UMA INTERFACE DE SOQUETES PADRÃO UNIX BSD 4.2 PARA MICROCOMPUTADORES DO TIPO IBM PC

ALESSANDRO ANZALONI - INSTITUTO TECNOLÓGICO DE AERONÁUTICA
GILBERTO S. MAYOR JR. - INSTITUTO TECNOLÓGICO DE AERONÁUTICA

RESUMO

A versão BSD 4.2 do sistema operacional UNIX apresenta um núcleo de redes que implementa a família de protocolos TCP/IP. Existe ainda uma interface bastante amigável com os programas aplicativos conhecida como Soquetes. Este trabalho descreve a implementação desta interface de soquetes em microcomputadores do tipo IBM PC utilizando sistema operacional DOS. Deste modo processos sendo executados em um microcomputador do tipo IBM PC, requisitando serviços desta interface, poderiam se comunicar com processos sendo executados sobre o sistema operacional UNIX BSD.

Através dos serviços fornecidos pela Rede Nacional de Pesquisa (RNP) foi possível obter as informações necessárias à respeito da família de protocolos TCP/IP (rfc 790 - 796).

ABSTRACT

The Unix BSD 4.2 has a network kernel that implements the TCP/IP protocol family. There is also a friendly interface with the users programs called Sockets. This work describes the implementation of this interface in an IBM PC environment running DOS. Processes running in the IBM PC requesting services from this interface would be able to communicate with processes running in any computer with an UNIX BSD operational system.

The information about the TCP/IP protocols were obtained using the services provided by the National Research Network that conceives access to the BITNET network.

I. INTRODUÇÃO

Existe hoje nas universidades e empresas de engenharia uma crescente utilização de estações de trabalho autônomas interligadas em rede. Estas estações de trabalho são compostas geralmente por super-microcomputadores que utilizam quase sempre o sistema operacional UNIX padrão BSD. Esta versão do sistema operacional UNIX exibe como uma de suas principais qualidades a implementação de um núcleo de rede formado pela família de protocolos TCP/IP. Este núcleo possibilita a interligação destas estações de trabalho a redes do tipo Internet. Os programas aplicativos tem acesso à diversas funções deste núcleo através de uma interface bastante amigável conhecida como Soquetes.

No entanto, existe ainda hoje nestes mesmos locais, um outro ambiente de computação representado pelos diversos microcomputadores compatíveis com IBM PC utilizando sistema operacional DOS. Seria de grande utilidade possibilitar o acesso destes microcomputadores à rede descrita acima. O que se deseja é possuir uma rede local heterogênea onde processos sendo executados sobre o sistema operacional UNIX BSD possam se comunicar com outros processos sendo executados sobre o DOS. O que se pretende então neste trabalho é a implementação da família de protocolos TCP/IP e de uma interface compatível com a interface de Soquetes do UNIX BSD para microcomputadores compatíveis com IBM PC utilizando sistema operacional DOS.

A implementação do protocolo TCP/IP permitiria o acesso destes microcomputadores a redes do tipo Internet. A interface de Soquetes proporcionaria um ambiente amigável para desenvolvimento de aplicativos de redes. Possibilitaria ainda, a migração de alguns

programas aplicativos escritos originalmente para o ambiente de programação UNIX BSD, para os microcomputadores compatíveis com IBM PC.

II. CONCEITOS GERAIS

Inicialmente serão definidos conceitos essenciais ao entendimento deste trabalho. Esta seção descreve a família de protocolos TCP/IP [1], a interface de redes existente no UNIX BSD 4.2 [2] e o programa NCSA TELNET.

II.1 PROTOCOLOS TCP/IP

TCP/IP é uma família de protocolos desenvolvidos pela pela agência de pesquisa do governo norte-americano, DARPA (*Defense Advanced Research Program Agency*), que possibilita a interconexão de diversos tipos de redes, sejam estas redes locais, ou de longa distância. É definida uma unidade básica de transmissão - o datagrama - e a forma com que este é transmitida através de uma rede, o que assegura uma implementação totalmente independente de aspectos funcionais do *hardware* da rede. A cada computador é associado um endereço que é reconhecido universalmente por toda a rede. Cada datagrama carrega um endereço fonte e um endereço destino. O roteamento é feito ponto a ponto em computadores destinados para tal fim denominados *gateways*. Assim, os programas aplicativos enxergam o conjunto de redes interconectadas como se fossem uma única rede. Esta rede virtual recebe o nome de Internet.

Conceitualmente, a Internet provê três tipos básicos de serviços distribuídos hierarquicamente em níveis [3]. No nível mais baixo existe um serviço não confiável de entrega de datagramas. No nível acima, segundo nível, existe um serviço de transporte confiável, com conexão, de entregas de mensagens utilizado pela

maior parte dos programas aplicativos. No terceiro nível se encontram os programas de aplicação que requisitam serviços aos níveis inferiores.

II.1.1 INTERNET PROTOCOL - IP

O protocolo Internet, usualmente denominado IP, foi desenvolvido para ser utilizado em um sistema formado pela interconexão de diversas redes comutadas por pacotes. O protocolo IP é responsável pela definição do datagrama e a forma exata com que este passa através da rede. Ele provê a transmissão de datagramas de um computador fonte a um computador destino possibilitando a fragmentação e remontagem dos datagramas caso seja necessário. É definido um conjunto de regras que especificam como os datagramas devem ser processados e como os erros devem ser tratados nos *gateways* e computadores.

Não existe um mecanismo de confirmação fim-a-fim, nem mecanismos de controle de fluxo e seqüenciamento. Isto significa que pacotes podem ser perdidos, duplicados ou entregues fora de ordem. O serviço é denominado sem conexão pois cada datagrama é processado de forma independente dos demais. O protocolo IP na realidade provê um serviço básico de transmissão de datagramas sobre o qual se apóiam os demais serviços da rede.

II.1.2 USER DATAGRAM PROTOCOL - UDP

O protocolo UDP provê um serviço de transmissão de mensagens não confiável, utilizando-se dos serviços prestados pelo protocolo IP, para transmitir as mensagens. Basicamente o protocolo UDP possibilita que vários processos em um mesmo computador possam simultaneamente requisitar serviços ao protocolo IP para enviar e receber mensagens.

O protocolo UDP utiliza um endereço de 16 bits denominado porta para definir um único processo em um computador. Uma porta identifica o processo ao qual um datagrama deve ser entregue. Cada mensagem UDP contém uma porta fonte e uma porta destino.

II.1.3 TRANSMISSION CONTROL PROTOCOL - TCP

TCP é um protocolo altamente confiável, com conexão, utilizado na comunicação fim-a-fim entre computadores. O protocolo TCP foi desenvolvido inicialmente para ser utilizado em um sistema hierárquico de protocolos o que possibilita a sua utilização em diferentes sistemas de comunicação. A utilização do protocolo TCP pressupõe a existência de um protocolo de nível inferior capaz de prover um serviço não confiável de entrega de datagramas - serviço este executado pelo protocolo IP em uma rede Internet.

II.1.3.1 CONEXÕES

Para possibilitar que muitos processos em um mesmo computador possam requisitar serviços ao protocolo TCP simultaneamente, é provido um conjunto de endereços ou portas em cada computador. A tupla formada por um endereço Internet que identifica um computador e uma porta que representa um processo neste computador é denominada soquete. Um par de soquetes identifica unicamente uma conexão. Uma conexão pode ser usada para transmitir dados em ambas as direções entre dois processos distintos. Um soquete pode ser utilizado em várias conexões distintas. Para se estabelecer uma conexão, um usuário realiza uma chamada a uma rotina denominada OPEN. A chamada a esta rotina retorna um nome local para esta conexão a ser utilizado em subsequentes referências a mesma.

II.2 INTERFACE DE REDES DO UNIX BSD 4.2

A partir da versão 4.2 o UNIX BSD passou a apresentar uma interface de comunicação entre processos que possibilita que processos localizados em diferentes computadores possam se comunicar de modo transparente. Esta interface de rede baseia-se em uma entidade abstrata denominada soquete. Um soquete é uma generalização do mecanismo de gerenciamento de arquivos do UNIX e provê um ponto terminal de comunicação em um computador. Esta interface possui um razoável grau de complexidade pois foi desenvolvida com o objetivo de possibilitar aos programas aplicativos acesso a todos os possíveis protocolos existentes em um computador através da mesma interface - nesta época ainda não se tinha idéia da futura aceitação universal dos protocolos TCP/IP. Deste modo ao enviar uma mensagem, um programa aplicativo deve não somente especificar o endereço do processo destino mas também deve informar a qual protocolo refere-se aquele endereço. Diferentes protocolos podem estar ativos em um computador ao mesmo tempo. Deste modo um endereço de um soquete completamente especificado é formado pela tupla <*protocolo,endereço local da porta, endereço Internet do computador*> [4].

I1.3 NCSA TELNET

O programa NCSA TELNET é um programa de domínio público, desenvolvido no Centro Nacional de Aplicações de Supercomputação (*National Center for Supercomputing Applications*) em Champaign nos Estados Unidos, que implementa a família de protocolos TCP/IP. O programa NCSA TELNET foi desenvolvido em camadas hierárquicas, seguindo o modelo OSI, que implementam os protocolos da família TCP/IP e alguns serviços à nível de sessão.

O programa NCSA TELNET foi dividido em módulos funcionalmente independentes o que tornou possível a separação e utilização somente dos módulos responsáveis pela implementação da família de protocolos TCP/IP.

III. IMPLEMENTAÇÃO DOS SOQUETES EM AMBIENTE DOS.

Nesta seção serão definidos os objetivos a serem alcançados pelo sistema. Pretende-se ainda descrever a arquitetura do sistema e alguns detalhes significativos da implementação.

III.1 OBJETIVOS DO SISTEMA

A interface de soquetes do UNIX BSD permite que vários processos residentes em diferentes computadores em uma Internet possam se comunicar de modo transparente. Os programas de aplicação possuem ainda diversos tipos de serviços disponíveis dependendo do grau de confiabilidade exigido pelo mesmo.

O trabalho propõe a implementação desta interface de soquetes padrão UNIX BSD 4.2 utilizando a família de protocolos TCP/IP, sobre o sistema operacional DOS, a ser utilizado em microcomputadores compatíveis com IBM PC. Desta forma seria possível escrever programas aplicativos a serem executados em microcomputadores pessoais do tipo IBM PC que se comunicassem com processos sendo executados em estações de trabalho utilizando o sistema operacional UNIX BSD. Seria possível ainda o transporte de diversos aplicativos escritos inicialmente no ambiente UNIX BSD para o ambiente DOS. Seria criada então uma rede heterogênea formada por microcomputadores pessoais do tipo IBM PC utilizando sistema operacional DOS e estações de trabalho utilizando o sistema operacional UNIX, ambas as partes comunicantes entre si. Um usuário

poderia acessar os diversos serviços fornecidos por uma estação de trabalho através de um microcomputador do tipo IBM PC. O sistema provê também a possibilidade de requisições simultâneas de pedidos de serviços prevendo-se a migração futura do mesmo para um ambiente multi-tarefa. A idéia central do projeto baseia-se na necessidade da criação de um núcleo de rede que possibilitaria o desenvolvimento de um ambiente de rede semelhante ao existente no UNIX BSD em computadores pessoais do tipo IBM PC utilizando algum sistema operacional multitarefa.

III.2 INTERFACE DE SOQUETES

Uma das dificuldades apresentadas pelo projeto foi a definição de um conjunto mínimo, não redundante e auto-suficiente de rotinas que possibilitassem que qualquer programa escrito originalmente utilizando as rotinas da interface de soquetes do UNIX BSD pudesse ser transportado para esta nova interface sem grandes transformações [5].

O conjunto final de rotinas escolhidas pode ser dividido em cinco grandes grupos:

1. Rotinas de manuseio de soquetes - abertura , assinalamento de nome e fechamento de soquetes.
2. Rotinas de transmissão e recepção de datagramas.
3. Rotinas de manuseio de conexões - estabelecimento e término de conexões, transmissão e recepção de mensagens através de uma conexão, etc.
4. Rotinas de uso geral que fornecem ao usuário transparência quanto ao modo de endereçamento da rede.
5 Rotinas de uso geral que fornecem ao usuário transparência quanto aos detalhes da arquitetura do IBM PC.

Determinadas opções existentes na interface de soquetes do UNIX, tal como a transmissão prioritária de dados (*out-of-band data*), necessitaram ser eliminadas devido a impossibilidade ou enorme complexidade de implementação sem uma devida recompensa. Decidiu-se no entanto não somente transportar apenas as rotinas da interface de soquetes do UNIX BSD mas tentar recriar todo o ambiente desta interface em um microcomputador do tipo IBM PC.

No sistema operacional UNIX BSD, o usuário tem acesso a vários programas utilitários de rede que lhe fornecem diversos serviços tais como correio eletrônico, *login* remoto, etc. Estes aplicativos utilizam os serviços prestados pelo núcleo de redes - interface de soquetes. Seria de grande interese recriar todo este ambiente de rede em um microcomputador do tipo IBM PC de tal modo que um usuário pudesse conectar o seu microcomputador a uma rede contendo estações de trabalho utilizando o sistema operacional UNIX BSD e utilizar os diversos serviços existentes nesta rede. No entanto, para tornar esta tarefa possível, é preciso atender às seguintes necessidades:

. criar um núcleo multitarefa sobre o sistema operacional DOS.

. criar um núcleo de rede que permita a requisição simultânea de serviços por diferentes processos.

A primeira necessidade pode ser satisfeita utilizando alguma versão do DOS multitarefa das diversas existentes no mercado, ou algum programa aplicativo que permita a execução simultânea de diversos processos, ou até mesmo implementando algum núcleo multitarefa.

O sistema desenvolvido possuiu como objetivo fundamental atender ao segundo requisito. Deste modo a arquitetura do mesmo deveria ser concebida visando a sua futura utilização em um ambiente compatível com o DOS mas que apresentasse a capacidade de executar diversos processos simultaneamente - multitarefa. Assim, a arquitetura implementada deve permitir a requisição de serviços por diversos processos ao mesmo.

Inicialmente pensou-se na implementação de uma biblioteca contendo as diversas funções do módulo de redes a ser utilizada pelos programas aplicativos. No entanto esta alternativa apresenta diversos inconvenientes em um ambiente multitarefa tal como a presença de diversas instâncias do mesmo código na memória do computador. Apresenta ainda a desvantagem de que uma pequena alteração no programa ou na definição de alguma rotina de interface de soquetes do UNIX BSD resultaria na manutenção de toda a biblioteca o que diminui a portabilidade do sistema e dificulta a sua manutenção.

Optou-se então pela criação de um módulo prestador de serviços residente e de uma pequena biblioteca a ser utilizada pelos programas aplicativos cuja função é requisitar serviços ao módulo residente. Esta pequena biblioteca contém todas as rotinas do módulo de emulação de soquetes sendo os nomes e os parâmetros destas rotinas idênticos aos das rotinas da interface do UNIX BSD. Todas as rotinas foram escritas de tal modo que uma chamada a uma das rotinas se faça de modo idêntico a chamada à mesma rotina no ambiente UNIX BSD. Ao utilizar esta biblioteca o usuário poderá escrever os seus aplicativos como se estivesse utilizando a biblioteca de redes do UNIX BSD. Todas as rotinas responsáveis pela

execução das funções dos protocolos TCP/IP se encontram no módulo residente. Deste modo, as rotinas desta biblioteca não executam as funções de rede mas apenas requisitam serviços ao módulo prestador de serviços residente. Esta requisição é feita através da geração de uma interrupção de *software* [6].

No entanto, a possibilidade da requisição simultânea de serviços provocou a existência de um problema clássico de mútua exclusão - região crítica. Como o código residente não é reentrante, e diversos processos podem tentar acessá-lo simultaneamente de maneira aleatória, é necessário estabelecer critérios que impossibilitem que dois processos iniciem a execução deste código de modo simultâneo. Decidiu-se então implementar utilizando as interrupções do DOS um mecanismo de semáforos que impedissem o acesso simultâneo de vários processos à região crítica.

III.3.1 IMPLEMENTAÇÃO DO MECANISMO DE SEMÁFOROS

A principal razão da escolha de se implementar este sistema utilizando semáforos foi o alto grau de portabilidade que o sistema deveria possuir. A implementação foi desenvolvida em um ambiente mono-tarefa mas visando a sua futura utilização em um ambiente multi-tarefa.

A adaptação deste sistema a qualquer ambiente multi-tarefa ocasionará a adaptação do mesmo às primitivas de sincronização de processos existentes no novo ambiente. É necessário portanto, utilizando o mecanismo dos semáforos somente reescrever, ou apenas adaptar, as rotinas dos semáforos às primitivas do sistema nativo. Esta escolha no entanto criou a necessidade de se implementar um mecanismo de semáforos sobre o DOS, não existente no

mesmo, visto que este é um sistema mono-tarefa. Este mecanismo deveria ser robusto o suficiente para permitir a sua utilização em qualquer novo ambiente desejável, e eficiente de modo que o seu funcionamento não afetasse o desempenho final do sistema. A melhor alternativa de se implementar o mecanismo de semáforos é utilizando o mecanismo de interrupções do microcomputador IBM PC. O processo de atendimento de uma interrupção no microcomputador PC pode ser descrito pelos seguintes passos [7]:

. salva flags e endereço de retorno na pilha.

. desabilita interrupções.

. desvia para endereço apontado pelo vetor de interrupções.

Caso a rotina de tratamento da interrupção não reabilite os flags de interrupção estas estarão bloqueadas durante toda a execução desta rotina. Somente serão habilitadas após a execução da instrução de retorno da interrupção - iret. Deste modo pode-se garantir que somente um processo é capaz de executar esta rotina de cada vez. Esta característica foi utilizada na implementação do mecanismo de semáforos. Para ler ou alterar o valor de um semáforo, um processo deve realizar chamadas às rotinas correspondentes, wait e *signal* [8]. Estas rotinas se encontram residentes e uma chamada à mesmas só pode ser feita através da geração de um sinal de interrupção o que garante que somente um processo de cada vez está lendo ou alterando o valor de um semáforo.

Cabe ainda ressaltar que este mecanismo não impõe nenhuma ordem de acesso à região crítica quando dois ou mais processos concorrem pelo mesmo recurso. Ou seja, não se garante que o primeiro processo a requisitar acesso à região crítica quando esta se encontra bloqueada seja efetivamente o primeiro a recebê-lo.

IV. DIFICULDADES ENCONTRADAS

Uma das maiores dificuldades encontradas na implementação desta interface foi a identificação correta do comportamento das rotinas da interface de soquetes do UNIX BSD diante das diversas situações que eventualmente podem ocorrer. O mapeamento deste comportamento, principalmente no que se refere às temporizações (*timeout*), foi realizado através de sucessivos testes com a interface de soquetes do UNIX BSD. Diversos programas aplicativos foram escritos nesta interface, simulando-se várias condições de erro, com a finalidade de se identificar corretamente quais decisões deveriam ser tomadas.

Um outro objetivo primordial desta implementação foi a de escrever um código com um alto grau de portabilidade. Deste modo tentou-se não utilizar qualquer função pertencente a um único compilador. Assim, o transporte deste sistema para qualquer outro ambiente de computação implica somente na reescrita das rotinas dependentes de *hardware*. Estas rotinas compõe basicamente o módulo da interface com a placa de rede - *driver* da placa de rede. No entanto, cabe ressaltar que o desenvolvimento deste módulo é particularmente trabalhoso devido à necessidade deste código possuir um alto desempenho - algumas rotinas devem necessariamente ser escritas em *Assembly*. Seria necessário ainda reescrever as rotinas dos semáforos visto que estas utilizam as interrupções do DOS.

IV. CONCLUSÃO

O sistema implementado se apresenta na fase final de testes. Existe na instituição um microcomputador do tipo IBM PC ligado

através de uma rede Ethernet a um super-microcomputador Tektronics. Processos desenvolvidos no microcomputador PC utilizando a biblioteca implementada se comunicam com outros processos sendo executados no Tektronics.

Diversos programas aplicativos que realizam pequenos serviços tais como transferência de arquivos entre dois computadores, escritos originalmente utilizando a biblioteca de soquetes do UNIX BSD 4.2, já foram transportados para o microcomputador IBM PC utilizando a biblioteca implementada.

Projetos futuros incluem o desenvolvimento de diversos aplicativos, tais como telnet, mail,etc, assim como do protocolo RPC, utilizando esta interface de soquetes.

REFERÊNCIAS

[1] POSTEL, J. B. "DARPA Internet Program Internet and Transmission Control Specifications", Defense Advanced Research Projects Agency, Arlington, USA, 1981.

[2] TEKTRONIX, Inc. "UTek Command Reference", Tektronix Inc, USA, 1986.

[3] COMER, D.E. "Internetworking with TCP/IP Principles, Protocols and Architeture", Prentice Hall Inc., Englewoods Cliffs, USA, 1988.

[4] SUN, Microsystems Inc. "Network Programming", Sun Microsystem Inc., USA, 1987.

[5] DIGITAL, Equipament Corp. "DEC-net DOS Programmer's Reference Manual", Digital Equipament Corp., USA, 1986.

[6] DUNCAN, R."Advanced Msdos" Microsoft Press, Redmond, USA, 1986.

[7] RECTOR, R. e ALEXY, G. "The 8086 Book",Osborne McGraw-Hill, Berkeley, USA, 1980.

[8] ARI, M.B. "Principles of Concurrent Program", Prentice-Hall International Inc.,Englewood Cliffs, USA, 1982.