

Uma Ferramenta para Monitoração de Tráfego NFS em Redes Unix

Maurício Atanásio Paulo Lício de Geus

Departamento de Ciência da Computação
Universidade Estadual de Campinas
13081-970 Campinas, SP

E-mail: {atanasio, paulo}@dcc.unicamp.br

Resumo

Este documento apresenta uma ferramenta para monitorar o tráfego em uma rede Unix, caracterizando esse tráfego por protocolo, identificando a origem e destino das chamadas, especificando o número de pacotes e quantidade de bytes por protocolo e entre duas estações quaisquer da rede. Trata em especial do tráfego NFS [Sandberg85] [Sun94], identificando a origem da chamada e o *filesystem* acessado, número de acessos ao *filesystem* e o total de bytes nestes acessos.

Abstract

This document presents a tool to monitor the traffic in a Unix network, that: i) classifies traffic by protocol; ii) identifies the source and destination of calls; iii) reports the number of packets and bytes, by protocol and between any two machines on the network. It especially treats NFS traffic, identifying the call and accessed filesystem, reporting the number of accesses and bytes in these accesses.

1 Introdução

As redes de computadores tiveram um crescimento vertiginoso nos últimos anos [Tanenbaum92]. É muito comum nos dias atuais redes de computadores com várias centenas de usuários, muitos desses sendo usuários remotos. Há alguns anos, o desafio na administração de redes não é somente implantá-las e mantê-las funcionando isoladamente, mas interconectar diversas redes locais executando diferentes protocolos.

Uma razão inicial para interconectar computadores em redes é para fornecer serviços centrais a serem compartilhados. Os softwares aplicativos são projetados para oferecer uma larga faixa de opções aos usuários. Isto faz com que fiquem cada vez mais volumosos. Além disso devido ao grande número de usuários por máquina, o tráfego entre máquinas servidoras e clientes é excessivo. A vazão exigida pela rede é cada vez maior. A vazão líquida máxima na rede mais comum (Ethernet) é de aproximadamente 1,18 MB/s ¹

¹Megabytes por segundo

[Geus90] para um canal que continua constante a 10 mb/s ². E o número de *workstations* nas redes continua a crescer: servidores são configurados com mais clientes e estes, cada vez mais exigentes, sobrecarregam severamente os servidores, influenciando no tráfego entre os mesmos e no tráfego da subrede à qual estão diretamente conectados.

Todo esse tráfego degrada o desempenho da rede. Uma solução é criar redes compostas por outras pequenas redes. Com as subredes, um tráfego local a essa rede não interfere no das outras, implicando numa melhora razoável do desempenho global. São bastante comuns redes heterogêneas compostas de vários segmentos de redes, que podem diferir na topologia ou protocolos de comunicação. Como exemplo de problemas surgidos nesse ambiente, temos [Byte91]:

- Que softwares estão instalados e em quais *filesystems* ?
- Que hardware compõe a rede ? E os equipamentos de interconexão de redes ?
- Como é a utilização média da rede ? Qual a variação durante um dia de trabalho ?
- Qual ou quais as aplicações dominantes na rede ?
- Muitas aplicações residem em servidores remotos. Como acessar e controlar esse acesso a esses sistemas?

Redes Ethernets possibilitam uma coleta de dados de forma a caracterizar o tráfego ao longo da rede/subrede, pois este tráfego é propagado para todas as máquinas conectadas nessa rede/subrede.

Com a necessidade de identificar e caracterizar o tráfego nas subredes e na rede do DCC/IC/UNICAMP ³ como um todo, inicialmente, a idéia seria a de se utilizar algum software já existente, de domínio público. O que se precisava era de um produto que pudesse identificar gargalos no sistema, assim como padrões de uso de programas e de dados.

Há várias ferramentas que realizam a tarefa de monitoração de tráfego em redes ethernet, como em [McCanne94], [Waldbusser91], [Smith90], [Mankin89], [Curry93], [Shipley91], [Blaze92] e [Kislitzin90]. Essas ferramentas são úteis na observação e coleta de tipos específicos de pacotes que trafegam na rede, sendo que algumas ainda realizam um gerenciamento mais completo sobre a rede ([Waldbusser91]). Outras tratam em específico do tráfego NFS ([Blaze92]), coletando e desenvolvendo estatísticas sobre esse tráfego.

Entretanto, mesmo que essas ferramentas apresentem nos relatórios gerados algum tipo de informação útil para o estudo a ser feito no ambiente do DCC, notou-se que ainda faltava algum tipo de informação que se precisava ter, por exemplo, caso haja um tráfego NFS pesado a um determinado *filesystem* num determinado servidor, qual a origem desse tráfego?

A dificuldade de se decidir por algum dos softwares analisados e com receio de perder muito tempo na procura de outros que atendessem às exigências verificadas, levou à decisão de planejar e implementar tal ferramenta, composta por um conjunto de *scripts*, implementados em PERL ⁴ [Schwartz91], que filtram informações geradas pelo comando

²megabits por segundo

³Departamento de Ciência da Computação - Instituto de Computação - UNICAMP

⁴Practical Extraction and Report Language

snoop [Sun95], do sistema operacional Solaris, da Sun Microsystems, Inc., especificando os protocolos, as origens/destino das chamadas e identificando os *filesystems* acessados via NFS, para um estudo de soluções de redução do tráfego NFS ao longo da rede, com soluções de cacheamento para disco local e replicação. Esses *scripts* serão apresentados nas secções seguintes.

2 A Rede do DCC

A rede de computadores do DCC, o caso a ser estudado, é composta de várias subredes, onde se destacam máquinas unix (rodando SunOS e Solaris), uma rede Novell, uma rede de máquinas WindowsNT, e uma rede de máquinas MacOS. A rede é administrada seguindo os seguintes objetivos [Geus94]:

- Simplicidade e flexibilidade de administração. Isso é possível devido ao uso extensivo de NFS, NIS e *automount*.
- Filesystems consistentes na visão do usuário através da rede (NFS, *automount*).

O NIS é usado para propagar a configuração e informações do sistema através de todas as máquinas Unix. O *automount* usa tais informações para disponibilizar uma hierarquia virtual única a todas as máquinas Unix.

Como todo serviço de rede, há 3 pontos onde podem ocorrer falhas no NFS: o servidor, o cliente e a rede propriamente dita. Além disso, há vários fatores que provocam queda de desempenho no NFS [Sun91] e que se aplicam à rede do DCC, entre outros:

- Redes congestionadas que impedem pacotes de alcançarem o cliente ou servidor;
- Servidores lentos que falham em responder dentro do limite de tempo dos RPC's;

Os usuários da rede se compõem de professores pesquisadores, estudantes de pós-graduação e graduação, e por funcionários da administração do Instituto de Computação.

3 A Hierarquia de Filesystems do DCC

A abordagem adotada no DCC é de isolar aplicações do sistema de outros softwares [Geus94]. O diretório */usr/local* foi dividido entre várias categorias de software. Por exemplo:

- */n/net* - todo software de rede, tais como: mosaic/www, gopher, ftp, recursos de DNS, etc.
- */n/dtp* - contém FrameMaker, TeX, aplicativos dvi e PS, entre outros.
- */n/draw* - contém AutoCAD, xfig e pacotes relacionados a desenho.
- */n/gnu* - todos os aplicativos GNU.
- outros.

Localmente, todas as hierarquias são armazenadas sob *filesystems* cujos nomes ajudam a configurar um esquema de *backup* da rede, como */l/repXX*, */l/homeXX*, */l/projXX*. */l* representa *filesystems* locais. *Rep* para repositórios de propósito geral, *home* para *homedirs* de categoria de usuários (*/h/staff*, */h/pos*, */h/grad*, etc) e *proj* para hierarquias de projetos (*/p/dicio*, */p/vid*, */p/db*, etc).

O *automount*, através do NFS e de mapas propagados pelo NIS, monta a hierarquia correta.

A hierarquia de *filesystems* do DCC oferece total transparência aos usuários. Devido à falta de espaço em disco, no presente momento é impossível utilizar soluções de replicação de *filesystems*. A replicação traria como vantagem imediata o aumento de confiabilidade no sistema. Com replicação, se houvesse um *crash* em um servidor, poder-se-ia montar a hierarquia através de outro servidor onde essa mesma hierarquia estivesse replicada. Uma outra possibilidade para o uso de replicação seria a restauração *on-line* de um determinado *filesystem* sem recorrer a *backup* em fita. Atualmente não há nada replicado na rede do DCC, além do sistema operacional.

4 Visão Geral da Ferramenta

A ferramenta como um todo consiste de um conjunto de *scripts* implementados em *PERL*. A escolha de *PERL* se deve ao fato de que ela é mais poderosa que os *shells*, permite uma manipulação fácil de textos, arquivos e processos e à sua grande utilização nessa área, entre outras características [Schwartz91].

O *script monitord* é responsável pela tarefa de monitorar o tráfego ao longo do *backbone* da rede e entre subredes. Esse *script* será executado ao longo do dia em intervalos regulares, diversas vezes, em um ou mais servidores selecionados, de forma a representar com certa fidelidade o tráfego real de um dia de trabalho no DCC.

O *script nfs_logd* é executado durante o período noturno, usando um dos relatórios gerado pelo *monitord*. Sua função principal é separar os acessos feitos a um servidor NFS, com os respectivos *filehandles* dos *filesystems* acessados nesse servidor, com a identificação da origem da chamada, o número de vezes que esse *filehandle* foi acessado com a respectiva quantidade em bytes dos acessos realizados e após isso, executa remotamente no servidor em questão o *script* *busca_fsd*.

O terceiro *script*, *busca_fsd*, é executado remotamente em cada servidor NFS que foi acessado durante o dia e identifica os *filesystems* acessados, gerando um relatório final com o *pathname* do arquivo/diretório acessado, a origem do acesso e com o número de vezes que foi acessado ao longo do dia.

O *script monitord*

A monitoração da rede é controlada pelo *script* chamado *monitord*, composto de uma inicialização e três subrotinas principais: *processa_interface*, *processa_relatório* e *processa_nfs*.

O *monitord* pode monitorar uma ou mais interfaces da máquina. Assim sendo, ele inicia com a identificação da máquina e de suas interfaces. Pode-se selecionar a interface a ser monitorada, caso se deseje analisar o tráfego de uma determinada subrede. Ele

controla o tempo de monitoração, dependendo se a rede está lenta ou não, aumentando ou diminuindo esse tempo, dependendo da situação verificada.

Ao longo do dia, a subrotina **processa_interface** é chamada a cada início do *monitord*. Através desta subrotina, a monitoração é feita pelo *snoop*. O *snoop* monitora a interface e gera dois arquivos para processamento *off-line*. O primeiro arquivo identifica a origem/destino da chamada, com o protocolo e quantidade em bytes. É feito um resumo do tráfego monitorado, gerando um relatório gravado em disco, dividido em três partes:

- Nome da máquina, interface, número total de pacotes, quantidade total de bytes e tráfego por segundo dessa monitoração.
- Relação de protocolos identificados, especificando o número de pacotes e o total de bytes por protocolo e respectivas porcentagens do total.
- Relação da origem/destino das chamadas (nomes das máquinas), com o número de pacotes, total de bytes e suas respectivas porcentagens do total.

Esses relatórios gravados em disco pela rotina *processa_interface*, serão resumidos num único relatório ao final de cada execução do *monitord*.

O Segundo arquivo gerado pelo *snoop*, separa as chamadas NFS para posterior processamento pela subrotina *processa_nfs*. A rotina *processa_nfs* lê o arquivo contendo as chamadas NFS. Desse arquivo, ela identifica a origem e destino da chamada e o *filehandle* do *filesystem* acessado, com um contador que determina o número de vezes que o *filehandle* foi acessado. Esses dados são gravados em disco e utilizados pelo programa *nfs_logd* que, durante o período noturno, vai chamar remotamente o *script* *busca.fsd* em cada servidor NFS para identificar os *filesystem* acessados.

A subrotina *processa_relatório* é executada ao final do *monitord*, fazendo um resumo do relatório feito no intervalo anterior e os relatórios feitos na execução atual. Ao final do dia, tem-se a monitoração do tráfego diário.

O *script* **nfs_logd**

O *nfs_logd*, como dito anteriormente, é executado no período noturno, para não afetar o desempenho das máquinas nos horários de pico. Ele inicia lendo o arquivo gerado pela subrotina *processa_nfs* do *monitord*, identificando os servidores NFS acessados durante o dia e separando os *filehandles* acessados em cada servidor.

Após gravar em uma hierarquia vista por todo o sistema os *filehandles*, com a origem e respectiva quantidade dos acessos, é executado um *remote shell* no servidor em questão. Este faz uma chamada ao *busca.fsd*, que será executado e é o responsável pela identificação do *filesystem* acessado e pela elaboração do relatório final.

Caso o *nfs_logd* tenha problemas em acessar um determinado servidor remotamente, após os acessos aos demais servidores, ele faz uma verificação para checar se algum servidor não foi acessado, deixando de relatar certos *filesystems* naquele servidor. Caso isso tenha acontecido, repete o processo de *remote shell* e invocação do *busca.fsd*.

O script `busca.fsd`

O `busca.fsd` se inicia com a leitura do arquivo gravado pelo `nfs_logd`. Após essa leitura ele começa a converter os `filehandles` nos `inodes` dos `filesystems` acessados. Com esses `inodes`, uma subrotina `procura_fh` é chamada para procurar os `inodes` do ponto de montagem, que também é identificado através do `filehandle` retornado pelo `snoop` no script `monitord`.

Tendo o `inode` do ponto de montagem, o `procura_fh` chama a subrotina `procura_dir` e dentro desta, a subrotina `pega_fh` é a responsável pela busca no `filesystem` do servidor para encontrar o `inode` procurado. Assim pode retornar o `path` do arquivo/diretório, gravando em disco o relatório final, que especifica a origem/destino da chamada, o `path` completo do arquivo/diretório acessado, quantidade em bytes e o total de acessos.

5 Conclusão

Como é de se esperar, a quantidade e distribuição do tráfego deve variar consideravelmente. O processo de monitoração está se iniciando e espera-se que em breve se tenha uma idéia do tráfego entre subredes, assim como padrões de uso de programas e dados.

A monitoração está sendo feita com intervalos regulares de 15 minutos, durante o dia, com aproximadamente 30 s de monitoração em cada intervalo. É feita no `backbone` da rede e nas subredes principais, completando o quadro de monitoração. Vale ressaltar que a monitoração em cada segmento é feita em intervalos diferentes, para não haver possibilidade de repetição do tráfego monitorado. Para se ter uma idéia mais realista do tráfego, esse tempo de monitoração pode ser aumentado.

Tendo essas informações, pode-se saber como e por quais máquinas certos `filesystems` são acessados. Com soluções de replicação e cacheamento para disco local dos arquivos mais acessados em máquinas que rodam SunOS 4.x, pode-se reduzir o tráfego NFS na rede. Além de possibilitar a otimização do uso do CacheFS⁵ [Sun95] em máquinas rodando o `Solaris`. Isto permite aumentar a disponibilidade do sistema (com a replicação) e o desempenho (tendo menos bytes trafegando ao longo da rede e menos acessos aos servidores remotos). Este é o próximo trabalho a ser feito. O cacheamento manual será feito para máquinas SunOS 4.x, já que o `Solaris` possui o CacheFS, do qual faremos uso.

O `monitord` apresenta uma limitação: o uso do `snoop` só é possível em máquinas com o sistema operacional `Solaris`. O `etherfind` [Sun90] (similar do `snoop` para SunOS 4.x) não fornece relatórios tão detalhados como o `snoop`, inviabilizando sua utilização.

Para portar o `monitord` para outro sistema, haveria a necessidade de se ter um similar do `snoop` nesse sistema. Uma opção seria o de se usar o `nfswatch` [Curry93], que é portátil para vários sistemas, com o inconveniente de que o `nfswatch`, apesar de identificar os `filesystems` acessados e a quantidade de acessos aos mesmos, não relaciona a origem dos acessos.

⁵Cache File System

Apêndice A

- Exemplo de parte do relatório gerado pelo **monitord** após um dia de monitoração na rede do DCC. A primeira parte do relatório (primeira linha) mostra a máquina que foi feita a monitoração (máquina paraná), a interface monitorada, o dia, total de pacotes, total de bytes, o tráfego médio em bytes por segundo e o intervalo médio de monitoração em cada execução do **monitord**. A segunda parte, indicada pela palavra *proto*, informa os protocolos, número de pacotes e porcentagem e o total de bytes com respectiva porcentagem do total. A terceira parte, indicada pela palavra *host* no início de cada linha, informa a origem/destino da chamada, com o número de pacotes, total em bytes e respectivas porcentagens do total.

parana	le0	8/Mar/96	252000	196254896	540.072	18.754947
proto	ARP		65	0.0258	3864	0.0020
proto	DNS		499	0.1980	56748	0.0289
proto	FINGER		12	0.0048	720	0.0004
proto	FTP		174	0.0690	12665	0.0065
proto	ICMP		182	0.0722	21004	0.0107
proto	IP		5	0.0020	300	0.0002
proto	NFS		96029	38.1067	107524322	54.7881
proto	NIS		2702	1.0722	484052	0.2466
proto	RIP		184	0.0730	73744	0.0376
proto	RLOGIN		1812	0.7190	135098	0.0688
proto	RPC		1030	0.4087	632470	0.3223
proto	TCP		113786	45.1532	81054227	41.3005
proto	TELNET		4453	1.7671	357141	0.1820
proto	UDP		197	0.0782	96592	0.0492
proto	XWIN		26570	10.5437	4434322	2.2595
host	parana-ecl		4	0.0016	268	0.0001
host	pascal2-amsterdam		240	0.0952	216480	0.1103
host	pascal2-atibaia		20	0.0079	2372	0.0012
host	t1l-grande		39	0.0155	5010	0.0026
host	t1l-jaguari		389	0.1544	402498	0.2050
host	yes-grande		3	0.0012	252	0.0001

Apêndice B

- Exemplo de parte do relatório gerado pelo **busca_fsd**. Informa a máquina que fez a chamada NFS, o destino da chamada, o diretório/arquivo acessado, número de acessos e o total de bytes trafegados nesses acessos.

atlantic = atibaia = 1 = /l/proj02/ahand	278	40596	
jaguari = atibaia = 1 = /l/proj02/ahand	7	1066	
ttl = atibaia = 1 = /l/proj02/ahand	4	584	
euler = atibaia = 1 = /l/proj02/ahand	2	292	
grande = fermat = 1 = /l/home83/lac3/tomasz/myroot		1	162
cmbm = grande = 1 = /l/rep09/gnu-all	30	4620	
marumbi = grande = 1 = /l/rep09/net	7	1182	
stack = grande = 1 = /l/rep09/net	4	656	
jaguari = grande = 1 = /l/rep09/net/Solaris	3	526	
pascal2 = grande = 1 = /l/rep09/net/Solaris	1	174	
euler = grande = 1 = /l/rep09/net/Solaris	1	166	
marumbi = grande = 0 = /l/home00/ic/felipeal/out/mar22.tgz		30	42640
ttl = grande = 1 = /l/home00/ic/marcyn	3	446	

References

- [Tanenbaum92] Andrew S. Tanenbaum. *Computer Networks*. Prentice-Hall, 3rd. Ed., 1992.
- [Sun91] Sun Microsystems. *Sun Performance Tuning Overview*. Sun Microsystems Technical White-Paper, 1991.
- [Sandberg85] Russel Sandberg. The Sun Network File System: Design, Implementaton and Experience. Proceedings of Usenix Conference. Summer 1985.
- [Sun95] Sun Microsystems. *The Cache File System in Solaris 2.5 System Administration Guide, Volume I*. Novembro 1995.
- [Sun94] Sun Microsystems. *NFS: Network File System - Version 3 Protocol Specification*. 1994.
- [Geus90] Paulo Lício de Geus. *Approaches to Live Image Transmission between Workstations over Limited-Bandwidth Networks*. Tese de Doutorado. University of Manchester-Inglaterra. Maio 1990.
- [Geus94] Paulo Lício de Geus. *An Overview of the Computing Facilities at DCC-IMECC-UNICAMP*. Julho 1994.
- [Byte91] Revista Byte. *Network Management*. Março 1991.
- [Schwartz91] Larry Wall e Randal L. Schwartz - *Programing Perl* - O'Reilly & Associates, Inc. - 1991.
- [Sun90] Sun Microsystems. *etherfind* in SunOS-4.1 Reference Manual AnswerBook, 1990.
- [Sun95] Sun Microsystems - *snoop* in Solaris Reference Manual AnswerBook 2.5, 1995.
- [Waldbusser91] Steven Waldbusser. *Remote Network Monitoring MIB - RFC 1271*. Network Working Group - Carnegie Mellon University. Novembro 1991.
- [Shiple91] Carl Shipley e Chingyow Wang - *Monitoring Activity on a Large Unix Network wiht Perl and Syslog*. Proceedings of LISA V Conference - outubro 1991.
- [Curry93] David. A. Curry e Jeff Mogul- *Nfswatch Reference Manual Pages*. Março 1993.
- [McCanne94] Steve McCanne e Craig Leres - *TCPDUMP 3.0.2* . Lawrence Berkeley Laboratory - Network Research Group. Junho 1994.
- [Smith90] Randy Smith - *NETMON User Commands*. University of Minnesota. 1990.
- [Mankin89] Allison Mankin - *NETMON User's Manual* - MITRE Corporation. 1989.
- [Kislitzin90] Katy Kislitzin - *Network Monitoring by Scripts*. Proceedings of LISA IV Conference - outubro 1990.
- [Blaze92] Matt Blaze - *NFS Tracing By Passive Network Monitoring*. Proceedings of Usenix Conference. Winter 92.