círculo. O *token* (sinal), que é um padrão especial de bit, viaja através desse círculo. Quando se quer enviar uma mensagem, um computador captura esse *token* e adiciona essa mensagem. Feito isto, a máquina deixa o sinal continuar circulando pela rede. Dessa forma, ao querer transmitir a estação espera pela permissão. Ao recebê-la, altera o padrão para "permissão ocupada" e transmite seus dados. A estação transmissora é responsável por retirar sua mensagem do anel e inserir a nova "permissão."

Quando capitalizado, temos que o *Token Ring* refere-se ao protocolo de rede de PC, desenvolvido pela IBM nos anos 70. A especificação *Token Ring* da IBM foi padronizada pela IEEE como o modelo IEEE 802.5, que define camada MAC para redes desse tipo [22].

4.12. *ARCnet*

O *ARCnet* (*Attached Resource Computer Network*) é um sistema de rede simples que suporta os 3 tipos de cabos primários, que são os cabos de par trançado, os coaxiais e os de fibra ótica, além das topologias em barra e estrela. Esse sistema foi desenvolvido pela *Datapoint Corporation* e introduzido no mercado em 1977. Apesar de não ter alcançado a popularidade do *Ethernet* e do *Token Ring*, o custo baixo e a flexibilidade do *ARCnet* originaram vários defensores fiéis à sua utilização [7].

4.13. *Internet Packet Exchange* (IPX)

O protocolo IPX é um protocolo da camada de rede, utilizado por sistemas operacionais *Novell Netware*. Assim como os protocolos UDP e IP, o IPX é um protocolo de datagrama utilizado para comunicações remotas, em que o *host* pode enviar uma mensagem sem estabelecer uma conexão com o receptor. Como exemplo, imagine um dispositivo localizado em uma rede diferente daquela em que você está conectado. O IPX permite a comunicação com esse dispositivo roteando as informações na direção desse através de qualquer rede intermediária. Temos ainda que protocolos de níveis superiores, como por exemplo o SPX e o NCP, que veremos adiante, são usados em serviços adicionais de correção de erros [19].

Os campos constituintes de um pacote IPX são os seguintes:

- *Checksum*:

É um campo de detecção de erros, em que cada pacote transmitido é acompanhado de um valor numérico calculado com base no seu número de bits.

- Tamanho de Pacotes:

Um campo de 16 bits que especifica o tamanho em bytes do datagrama IPX completo. Os pacotes IPX podem ser de qualquer tamanho acima do tamanho da unidade de transmissão média máxima (MTU). Ainda não existe fragmentação de pacotes IPX.

- Controle de Transporte:

Um campo de 8 bits que indica o número de roteadores pelos quais o pacote passou. Quando o valor desse campo alcança 15, o pacote é descartado sob a hipótese de que um *loop* de roteamento esteja ocorrendo.

- Tipo de Pacote:

Um campo de 8 bits que especifica o protocolo de camada superior para receber as informações do pacote. Dois valores comuns desse campo são o 5, que especifica o SPX, e o 17, que especifica o NCP.

- Rede de Destino, Nó de Destino e Soquete de Destino:

Esses três campos especificam as informações de destino.

- Rede de Origem, Nó de Origem e Soquete de Origem:

Esses três campos especificam as informações de origem.

- Dados das Camadas Superiores:

Esse campo contém informações para os processos das camadas superiores [7].

4.14. *Sequenced Packet Exchange* (SPX)

O protocolo SPX é um protocolo da camada de transporte utilizado por redes *Novell Netware*. Na verdade, o SPX é o protocolo de transporte *Netware* mais usado. Assim como o protocolo TCP e muitos outros protocolos de transporte, o SPX é um protocolo de conexão orientada, isto é, ele oferece serviços de conexão orientada entre 2 nós de uma rede [4]. Devido a essa sua similaridade com o TCP, e mais a similaridade do protocolo IPX com o IP, temos que, juntos, o IPX/SPX provêem serviços de conexão similares ao TCP/IP. O SPX é utilizado ainda em aplicações cliente/servidor [20].

4.15. *Netware Core Protocol* (NCP)

O NCP implementa a comunicação entre servidores e estações. Esse protocolo mantém seu próprio controle de conexão e de erro nos pacotes, não permitindo que protocolos diferentes executem essa tarefa. Consequentemente, ele ocupa 3 camadas do modelo OSI: a camada de transporte, devido à conexão realizada por ele; a camada de sessão, devido a comunicação de 2 vias e a camada de apresentação, que vai gerenciar a maneira como os dados serão representados [4].

Por último, depois de vermos os protocolos da Internet e de uma rede *Novell*, estudaremos o protocolo utilizado em uma rede NT, que é o NetBEUI.

4.16. *NetBios Enhanced User Interface* (NetBEUI)

O NetBEUI é uma versão atualizada do protocolo NetBios, sendo utilizado por sistemas operacionais de rede, como por exemplo, gerentes e servidores de LAN's, *Windows for Workgroups*, Windows 95 e Windows NT. Esse protocolo foi originalmente designado pela IBM para o seu gerente servidor de LAN, sendo mais tarde desenvolvido pela Microsoft e *Novell.*

O NetBios (*Network Basic Input Output System*), a versão anterior do protocolo NetBEUI, é uma interface de programa de aplicações (API – *Application Program Interface*), que atualiza a BIOS do DOS, por meio da adição de funções especiais de LAN's. Temos que quase todas as LAN's para PC's são baseadas no NetBios. Alguns fabricantes de LAN's até mesmo estenderam esse protocolo, adicionando a ele algumas capacidades de rede. O formato das mensagens transmitidas pelo NetBios é chamado de Servidor de Bloco de Mensagens (SMB – *Server Message Block*) [24], sendo esse um protocolo utilizado para compartilhamento de arquivos, impressoras, portas seriais e ambientes de rede, como, por exemplo, os ambientes NT e UNIX [9] [30].

5. *Local Area Network* (LAN)

Uma LAN, como já vimos anteriormente, é um tipo de rede de computadores, em que a área ocupada pelas máquinas que a compõem é relativamente pequena. Exemplificando, temos que a maioria das LAN's são confinadas a um único prédio ou a um grupo de prédios. Contudo, uma LAN pode ser conectada a outras LAN's quaisquer a qualquer distância, através de linhas de telefone, ondas de rádio, etc. Um sistema de LAN's que são conectadas dessa forma passa a ser chamado de *Wide Area Network* (WAN).

A maioria das LAN's conectam *workstations* e computadores pessoais. Temos que cada nó tem a sua própria CPU, através da qual os programas são executados, além de também serem capazes de acessar dados e quaisquer dispositivos na LAN. Isto é, vários usuários podem compartilhar dispositivos, como por exemplo impressoras *lasers*, tão bem como compartilham dados. Os usuários também podem utilizar a LAN para comunicarem-se entre si, ou através do envio de *e-mails*, ou através de sessões de *chat*.

Abaixo, temos as características que diferenciam uma LAN de outra:

- Topologia:

É o formato de uma LAN, ou de outros sistemas de comunicações quaisquer, que determina a distribuição geométrica dos dispositivos que a compõem. Os três principais tipos de topologia que são utilizadas em LAN's são:

1. Topologia *bus*:

Todos os dispositivos estão conectados a um cabo central, conhecido como *bus* ou *backbone*. Redes que utilizam esse tipo de topologia são relativamente baratas e fáceis de instalar (nos casos de redes pequenas). Temos ainda que sistemas *Ethernet* utilizam uma topologia *bus*.

2. Topologia *ring*:

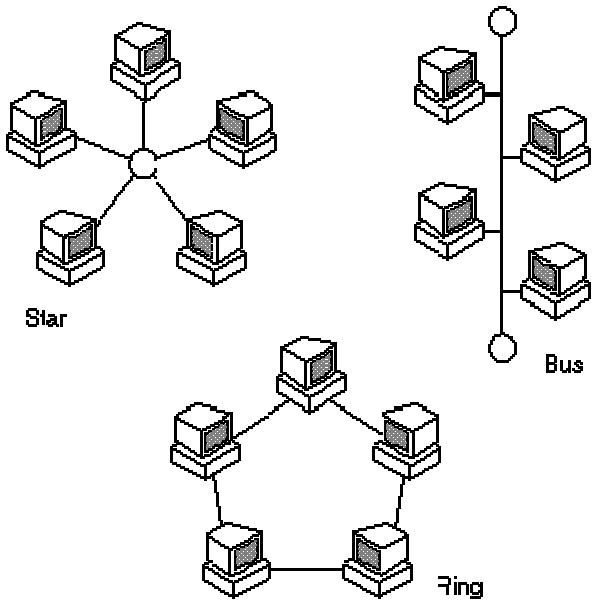
Todos os dispositivos são conectados uns aos outros no formato de um *loop* fechado, de forma que cada um dos dispositivos está conectado diretamente com outros dois dispositivos, cada um de um lado. As topologias *ring* são caras e difíceis de instalar, mas em compensação oferecem uma alta largura de banda, além de poderem percorrer longas distâncias.

3. Topologia *star*:

Todos os dispositivos são conectados a um *hub* central. Redes que usam uma topologia *star* são geralmente fáceis de instalar e gerenciar, podendo porém ocorrer congestionamentos de tráfego, já que os dados tem que passar através do *hub*.

É importante ressaltar que essas topologias podem ser combinadas. Por exemplo, uma rede *bus-star* consiste de uma alta largura de banda do tipo *bus*, chamada de *backbone*, que conecta um conjunto de segmentos do tipo *star* com larguras de banda menores.

Abaixo, temos a representação gráfica de cada um dos 3 tipos de topologia apresentados.



- Protocolos:

São as regras e especificações de codificação utilizados na transmissão de dados. Os protocolos também determinam a arquitetura utilizada pela rede, que pode ser ou ponto-a-ponto ou cliente/servidor.

- Mídia:

Os dispositivos podem ser conectados através de cabos de par trançado, coaxiais ou de fibra ótica. Entretanto, existem casos de redes em que esse tipo de conexão de mídia não é feita. Nesses casos, os dispositivos comunicam-se através de ondas de rádio.

As LAN's são capazes de transmitir dados a taxas muito rápidas, muito mais rápidas do que os dados que podem ser transmitidos em uma linha de telefone. Porém, as distâncias são limitadas, e também há um limite no número de computadores que podem ser conectados a uma única LAN [23].

Abaixo, falaremos à respeito das funções dos componentes básicos de uma rede local, baseando-nos naqueles que são utilizados na rede local do CBPF.

5.1. *Hub*

Um *hub* é um dispositivo utilizado para conectar os equipamentos que compõem uma LAN. Com ele, as conexões da rede são concentradas, fato esse que faz com que o *hub* também seja conhecido como concentrador, ficando cada equipamento em um segmento próprio. Com isso, o gerenciamento da rede é favorecido e a solução de problemas facilitada, uma vez que o defeito fica isolado no segmento da rede.

Os *hubs* mais comuns são os *hubs Ethernet 10Base-T* (conectores RJ-45), sendo eventualmente parte integrante de *bridges* e roteadores [5].

Um tipo existente de *hub* é o passivo. Esse *hub* serve simplesmente como um canal de dados, que possibilita o tráfego desses dados de um dispositivo, ou segmento, para outro dispositivo qualquer.

Um segundo tipo de *hub* é o chamado *hub* inteligente. A diferença desse dispositivo para o anterior é que esse último inclui aspectos adicionais que permitem que um administrador monitore todo o tráfego que está passando pelo *hub* e configure cada porta desse. Esse tipo é também conhecido como *hub* gerenciável.

Por último, ainda temos o *switching hub*, que é o tipo de *hub* utilizado na rede do CBPF. O *switching hub* é um tipo especial de *hub* que transmite pacotes para a porta apropriada, baseando-se no endereço do pacote. Já os *hubs* convencionais simplesmente fazem *broadcast* de cada pacote repetidas vezes para cada porta. Assim, vê-se que, pelo fato do *switching hub* transmitir cada pacote somente para a porta requerida, a performance da rede torna-se muito melhor, pois evita-se assim que ela fique lenta. Temos ainda que a maioria dos *swtiching hubs* também suportam balanceamento de carga, de forma que as portas são redirecionadas dinamicamente para segmentos diferentes da LAN, baseando-se nos padrões de tráfego, evitando assim que um dispositivo seja inundado com dados. Alguns *switching hubs* mais novos suportam tanto portas *Ethernet* tradicionais, de 10 Mbps, quanto portas *Fast Ethernet* de 100 Mbps. Isto permite que o administrador de rede estabeleça um canal *Fast Ethernet* para dispositivos de tráfego elevado, como servidores por exemplo [25].

5.2. *Switch*

O *switch* é um dispositivo de diversas portas, com cada uma delas podendo ser conectada ou a várias estações (sob a forma de uma LAN), ou a uma única estação. A sua função é segmentar uma rede muito grande em LAN's menores e menos congestionadas, de forma a melhorar o desempenho da rede. Esse aumento de performance é obtido fornecendo a cada porta do *switch* uma largura de banda dedicada. No caso de redes locais diferentes serem conectadas em cada uma dessas portas, pode-se transmitir dados entre essas LAN's conforme o necessário. O *switch* também provê uma filtragem de pacotes entre LAN's que estejam separadas.

Os *switches* geralmente suportam as implementações *Ethernet*, padrão IEEE 802.3, de 10 Mbps. É comum encontrar *switches* cujas portas operam a velocidades diferentes, permitindo conexões de até 100 Mbps, utilizando a especificação *100Base-T*. Com um funcionamento independente do meio de transmissão, o tipo de meio que pode ser ligado ao *switch* é uma questão de implementação, sendo possível ligar LAN's com diferentes meios de transmissão a diferentes portas de um mesmo *switch*.

Esse dispositivo funciona com base em barramentos (*backplanes*) internos de alta velocidade, utilizados na transmissão de pacotes entre as suas portas, sendo esses compatíveis com a tecnologia ATM. As estações de uma rede local utilizam uma banda passante igual a da porta a que estão conectadas. É possível também ligar uma mesma estação a mais de uma porta do *switch*, o que aumenta a banda passante disponível para essa estação. Para isso, basta que todas as placas de rede dela sejam conectadas a portas diferentes do *switch* [2].

5.3. Roteador

O roteador é um dispositivo que conecta duas LAN's diferentes, roteando os pacotes entre elas. Esse dispositivo é operado nas camadas física, de *link* de dados e de rede. O seu funcionamento é similar a uma *bridge*. A diferença é que o roteador provê funcionalidades adicionais, como por exemplo, a capacidade de filtrar pacotes e transmiti-los para lugares diferentes, baseando-se em critérios que tenham sido pré-estabelecidos. Já uma *bridge* é um dispositivo independente de protocolo; isto é, ela simplesmente transmite pacotes sem analizá-los ou roteá-los novamente. Consequentemente, uma *bridge* é mais rápida que um roteador; mas, em compensação, o roteador é mais flexível [28].

Para estabelecer a conexão entre as LAN's, o roteador utiliza um protocolo de roteamento, como por exemplo o RIP (*Routing Information Protocol*) ou o OSPF (*Open Shortest Path First*), para obter informações sobre a rede. Em relação aos exemplos dados, o protocolo OSPF é mais eficiente do que o protocolo RIP, uma vez que esse último especifica como o roteador transfere tabelas de roteamento inteiras, enquanto o primeiro transfere apenas as informações de roteamento que foram alteradas desde a última transferência. O protocolo que for utilizado baseia-se em algoritmos para escolher a melhor rota, sendo composto por vários critérios conhecidos como "Métrica de roteamento". Os roteadores podem também comprimir e compactar dados.

Os roteadores permitem que LAN's tenham acesso a WAN's (*Wide Area Networks*). Normalmente, um roteador é composto de uma porta LAN, que pode ser do tipo *Ethernet* ou *Token Ring*, e várias portas WAN, como por exemplo o PPP (*Point-to-Point Protocol*), um protocolo que permite a conexão de um computador com a Internet; o *Frame-Relay*, um protocolo de comutação de pacotes (*packet switching*) que conecta dispositivos em uma WAN, ou o ISDN (*Integrated Services Digital Network*), que é um modelo de comunicação internacional usado para envio de voz, vídeo e dados através de linhas digitais de telefones.

Os roteadores normalmente trabalham com IP, IPX e os endereços IP's definidos na tabela de roteamento repassados à rede WAN [5].

Temos que o roteador é amplamente utilizado pela Internet para transmitir pacotes de um *host* para outro. Nesse contexto, existem 3 aspectos de roteamento importantes:

1. Determinação do endereço físico.
2. Seleção de *gateways* da rede interna.
3. Endereços simbólicos e numéricos.

O primeiro desses aspectos é necessário quando um datagrama IP é transmitido por um computador. É necessário dividir esse datagrama com o formato de *frame* que estiver sendo utilizado pela rede local, ou pelas redes, na qual o computador está ligado. Essa divisão requer a inclusão de um endereço físico, ou de rede local, no *frame*.

O segundo aspecto é necessário porque a Internet consiste de um número de redes locais interconectadas por um ou mais *gateways*. Alguns *gateways*, que nada mais são do que roteadores, algumas vezes apresentam conexões físicas ou portas em várias redes. A determinação do *gateway* e portas apropriadas de um datagrama IP é chamada de roteamento e também envolve *gateways* trocando informações entre si no modo padrão.

O terceiro aspecto envolve tradução de mensagens para endereços IP numéricos, sendo esse processo executado pelo serviço DNS (*Domain Name Service*) [29].

6. A Rede do CBPF

Nesse item, falaremos à respeito da rede estruturada do CBPF. Primeiramente, serão vistos os equipamentos que estão instalados na Coordenação de Atividades Técnicas (CAT), encontrados em uma sala conhecida como *site*, e que controlam tudo o que está relacionado à rede do CBPF. Nesse item, serão especificados os modelos de cada equipamento. Em seguida, será estudada a estruturação utilizada na rede do CBPF. Veremos como os equipamentos utilizados pela rede estão distribuídos, tanto no aspecto físico quanto no aspecto lógico.

6.1. Equipamentos

Neste item, serão dadas as especificações dos equipamentos encontrados no *site* da CAT e que controlam a rede do CBPF:

- *OnCore Switching Hub*: Concentrador de 17 *slots*, modelo 6017A-AC, fabricado pela *3Com*;
- *Backplane*: 3 barramentos do *OnCore*, modelos de placa 3C96017C-P cada um, fabricados pela *3Com*;
- Módulos *hub*: 5 módulos *hub 10Base-T* (conectores RJ-45), todos modelo de placa 6140M-TPP, com saída para 24 portas cada um, fabricados pela *3Com*. Cada um desses módulos ocupa 2 *slots* do *OnCore*;
- Módulos *switch* de 10 Mbps: 2 módulos *switch*, modelos de placa 6612M-TP e 6612D-TP, com saída para 12 portas cada um, fabricados pela *3Com*. Cada um deles ocupa 1 *slot* do *OnCore*, sendo que estão interligados;
- Módulo *switch* de 100 Mbps: 1 módulo *switch*, modelo de placa 6604M-TX, com saída para 4 portas, fabricado pela *3Com*. Ocupa 1 *slot* do *OnCore*. Atualmente, esse módulo não está sendo utilizado;
- Módulo controlador: Módulo de controle do *OnCore*, modelo de placa 6000M-MGT, fabricado pela *3Com*. Ocupa 1 *slot* do *OnCore*;
- Módulos de alimentação: 2 módulos de alimentação, modelos de placa 6000M-RCTL cada um, fabricados pela *3Com*;
- *Stackable hubs*: 12 *SuperStack II Port Switch Hub* de 10 Mbps, todos modelos de placa 3C16401, com saída para 24 portas cada um, fabricados pela *3Com*. Um *stackable hub* é um tipo de *hub* que pode ser ligado um sobre o outro, formando uma "pilha";

- *Stackable switch* de 100 Mbps: 1 *SuperStack II Switch 1000*, modelo de placa 3C16900A, com saída para 24 portas, fabricado pela *3Com.* Esse *switch* não está sendo utilizado atualmente;
- Roteador: Roteador de 12 *slots*, modelo 7500 *Series*, fabricado pela *Cisco Systems Inc.*;
- *No-breaks*: 3 *no-breaks* inteligentes, fabricados pela *Engetron*;
- *Hub*: 1 *Accton EtherHub-8s*, com saída para 8 portas, fabricado pela *SmartWatch*.

6.2. Estruturação

Toda a rede do CBPF é estruturada, permitindo assim um melhor gerenciamento dos seus dispositivos. Com esse gerenciamento, é possível manter a qualidade na performance da rede, através do controle de possíveis problemas nessa.

6.2.1. Estruturação Física

Neste item, será abordada a disposição dos equipamentos que se encontram no *site* da CAT, e que controlam toda a rede do CBPF. Primeiramente, será explicado como está estruturado todo o cabeamento da rede, e em seguida verificaremos como os equipamentos do *site* estão ligados uns aos outros.

Em relação ao cabeamento, temos que no bloco terminal (*patch panel*), encontrado no *site* na CAT, estão ligados vários cabos de par trançado de 25 pares. Mais especificamente, esses cabos estão ligados na parte de trás do bloco primário do *patch panel.* De lá, eles saem, tendo sido passados por dentro da parede, chegando em todos os andares do CBPF, com exceção do terceiro. Em cada um dos andares em que o cabeamento chega, os cabos de 25 pares são ligados na parte de trás de um pequeno bloco terminal, fixado na parede do corredor do andar em questão. Todos os andares, menos o terceiro, tem esse bloco. Na parte da frente desses blocos, também saem cabos de par trançado, só que de 4 pares, que são então distribuídos, também por dentro da parede, pelas salas do andar. Cada um desses cabos tem uma tomada correspondente em cada sala, e são dessas tomadas que são feitas as ligações com as máquinas (também com cabos de 4 pares). No caso do terceiro andar, os cabos que saem dos dispositivos de rede estão ligados diretamente ao bloco primário do *patch panel* do *site.*

Como já explicado, esses cabos de 4 e de 25 pares estão ligados na parte de trás do bloco primário do *patch panel.* Da parte da frente desse mesmo bloco saem cabos de 4 pares que são então conectados à parte da frente do bloco secundário do *patch panel.* Essa ligação garante uma maior flexibilidade para possíveis mudanças de pontos de rede. Ainda, é importante dizer que no caso desses cabos conectados ao blocos primário e secundário do *patch panel*, os conectores utilizados são do tipo IDC, enquanto que os conectores utilizados nos cabos que estão ligados nas máquinas são do tipo RJ-45.

Da parte de trás do bloco secundário do *patch panel* saem cabos de 4 pares, que estão conectados às portas dos módulos *hub*, a algumas portas do módulo *switch* de 10 Mbps e aos *stackable hubs*. Além desses, alguns cabos de 4 pares também estão ligados ao roteador.

Em relação ao *OnCore*, temos ligados nele 5 módulos *hub*, 1 módulo *switch* de 10 Mbps, 1 módulo *switch* de 100 Mbps (não utilizado), 1 módulo controlador e 2 módulos de alimentação. Os módulos *hub*, *switch* de 10 Mbps, *switch* de 100 Mbps e controlador estão todos ligados aos barramentos (*backplanes*) do *OnCore*. Por esses barramentos são transmitidos os dados entre os módulos. Por esse motivo, os módulos de alimentação (parte elétrica) não são ligados a eles, e sim em um dos 3 *no-breaks* encontrados no *site*.

Em relação aos *no-breaks* do *site*, temos que, como dito no parágrafo acima, os módulos de alimentação do *OnCore* estão ligados em um deles. Ainda no *no-break* em questão, encontram-se ligadas as 3 fontes de alimentação do *OnCore*, além de uma fonte que está ligada diretamente ao *stackable switch* de 100 Mbps. No segundo *no-break*, estão ligados alguns servidores encontrados no *site*, como por exemplo as máquinas CBPFSU1 e MESON, enquanto que no terceiro está ligada a fonte de alimentação do roteador. Sempre que, por algum motivo, houver uma falha na alimentação de qualquer um desses equipamentos, o *no-break* correspondente é ativado, garantindo assim que eles continuem a funcionar normalmente.

Um dos principais equipamentos encontrados no *site*, se não o mais importante, é o roteador. No caso do roteador utilizado no CBPF, apenas um dos seus *slots* está sendo utilizado. Nesse *slot* está ligado um módulo com saída para 6 portas. Em cada uma dessas portas está ligada cada uma das 6 redes, ou segmentos, do CBPF:

1) Rede 10: ligada à porta 24 do último *stackable hub*, contando-se todos os *stackable hubs* no sentido de cima para baixo, utilizando um cabo de par trançado invertido (cor azul) de 4 pares;
2) Rede 100: ligada à porta 9 do módulo *switch* de 10 Mbps, utilizando um cabo de par trançado de 4 pares;
3) Rede 250: ligada à porta 24 do penúltimo *stackable hub*, contando-se todos os *stackable hubs* no sentido de cima para baixo, utilizando um cabo de par trançado invertido (cor azul) de 4 pares;
4) Rede 252: ligada à porta 6 do módulo *switch* de 10 Mbps utilizando um cabo de par trançado de 4 pares;
5) Rede 253: ligada à porta 2 do módulo *switch* de 10 Mbps, utilizando um cabo de par trançado de 4 pares;
6) Rede 254: ligada ao LNCC (Laboratório Nacional de Computação Científica), utilizando fibra ótica.

Em relação ao cabeamento, são ligados *transceivers* a cada uma das portas do módulo do roteador, uma vez que os seus conectores são do tipo AUI, diferentes dos conectores RJ-45 do módulo *switch* e dos *stackable hubs*. No caso dos cabos de par trançado invertidos, a sua utilização deve-se ao fato de estarem-se fazendo ligações entre os *hubs* e o *switch*.

Por último, como ainda existem algumas máquinas no CBPF que utilizam-se do antigo cabeamento coaxial, existe um pequeno *hub* de 8 portas, que está ligado à porta 8 do módulo *switch* de 10 Mbps. Esse cabo coaxial, assim como o cabeamento de par trançado, chega até o *site* da CAT sendo passado por dentro da parede. A esse *hub* é ligado também um pequeno adaptador de tensão, que está ligado diretamente à uma das fontes de alimentação vazias do *OnCore*.

6.2.2. Estruturação Lógica

Em relação à estruturação lógica da rede do CBPF, temos que esta foi dividida em 6 redes menores (segmentos), que são:

1) A rede 10, onde estão agrupados os computadores utilizados para testes;
2) A rede 100, onde estão agrupadas todas as máquinas do LAFEX;
3) A rede 250, onde encontram-se os computadores da CDI e alguns do CBPF;
4) A rede 252, onde estão o restante dos computadores do CBPF;
5) A rede 253, onde encontram-se os computadores da CAT e as máquinas SUN;
6) A rede 254, que liga o CBPF ao LNCC (Laboratório Nacional de Computação Científica).

Em cada uma dessas 6 redes encontram-se, além dos computadores, outros vários dispositivos de rede, como por exemplo impressoras, servidores, etc. Todos esses dispositivos comunicam-se entre si, através de transmissão de dados. No caso de 2 dispositivos que pertençam a redes diferentes, os dados são transmitidos de uma rede para outra através do roteador. Com isso, tínhamos que, quando existiam vários usuários comunicando-se entre si, simultaneamente, através das redes do CBPF, o tráfego de pacotes tornava-se intenso, o que prejudicava sensivelmente a performance da rede; isto é, ela tornava-se mais lenta, já que o roteador acabava sendo sobrecarregado.

Para evitar tal problema, foram criados os chamados domínios de colisão (*collision domains*). Um domínio de colisão é um grupo de máquinas que são agrupadas, por meio ou de software ou de hardware, de tal forma que elas comuniquem-se apenas entre si. Isto significa que quando um *host* qualquer transmite um pacote através da rede, somente os *hosts* pertencentes ao mesmo domínio de colisão do *host* transmissor irão receber esse pacote. Assim, congestionamentos de tráfego na rede são evitados. Fisicamente, os domínios de colisão encontrados na rede do CBPF são representados por grupos de portas, agrupadas logicamente, dos módulos *hubs* instalados no *OnCore*, além dos *stackable hubs*, sendo cada um deles um domínio de colisão. Abaixo, são apresentados os domínios de colisão existentes na rede do CBPF:

1) ETH1: Domínio de colisão determinado pelo *OnCore*, onde estão agrupados os PC's da CAT (rede 253);
2) ETH2: Domínio de colisão determinado pelo *OnCore*, onde estão agrupados alguns dos PC's do CBPF (rede 252);
3) ETH3: Domínio de colisão determinado pelo *OnCore*, onde estão agrupadas algumas máquinas do LAFEX (rede 100);

4) ETH4: Domínio de colisão determinado pelo *OnCore*, onde estão agrupadas algumas máquinas SUN (rede 253);
5) STACK1: *Stackable hub* utilizado como domínio de colisão, onde encontram-se agrupados mais PC's do CBPF (rede 252);
6) STACK2: *Stackable hub* utilizado como domínio de colisão, onde também encontram-se agrupados PC's do CBPF (rede 252);
7) STACK3: *Stackable hub* utilizado como domínio de colisão, onde estão agrupados outros PC's do CBPF (rede 252);
8) STACK4: *Stackable hub* utilizado como domínio de colisão, onde estão agrupados mais PC's do CBPF (rede 252);
9) STACK9: *Stackable hub* utilizado como domínio de colisão, onde estão agrupadas mais algumas máquinas do LAFEX (rede 100);
10) STACK10: *Stackable hub* utilizado como domínio de colisão, onde está o restante das máquinas do LAFEX (rede 100);
11) STACK11: *Stackable hub* utilizado como domínio de colisão, onde estão agrupadas as máquinas da CDI e o restante das máquinas do CBPF (rede 250);
12) STACK12: *Stackable hub* utilizado como domínio de colisão, onde estão agrupados todos os computadores utilizados para testes (rede 10).

Apesar dos domínios de colisão evitarem possíveis congestionamentos na rede, ainda existe um problema a ser resolvido. Por exemplo, se um *host* que pertence ao domínio de colisão ETH1 quiser transmitir algum dado ou informação para outro *host* que encontra-se no domínio ETH4, como será feita a transmissão, se apenas os *hosts* pertencentes ao domínio ETH1 receberão este pacote?

A solução desse problema é a criação de *vbridges* (*virtual bridges*). Uma *vbridge* é um grupo de domínios de colisão, que são agrupados por meio de software. Quando uma máquina pertencente a um domínio qualquer quiser comunicar-se com outra máquina que se encontra em um domínio diferente, o pacote transmitido pela máquina transmissora é enviado através da rede, passando então pela *vbridge*, que o redireciona para o domínio onde se encontra a máquina receptora. Fisicamente, as *vbridges* encontradas na rede do CBPF são representadas por grupos de portas, agrupadas logicamente, do módulo *switch* de 10 Mbps, instalado no *OnCore*. Abaixo, estão as *vbridges* encontradas na rede do CBPF:

1) *Vbridge* 1: É formada pelos domínios de colisão ETH1, ETH4, pelas máquinas servidoras CBPFSU1 e CBPF-CAT, além do roteador;
2) *Vbridge* 2: É formada pelos domínios de colisão ETH2, STACK1, STACK2, STACK3, STACK4, pelos computadores que ainda utilizam o cabeamento coaxial (rede 252) e pelo roteador;
3) *Vbridge* 3: É formada pelos domínios de colisão ETH3, STACK9, STACK10 e pelo roteador;
4) *Vbridge* 4: É formada pelo domínio de colisão STACK11 e pelo roteador.

Como podemos perceber, em todas as *vbridges* encontra-se o roteador. Isto é necessário porque se, por exemplo, uma máquina do domínio de colisão ETH1, pertencente à *vbridge* 1, quiser transmitir dados a uma outra máquina, que encontra-se no domínio de colisão ETH2, constituinte da *vbridge* 2, o pacote deve passar pelo roteador; isto é, a informação é enviada pela máquina transmissora, sendo analisada pela *vbridge*. Ao verificar que aquela informação está direcionada para uma máquina que não

se encontra nela, a *vbridge* em questão redireciona então os dados para o roteador, que então os envia para o domínio onde se encontra a máquina receptora.

Referências:

Em relação às referências de páginas *web*, é importante ressaltarmos que páginas da Internet são altamente dinâmicas, podendo mudar desde parte do seu conteúdo até o próprio endereço. Devido a isto, existe a possibilidade de que, ao consultar alguma página, essa não mais exista ou então tenha outras informações.

[1] "Redes de Computadores – Segunda Edição Americana" / *Andrew S. Tanenbaum*

[2] "Redes de Computadores – Das LAN's, MAN's e WAN's até às Redes ATM – Segunda Edição" / Luiz Fernando Gomes Soares, Guido Lemos e Sérgio Colcher

[3] "*Using Linux*" / *Jack Tackett Jr.*, *David Gunter* e *Lance Brown.*

[4] "Netware 4 para profissionais" / *Doug Bierer, Charles Hatch, Dee Anne Higley, Timothy Gendreau & Karanjit Siyan*.

[5] "Guia Internet de Interconectividade" / Publicação da *Cyclades* Brasil.

[6] "*Systimax Structured Cabling Systems* – Guia de Referência Rápida" / Documentação da *Lucent Technologies*.

[7] "*Cisco Connection Documentation*" / Documentação da *Cisco Systems Inc.*

[8] "Netware v4.10 – Considerações Iniciais" / Nota Técnica escrita por Alexandre Ferreira Novello & Nilton Alves Jr.

[9] "Windows NT Server 4.0 – Considerações Iniciais" / Nota Técnica escrita por Alexandre Ferreira Novello & Nilton Alves Jr.

[10] "Introdução à Internet - Escola de Verão 98" / Nota Técnica escrita por Denise Coutinho de A. Costa, Eduardo Fahr Pessôa, Fernanda Santoro Januzzi, Helio Sergio Nigri, João Marques Ferreira, Leonardo Ferreira Carneiro & Nilton Alves Jr.

[11] http://www.freesoft.org/CIE/Topics/79.htm.

[12] http://www.pcwebopaedia.com/IP_adress.htm.

[13] http://www.freesoft.org/CIE/Topics/83.htm.

[14] http://www.freesoft.org/CIE/Topics/85.htm.

[15] http://www.pcwebopaedia.com/*Telnet*.htm.

[16] http://www.pcwebopaedia.com/SMTP.htm.

[17] http://www.pcwebopaedia.com/NFS.htm.

[18] http://www.pcwebopaedia.com/DNS.htm.

[19] http://www.pcwebopaedia.com/IPX.htm.

[20] http://www.pcwebopaedia.com/SPX.htm.

[21] http://www.pcwebopaedia.com/Ethernet.htm.

[22] http://www.pcwebopaedia.com/token_ring_network.htm.

[23] http://www.pcwebopaedia.com/local_area_network_LAN.htm.

[24] http://www.pcwebopaedia.com/hub.htm.

[25] http://www.baynetworks.com/products/Papers/wp-primer.html.

[26] http://www.pcwebopaedia.com/flag.htm.

[27] http://www.pcwebopaedia.com/NetBeui.htm.

[28] http://www.pcwebopaedia.com/router.htm.

[29] http://www.scit.wlv.ac.uk/~jphb/comms/iproute.html.

[30] http://samba.anu.edu.au/cifs/docs/what-is-smb.html.