Método passivo para monitoração de SLA de aplicações sensíveis ao atraso baseado em hash

Emir Toktar[*], Guy Pujolle[*], Edgard Jamhour, Manoel Camillo Penna, Mauro Fonseca

Pontifical Catholic University of Paraná, PUCPR, PPGIA,
Rua Imaculada Conceição 1155, CEP 80215-901, Curitiba, Brasil.

[*]University of Paris VI, LIP6 Lab. 8, rue du Capitaine Scott, 75015, Paris.

emir.toktar@etu.upmc.fr, Guy.Pujolle@lip6.fr,
{jamhour, penna, mauro.fonseca}@ppgia.pucpr.br

Abstract. *This paper presents an approach for monitoring service level agreements (SLA) for delay-sensitive applications. The approach defines a passive measurement mechanism based on the generation of labels to the IP packets, using hash functions. It permits to evaluate end-to-end performance indicators such as delay and packet loss for both, individual or aggregated flows. This proposal has been evaluated on a video streaming scenario.*

Resumo. *Este artigo apresenta uma estratégia para monitoração de acordos de nível de serviço (SLA) de aplicações sensíveis ao atraso. A estratégia propõe um mecanismo de medição passiva baseada na geração de etiquetas para os pacotes IP, utilizando funções de hash. A estratégia permite avaliar indicadores de desempenho como atraso e perdas de pacotes entre dois pontos quaisquer da rede, tanto para fluxos individuais quanto para o tráfego agregado. A proposta foi avaliada num cenário com tráfegos de vídeo.*

1. Introdução

Com a evolução da Internet surgiram várias aplicações, desde transferências de arquivos até comércio eletrônico e aplicações multimídia, aumentando o tráfego da rede e tornando mais complexa sua monitoração. A necessidade de medição é uma conseqüência direta da necessidade de ofertar serviços de comunicação com qualidade, o que ocorre simultaneamente com o aumento significativo das taxas de transmissão. O resultado é o crescimento de estudos e propostas de infra-estruturas de medição para redes IP. As infra-estruturas de medição têm se baseado em duas abordagens principais, a medição ativa que injetam na rede pacotes chamados de sonda, e efetuam as medições com base nestes pacotes; e a medição passiva que se refere ao procedimento de coletar informações sobre o tráfego de rede, sem qualquer intrusão ou modificação.

Por outro lado, para a maioria dos usuários, a Internet é simplesmente um meio de conexão entre aplicações, não estando assim interessados nos detalhes técnicos da oferta de serviço, mas sim na sua qualidade. Disciplinas, como a gestão de qualidade de serviço (SLM – *Service Quality Management*), foram introduzidas para assegurar ao usuário a qualidade desejada. A base destes procedimentos são os acordos de nível de serviço (SLA – *Service Level Agreement*), estabelecidos entre os usuários e os provedores de serviços, e que são expressos em cláusulas contratuais. A base para a

gestão de qualidade de serviço é a medição, já que os SLAs são definidos através de indicadores de qualidade de serviço. Além disso, o processo de medição e cálculo de indicadores deveria expressar a qualidade experimentada pelo cliente. Conceitos como indicadores chaves viabilizam a gestão da qualidade, definindo para todos os serviços, indicadores chaves de desempenho (KPI – *Key Performance Indicator*) e indicadores chaves de qualidade (KQI – *Key Performance Quality*). Os KPIs de um serviço formam o conjunto dos indicadores que afetam a qualidade do mesmo, e que precisam ser avaliados para se determinar tecnicamente sua qualidade. Os KQIs de um serviço devem expressar a qualidade percebida pelo cliente (QoE – *Quolity of Experience*), ou qualidade da experiência, como vem sendo denominada.

O presente estudo propõe um método de medição de KPI de fluxos individuais de tráfego sensível ao atraso, em redes IP, através de um mecanismo de medição passiva, baseado na assinatura dos pacotes através de funções *hash*. O estudo propõe uma arquitetura de medição, e discute como as funções *hash* podem ser usadas para construir um sistema de medição fim-a-fim, que permita a avaliação do atraso, variação do atraso e da perda de pacotes, em um único sentido, ou seja, os KPIs de aplicações como voz e vídeo.

Este artigo está estruturado da seguinte forma: a seção 2 apresenta os trabalhos relacionados, em seguida, na seção 3, discute-se a motivação do estudo, com introdução aos conceitos de indicadores de qualidade e desempenho, e se descreve um experimento preliminar baseado em medição ativa, a partir do qual se verificou suas limitações na avaliação de KPI de atraso e perda de pacotes para fluxos individuais. A seção 4 apresenta nossa proposta de medição e a seção 5 mostra os resultados da estratégia em um cenário com tráfego de vídeo. Finalmente, a conclusão resume os principais aspectos deste estudo e aponta para os trabalhos futuros.

2. Trabalhos Relacionados

Medição em redes IP privadas e na Internet é um problema complexo e aberto, com intensa atividade de pesquisa e desenvolvimento. Além da instrumentação necessária, baseada em soluções de hardware e software, a medição envolve a definição de métricas e procedimentos comuns, que garantiriam a compatibilidade entre dados coletados, possibilitando tornar, especialmente na Internet, os resultados disponíveis para um número abrangente de usuários.

No trabalho, [Paxson et al. 1998] apresentam os objetivos e requisitos do projeto NIMI (*National Internet Measurement Infrastructure*), onde uma plataforma de medição ubíqua para a Internet é proposta. Medições de atrasos de ida-e-volta (*round-trip*) e de perdas de pacote foram avaliadas em seu trabalho, utilizando processos de sonda dedicados (*probes daemons*) em 50 sites através da Internet nos Estados Unidos. Uma questão operacional abordada é o problema de sondas instaladas em computadores não dedicados para fins de medição e a sua sincronização de tempo, usando-se apenas o protocolo NTP (*Network Time Protocol*). Os focos essenciais de sua discussão são, entretanto, as questões arquiteturais e de escalabilidade, enfatizando o controle descentralizado das medições; a autenticação forte e segurança; os mecanismos distribuídos para direitos e controle de acesso; e os mecanismos simplificados de configuração e manutenção da plataforma.

O grupo de trabalho de Métricas de Desempenho para redes IP (*IP Performance Metrics (IPPM) work group*) da IETF tem desenvolvido um conjunto de métricas padrão que podem ser aplicadas para qualidade, desempenho, e confiabilidade dos serviços de entrega de dados da Internet. Os padrões do IPPM são úteis para uniformização das medições que podem ser realizadas na Internet ou em redes IP privadas, sendo muito importantes para a definição indicadores chave de desempenho com significado comum. Por exemplo, as métricas de atraso de pacotes em único sentido (*one-way packet delay*) [Almes et al. 1999a], e perda de pacotes em único sentido (*one way packet loss*) [Almes et al. 1999b], já foram utilizadas em plataformas de medições como Surveyor [Kalidindi et al. 1999] e RIPE NCC [Georgatos et al. 2001]. Estes, são exemplos de unificação de procedimentos, medições de um único sentido (*one-way*) são antagônicas em relação às medições de ida-e-volta (*round-trip*), já que existem muitas instâncias nas quais os caminhos de ida e volta são muito diferentes entre si (em termos de rota), e consequentemente produzem valores para medidas atraso mínimo, variação no atraso ou perda de pacotes.

Duas abordagens principais podem ser consideradas na medição de redes de comunicação: medição ativa quando tráfego é injetado na rede para os objetivos de medição; e medição passiva, onde coletores de dados são dispostos em pontos estratégicos da rede para coleta de tráfego. Os projetos Surveyor e RIPE NCC adotaram a abordagem de medição ativa e os procedimentos e métricas definidas pelo IPPM para obtenção de dados de desempenho unidirecionais em redes IP. A infra-estrutura que ambos desenvolveram para realizar as medições é semelhante, composta de dispositivos de medição, um servidor de análise e um banco de dados. Os dispositivos de medição são equipamentos dedicados, com relógios GPS e sistema operacional modificado para melhorar a precisão no relógio do sistema. O servidor é responsável por controlar e configurar as medições, coletar e catalogar os resultados no banco de dados e finalmente, analisar os dados disponibilizados e acessados através do protocolo http. A maior diferença entre os trabalhos, é que o Surveyor modifica o *driver* da placa de rede no nível do *kernel* para gerar o *timestamp* para os *sockets*, permitindo uma maior precisão da informação.

O grupo de trabalho para Amostragem de Pacotes (*IETF Packet Sampling* (*PSAMP*) *working group*) está trabalhando na definição de padrões para medição passiva na Internet. [Duffield et al. 2001], publicaram um Internet-Draft que propõe um *framework* para medição passiva, com os seguintes requisitos: "(i) ser suficientemente genérica para servir de base para um amplo espectro de tarefas operacionais, (ii) apoiar-se em um conjunto pequeno de primitivas que facilitem seu uso em interfaces de roteadores ou dispositivos de medição dedicados, mesmo para altas velocidades". O *framework* proposto é decomposto em três partes principais, seleção de pacotes para medição, criação e exportação dos relatórios de medição, e conteúdo e formato dos registros de medição.

As técnicas para seleção de pacotes para medição são organizadas no PSAMP como técnicas de amostragem e técnicas de filtragem, apresentadas por [Zseby et al. 2005]. As técnicas de filtragem incluem a amostragem sistemática, onde a escolha do ponto de início e da duração do intervalo de seleção é efetuada através uma função determinística; e a amostragem randômica, onde a escolha se dá segundo um processo aleatório. A filtragem corresponde à seleção de pacotes com base no seu conteúdo,

separando todos os pacotes com certa propriedade dos demais. Duas categorias de filtragem são discutidas, a primeira, baseada na correspondência de campos, ou seja, um pacote é selecionado se um campo específico no mesmo é igual a um valor pré-definido. A segunda é a filtragem baseada em *hash*. Dois tipos de uso são previstos para a seleção baseada em *hash*: uma maneira de realizar a amostragem randômica de modo aproximado usando o conteúdo do pacote para gerar variáveis pseudo-randômicas, e uma maneira de selecionar de forma consistente subconjuntos de pacotes que compartilham uma propriedade comum. Um exemplo do segundo caso é o método de observação de tráfego ou de medição passiva do projeto "*Trajectory Sampling*" [Duffield et al. 2005], que permite reconstruir sua trajetória dos fluxos do tráfego através de um domínio, independentemente do seu ponto de entrada (*ingress*) ou de saída (*egress*), observando-se apenas as trajetórias de amostras de pacotes que atravessam a rede.

3. Motivação

3.1 Medição e SLA

A base para a avaliação da qualidade dos serviços de comunicação é a monitoração de indicadores definidos pelo cliente e pelo fornecedor do serviço em um SLA. A qualidade depende de fatores técnicos que podem ser controlados pelo provedor, mas também depende da natureza da interação do cliente com o serviço, o que tem sido identificado na literatura com qualidade de experiência (QoE – *Quality of Experience*). Uma questão essencial é como mapear as medidas de percepção de QoE em medidas para SLAs.

Um dos conceitos, com ampla aceitação na comunidade científica e industrial, para possibilitar este mapeamento, foi proposto pelo TeleManagement Fórum, que consiste em definir indicadores chave de desempenho e de qualidade (KPI – *Key Performance Indicator* e KQI – *Key Quality Indicator*) [TMForum 2004]. KPIs são os indicadores fundamentais para medir o desempenho de um determinado serviço, enquanto que KQIs são indicadores derivados dos KPIs, que podem representar a QoE. Um KQI é derivado de um número de fontes de informações, incluindo métricas de desempenho do próprio serviço ou métricas de serviços subjacentes. Como um serviço ou aplicação é suportado por diversos de elementos de serviço, vários KPIs distintos podem ser necessário para calcular um KQI particular. Segundo a definição do *TeleManegement Forum*, o mapeamento entre KQI e KPI é dependente de aplicação, e pode ser simples ou complexo, empírico ou formal.

De modo geral um KQI é definido a partir de um conjunto de KPIs e cada KPI ou KQI terá faixas limiares superiores e inferiores de advertência ("*Lower Warning*/*Upper Warning*") e de erro ("*Lower Error*/*Upper Error*"), conforme mostrado na figura 1.

Figura 1. Parâmetros de limiares de KPI/KQI.

Os KPI são então combinados por alguma função empírica ou teórica para calcular o valor do KQI relacionado. A figura 2 ilustra este relacionamento de vários parâmetros (*P*) através de através de uma função $f(P_1,P_2,...,P_n)$ e a sua relação com parâmetros KQI representados por uma função $F(S_1,S_n)$. Por exemplo, um conjunto de valores de KPIs sinalizando uma situação de advertência podem significar que o serviço encontra-se em tal estado de degradação que significa a não disponibilidade do mesmo, logo, deveriam ser considerados como um erro indicando uma violação de KQI.

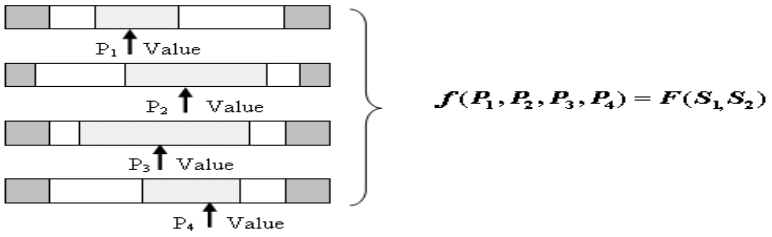


Figura 2. Combinação de KPI para determinar um KQI.

A definição de funções de vinculação entre KQIs e KPIs é um importante tema de investigação atual. Trabalhos consideráveis neste sentido têm sido desenvolvidos por entidades como ITU (*E-Model*) [ITU-T 1999 e 2005] e outras entidades comerciais como NetForecast (*www.netforecast.com*). Neste trabalho a qualidade do serviço "*streaming video*" é avaliada através dos KPIs "perda de pacotes" e "atraso", que são obtidos a partir dos registros coletados através do método de medição proposto.

3.2 Avaliação de medição ativa para fluxos individuais de vídeo

As medições em redes de comunicação são fundadas em duas abordagens distintas. A medição ativa ocorre quando o procedimento de medição injeta na rede pacotes específicos, denominados pacotes de sonda, para realizar as medidas. Por outro lado, a medição passiva se refere ao processo de monitorar o tráfego de rede, gravando-o para análise posterior, sem que haja injeção de novo tráfego ou modificação do mesmo.

No que concerne este estudo, alguns indícios pareciam indicar que medição ativa seria mais adequada para medir o tráfego agregado, sendo menos indicada em situações onde é necessário obter o desempenho de fluxos individuais. A razão para esta conjectura é que os pacotes injetados na rede tendem a distribuir-se de modo não controlável através do tráfego agregado, não possibilitando uma leitura fina do desempenho dos fluxos individuais a partir da amostra de tráfego representado pela sonda.

Para permitir um padrão de comparação com a estratégia de medição passiva, descrita nas seções 4 e 5, decidiu-se realizar um experimento preliminar baseado em medição ativa. O experimento foi realizado em ambiente controlado, através da injeção de sondas juntamente com tráfego de vídeo, para se avaliar a qualidade de fluxos individuais. As sondas contêm dados como hora local do gerador de sonda e número de seqüência, que foram comparadas com o relógio do consumidor de sonda. Os dados carregados pelas sondas foram então usados para calcular os KPIs atraso e perda de pacotes. Neste experimento os geradores de sonda foram instalados em três computadores cliente que requisitavam o serviço de vídeo e o consumidor de sonda foi instalado no servidor de vídeo, conforme ilustrado na figura 3.

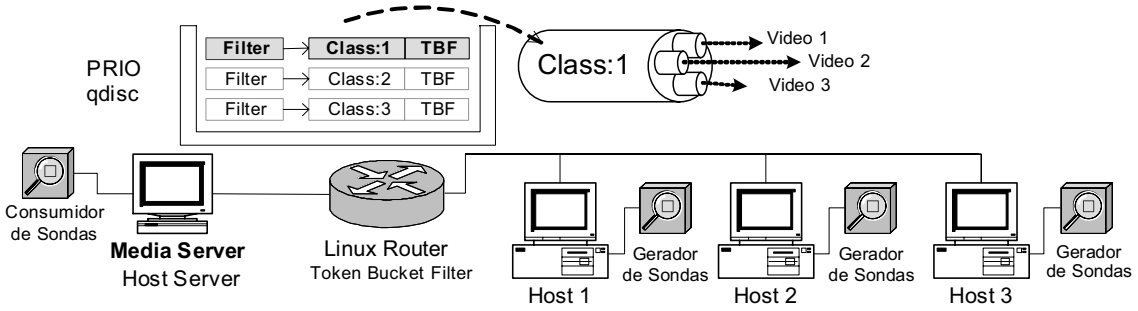


Figura 3. Cenário do teste de sondas (*probes*)

Cada vídeo tinha uma duração máxima de 2.36 minutos, com taxa (*bit rate*) de 1.32 Mbps, codificador de áudio *Windows Media Audio* 9.1, 128 kbps, 48 kHz, 2 canais 24 *bit* (A/V) 1-*pass* CBR e codificador de vídeo o *Windows Media* 9. Os geradores de sonda foram ajustados para enviar 48 *bytes* de dados como data, hora local e a seqüência dos pacotes, usando o protocolo UDP, com um intervalo de 50ms entre as amostras. Os geradores e consumidor de sonda foram implementados com a plataforma .Net 2.0. Os clientes foram computadores DELL 1.5GHz com 256MB de memória e o computador servidor foi um *Centrino Duo Core* T2300 1.66GHz, 512MB RAM.

A sincronização de tempo foi obtida usando o protocolo NTP, com uma diferença máxima observada de 10 ms, o que pode gerar um erro de até 2,5%, considerando que o *upper error threshold* para o KPI atraso seja 400ms. Para contornar o problema de sincronização entre os diversos computadores envolvidos, adotou-se o deslizamento de valores de relógios para ajuste de valores no cálculo dos resultados. A cada intervalo de 400 ms são enviadas oito sondas. Todavia, não se pode garantir que as oito sondas chegarão eqüidistantes no destino final. Para forçar o efeito de atraso e perda de pacotes foi introduzido um filtro de balde de fichas no roteador.

Deve-se observar que o tamanho total dos quadros de sonda foi de 90 bytes (14 bytes de enlaces + 20 bytes de cabeçalho IP + 8 bytes UDP + 48 bytes com dados da sonda). Esse tamanho é bastante inferior a um quadro de vídeo RTP (*Real Time Protocol*), que atingiu o tamanho máximo de 1393 bytes no experimento. Dessa forma, mesmo quando o buffer do roteador estava próximo do seu limite, os quadros RTP eram descartados, mas não os tráfegos de sonda. Dessa forma, apesar das sondas estarem distribuídas a uma taxa de 60 sondas/s, frequentemente, as sondas se acumularam na fila do roteador, perdendo totalmente sua conotação de tempo. A conclusão do experimento, confirmou que o uso de sondas não produz o resultado esperado, e que uma abordagem de medição passiva deveria ser aplicada para monitorar o desempenho de fluxos individuais de vídeo.

4. Estratégia Proposta

Conforme discutido na seção anterior, a estratégia de medição baseada em sondas é pouco adequada à aferição de SLAs de fluxos individuais sensíveis ao atraso. Como alternativa este estudo desenvolveu uma estratégia não intrusiva, no qual as medições de atraso e de perda de pacotes são feitas analisando os próprios pacotes dos fluxos individuais. Os principais elementos utilizados nessa estratégia podem ser observados na figura 4. Como o objetivo da estratégia é a aferição do SLA fim-a-fim, é necessário incluir componentes adicionais em cada nó de ingresso e egresso da rede. A inclusão desses componentes pode ser feita de várias formas, mas é necessário que os módulos

tenham acesso a todos os pacotes que atravessam a interface dos nós. Uma estratégia prática consiste em incluir um equipamento com a interface de rede em modo promíscuo, no mesmo segmento de rede que o nó monitorado. Os pacotes capturados em modo promíscuo são filtrados de acordo com os critérios do fluxo a ser monitorado (por exemplo, filtragem baseada em endereços IP e portas TCP/UDP de origem e destino). Para cada pacote monitorado, uma etiqueta (*label*) é calculada computando-se um algoritmo de hash sobre alguns campos selecionados do cabeçalho e parte do campo de dados. Essa etiqueta, juntamente com as informações de identificação do fluxo e a hora local são armazenados sistematicamente em um arquivo de *log*.

As entradas dos *logs* dos nós de ingresso e egresso são transmitidas sob demanda para um coletor central, responsável por calcular os elementos de KPI e KQI, utilizados na aferição do SLA. O coletor processa os *logs* e gera dados com informações dos KPIs, qualificando as amostras segundo os limiares discutidos na seção 3, para posterior validação do SLA. Os *logs* são descartados no coletor após seu processamento e qualificação. Igualmente, na medida em que os dados são transmitidos para o coletor eles podem ser descartados dos *logs* de ingresso e egresso, evitando a sobrecarga dos mesmos. Esse procedimento é gerenciado através de um protocolo de configuração e transmissão de métricas que padroniza a comunicação entre os elementos de medição. O protocolo é responsável por configurar os filtros e controlar o envio de informações dos nós de ingresso/egresso para o coletor. Observe que, dependendo da forma como o filtro de pacotes é definido, é possível utilizar a mesma técnica para aferição de fluxos individuais ou agregados.

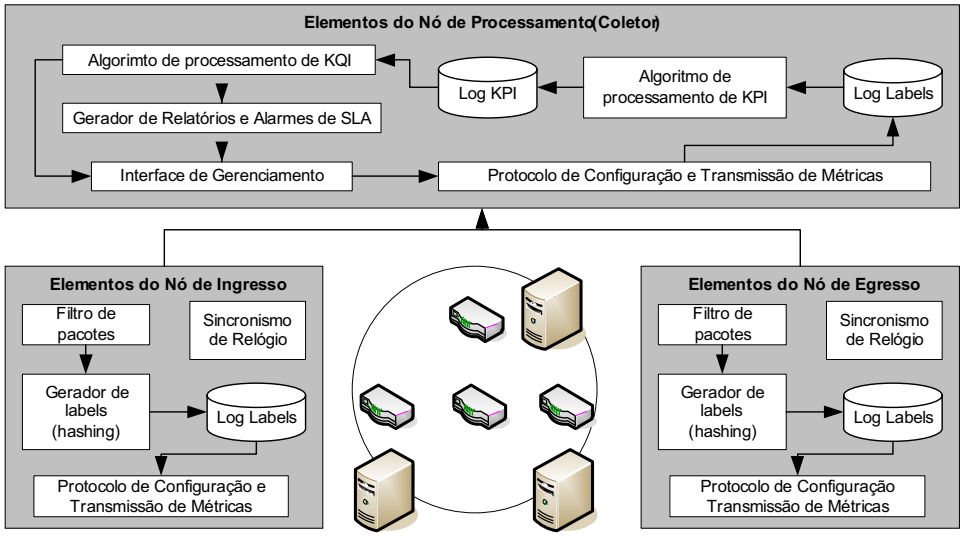


Figura 4. Visão geral do sistema de medição por etiquetas.

4.1. Utilização de Etiquetas

A estratégia de análise não intrusiva dos pacotes necessita que cada pacote do nó de ingresso seja comparado com o seu correspondente no nó de egresso. Desta forma, é necessário utilizar alguma técnica que permita identificar que se trata do mesmo pacote. A utilização de vários campos para identificação de um pacote, além de ser dispendiosa em termos de armazenamento e transmissão, pode ser muito custosa em termos de processamento, pelas razões já bem discutidas na teoria de bancos de dados relacionais. Por essa razão, nossa estratégia atribui etiquetas aos pacotes utilizando algoritmo *hash*.

A estratégia de atribuir etiquetas aos pacotes, usando um algoritmo de *hash*, está alinhada com as recomendações do PSAMP (ainda na forma de *draft*). De certa forma, essa estratégia também foi utilizada em projetos pioneiros, como o descrito em [AT&T 2006]. O projeto da AT&T, todavia, utiliza os algoritmos de *hash* para projetar os filtros de pacotes, uma vez que a estratégia é baseada em amostragem.

A questão de utilizar uma etiqueta que represente um identificador único para o pacote tem sido bastante discutida, com trabalhos que argumentam que apenas dados do cabeçalho IP são suficientes para gerar a informação. A especificação do PSAMP recomenda usar no mínimo quatro bytes do campo de dados IP. O projeto de amostragem de pacotes da AT&T utilizou 20 bytes do campo de dados.

A RFC 1263 discute o uso do campo de identificação (bits 32-47) do cabeçalho IP, mas indica que esse campo pode ser facilmente exaurido. Este campo de 16 bits permite identificar até $((2^{16} -1) = 65.535)$ pacotes. A RFC 4459 e [Mathis et al. 2007] alertam que se um dispositivo encaminhar 1.500 bytes com tempo de vida no máximo de 30 segundos, a taxa deve ser limitada a 26Mb/s para não exceder o limite de 65.535 identificadores de pacote. Se for considerada uma interface de 1Gb/s com pacotes de 1500 bytes, haveria necessidade de repetir o identificador em menos de 1 segundo.

4.2 Seleção dos Campos para Cálculo da Etiqueta

A operação de *hash* efetuada nos nós de ingresso e egresso deve ser a mais eficiente possível, pois os pacotes devem ser processados continuamente na medida em que vão sendo transmitidos. Quanto maior a velocidade da rede, menor o tempo disponível para essa operação. Dessa forma, é necessário limitar ao máximo os campos dos pacotes utilizados para cálculo da etiqueta.

De acordo com sua entropia e mutabilidade ao longo do caminho, os campos do pacote IP podem ser classificados em três categorias [Duffield et al. 2005]: (i) campos variáveis: campos do cabeçalho IP que sofrem alterações ao longo do caminho, (ii) campos com baixa entropia: são campos fixos que não sofrem variação de conteúdo durante o trajeto, mas tem pouca ou nenhuma diferenciação entre pacotes observados. (iii) campos com alta entropia: são campos fixos que não sofrem variação ao longo do caminho e tem grande diferenciação entre os pacotes observados.

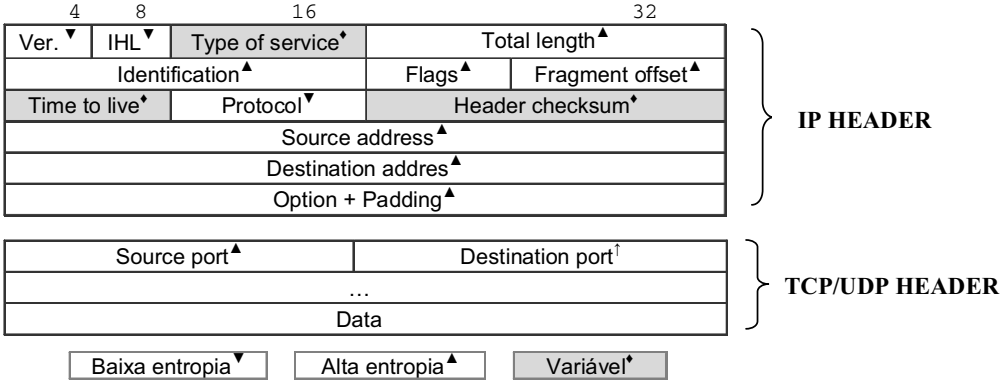


Figura 5. Variação de conteúdo do cabeçalho IP

Os campos do cabeçalho IP e sua classificação em relação à entropia e mutabilidade são mostrados na figura 5. Os campos variáveis são: *Time to Live* - TTL (bits 64-71), o qual é diminuído a cada salto. *Type of Service* - ToS (bits 8-15), já que

este pode ser reclassificado em um roteador. Idem para sua redefinição como byte *DS*. *Header Cheksum* (bits 80-95), pois este é recalculado na alteração do conteúdo do cabeçalho.

Os seguintes campos são de baixa entropia: *Version* (bits 0 – 3): diferencia apenas IPv4 de IPv6. *Internet Header Length - IHL* (bits 4-7): alterado apenas na presença de opções IP. *Protocol* (bits 72-79): na maior parte dos casos indica apenas TCP ou UDP.

Os seguintes campos são fixos e com alta entropia: *Source* e *Destination IP Address* (bits 96-159): apresentam alta entropia entre fluxos, mas são invariáveis para pacotes do mesmo fluxo. *Identification* (bits 32-47): identificador único para fragmentos do mesmo pacote. Este campo apresenta alta entropia mesmo para pacotes de um mesmo fluxo. Campos de outros protocolos, como TCP e UDP, quando presentes, também podem ser utilizados. Os campos *Source* e *Destination Port* do TCP/UDP apresentam alta entropia entre fluxos, mas são invariáveis para pacotes do mesmo fluxo.

Apesar de, teoricamente, apresentarem alta entropia, os seguintes campos são pouco úteis para o cálculo da etiqueta: *Total length* (bits 16-31), *Flags* (bits 48-51) e *Fragment offset* (bits 52-63). A razão para isso é a grande concentração de pacotes de tamanho semelhante, como aqueles iguais ao MTU da rede, ou os pacotes de controle TCP. Igualmente, devido à ação do protocolo TCP, a ocorrência de fragmentação é bastante rara, fazendo com que a contribuição desses campos seja pequena quando comparada ao aumento de complexidade da operação de *hash*.

Devemos salientar que para efeito de cálculo do atraso e de perda de pacotes, basta garantir que a etiqueta seja única em um intervalo de tempo que é função do atraso oferecido pela rede e pelo atraso máximo imposto pelo SLA. Mesmo assim, somente o uso dos campos de alta entropia pode não ser suficiente para garantir que o identificador seja único nesse intervalo, principalmente em redes de alta velocidade. Essa é a justificativa para utilização também de alguns bytes do campo de dados, diminuindo a possibilidade de repetição da etiqueta. Ainda, a quantidade de dados necessários para formar um identificador realmente único para o pacote seria muito grande, e de pouca valia para efeito da medição de perda de pacotes e atraso, pois pacotes acima de um atraso máximo são considerados perdidos.

Os fatos expostos acima definiram a utilização dos campos de alta entropia (endereço IP de origem e destino, identificador e portas de origem e destino, quando disponíveis), conjuntamente com oito bytes do campo de dados para efeito do cálculo da etiqueta (quando existe o protocolo de transporte, os oito bytes são coletados após o cabeçalho de transporte). Apesar do PSAMP recomendar um mínimo de quatro bytes, optou-se por utilizar oito bytes para compensar a não inclusão do campo de deslocamento de fragmento no cálculo do *hash*. Segundo definição descrita na RFC 791, o menor tamanho de um pacote fragmentado é o cabeçalho IP seguido de oito bytes de dados. O resultado da função *hash*, através da computação destes valores, tem probabilidade de colisão extremamente baixa.

4.3 Algoritmo de Hash

A técnica de *hashing* é considerada um efetivo método para organizar e acessar dados, e muito utilizada em banco de dados, sistemas operacionais, compiladores e outras aplicações. O algoritmo *hash* é uma função matemática que recebe uma entrada de dados de tamanho variado e resulta em um valor de tamanho fixo. Estas funções ainda podem ser classificadas por uma variedade de características, já discutidas por vários autores. Não é o intento do trabalho discutir modelos matemáticos e aplicações das funções *hash*.

Para a escolha da função *hash* neste trabalho foram considerados o impacto computacional, que seria gerado no momento da captura e assinatura dos pacotes, e a probabilidade de gerar colisões, que ocorrem quando dois vetores de entrada produzem o mesmo vetor de saída. As funções *hash* consideradas foram o *MD5*, *Tiger128* [Ross e Biham 1996], *Jenkins32* [Jenkins 1997], *S-Box32* e uma variação desta última com 64 bits. A função *Jenkins* é citada no esboço do PSAMP, descrito por [Zseby et al. 2005]. A avaliação das funções *hash*, incluindo *Jenkins32*, SHA-1 e outras, mais o algoritmo *S-Box32* utilizado neste trabalho é apresentada por [Mulvey 2006]. Utilizando a linguagem C#, foi gerado um vetor de bytes tipo inteiro variando "bit a bit" o valor na faixa de 0 até 2^{21} como entrada da função *hash*. O tempo de processamento e o número de colisões foram observados e apresentados na figura 6. Com o resultado, para minimizar a possibilidade de colisões, adotou-se um *hash* de 128 *bit*, o *Tiger*, que processou o vetor na metade do tempo do *MD5*. Segundo avaliações de [Biham 2006], este algoritmo é capaz de processar até 132Mbps contra 37Mbps do *MD5*.

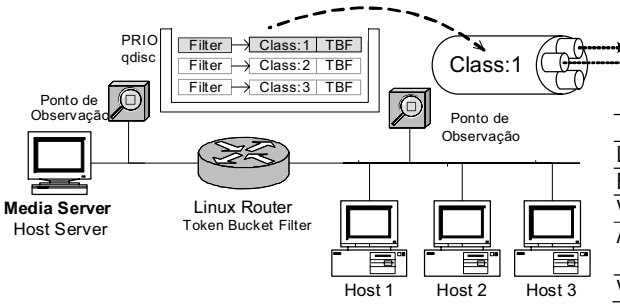
```
00000000000000000000 // (hex) 00 0000 (dec) 0.000.000
00000000000000000001 // (hex) 00 0001 (dec) 0.000.001
......
01111111111111111111 // (hex) 1F FFFF (dec) 2.097.151
10000000000000000000 // (hex) 20 0000 (dec) 2.097.152

MD5       00:01:01.1875000 sec  [no collisions] (128 bit)
Tiger     00:00:31.1250000 sec  [no collisions] (128 bit)
SBox64    00:00:06.9218750 sec  [no collisions]  (64 bit)
SBox32    00:00:05.0156250 sec  [494 collisions] (32 bit)
```

Figura 6. Resultado do processamento das funções *hash*

5. Estudo de Caso

A fim de avaliar a estratégia proposta para medição de fluxos individuas sensíveis ao atraso foi montado um experimento ilustrado pela figura 7. O experimento consistiu em transmitir fluxos de vídeo entre um servidor e três receptores. O experimento foi executado em uma rede LAN controlada utilizando como roteador um computador com o sistema operacional *Linux*, responsável por transportar o tráfego agregado.

A fim de simular um gargalo na rede, o roteador foi programado com uma política de escalonamento do tipo balde de fichas (*token bucket* - TBF). Essa estratégia permitiu provocar atraso e perda de pacotes sobre os fluxos avaliados no experimento. O *script* utilizado para configuração do TBF está ilustrado na figura 8. Os parâmetros que definem o TBF são: "*rate*" especifica a vazão média, "*burst*" determina o tamanho do balde em bytes, "*peakrate*" determina a velocidade máxima para esvaziar o balde. O parâmetro "*latency*" especifica o tempo máximo que um pacote pode aguardar na fila antes de ser processado, e influencia diretamente a taxa de descarte.



	Movie 1	Movie 2	Movie 3
Length	02:22	02:17	02:20
Bit rate	1,32 Mbps	1,32 Mbps	1,32 Mbps
Vídeo size	320 x 240	160 x 128	320 x 240
Áudio codec	Windows Media Audio 9.1 Prof. 128kbps, 48kHz, 2 channel 24 bit (A/V) 1-pass CBR		
Vídeo codec	Windows Media Video 9		

Figura 7. Cenário do teste de fluxo de vídeo.

No experimento, foram adotados os seguintes valores: *burst* de 5Kbytes, *latency* de 70ms e três configurações de taxas "*rate*": 3.8Mbps, 4.2Mbps e 4.8Mbps. Esses parâmetros foram escolhidos para forçar a ocorrência do atraso e perda de pacotes nos fluxos de vídeo avaliados no experimento.

```
tc qdisc del root dev eth0
tc qdisc add dev eth0 root handle 1: prio
tc filter add dev eth0 parent 1:0 prio 1 protocol ip u32 match ip tos 0x00 0xff flowid 1:1
tc qdisc add dev eth0 parent 1:1 handle 10: tbf rate 3.8mbit burst 5kb latency 70ms
```

Figura 8. Comandos usados para configuração do qdisc.

Para execução do experimento, os clientes foram representados por três computadores com sistema operacional Windows XP, equipados com Windows *media player* 10. O tempo de *buffer* foi reduzido para um segundo, minimizando o efeito de *playback* e foi utilizado o modo de vídeo sob demanda. Os clientes são computadores INTEL DELL 1.5GHz com 256MB de memória. No lado do servidor, foi utilizado o Windows Media Server com três serviços de vídeos distintos. Este, um computador *Centrino Duo Core* T2300 1.66GHz com 512MB RAM. A opção "*fast cache*" foi desabilitada para forçar o servidor de vídeo a usar o protocolo RTP (*Real Time Protocol*) sobre UDP, chamado de RTSPU. Sem a memória *cache* no servidor, cada cliente que requisitou conteúdo para o servidor de multimídia foi atendido por uma nova conexão no servidor. A configuração da rede foi *Fast-Ethernet* (100 Mbps). Cada cliente fez uma requisição para um vídeo de duração semelhante, mas características diferentes, ilustrado na figura 7.

Conforme indicado na figura 7, foram utilizados dois computadores operando em modo promíscuo, responsáveis pela marcação dos pacotes no ingresso e egresso da rede. Os computadores são do mesmo tipo utilizado para as estações cliente solicitarem serviço de vídeo. Para a sincronização de tempo nos testes, utilizou-se um servidor de NTP instalado no computador servidor inicialmente sem o roteador. Durante os testes, observou-se uma variação máxima de 10ms entre os ajustes de relógios. Para compensar esse efeito, foi adotado o deslizamento de valores de relógios, conforme discutido na estratégia de medição ativa, na seção 3.2.

O sistema de monitoramento foi desenvolvido utilizando-se a plataforma .Net 2.0, a linguagem C# e a biblioteca [WinPcaP 2006]. A biblioteca *WinPcap* é um código aberto para captura de pacotes e análise de rede para plataforma Windows 32 bits. Para coletar apenas o tráfego de vídeo dos fluxos, adotou-se o uso de filtros na captura de pacotes passados como parâmetro para biblioteca *WinPcap*, o que possibilita uma coleta mais objetiva sobre que dados se deseja observar.

O algoritmo de processamento de KPI é apresentado na figura 9. Os parâmetros dos vetores deste, recebem valores dos arquivos de *logs* dos pontos de observação. Os valores hash dos vetores encontrados armazenam um valor de KPI e o endereço IP destino do pacote do *log,* para análise posterior do algoritmo KQI. A pesquisa dos vetores é limitada no avanço e no retrocesso otimizando as pesquisas. Um avanço muito longo no índice destes vetores, indica um pacote não encontrado ou com atraso muito alto. O retrocesso do índice destes vetores, indica pacotes fora de ordem.

```
int FORWARD_SEEK = 1600; // Upper Limit search: packet too delayed!!
int OFFSET = 512; // Search go back: packet out of order.
set KPI value; // {-2,-1,0,1,2} Values to thresholds;
LoadArray(LogConsumidor,Log'sGenerator); // Vectors: szHash[], szIPSrc[]
for (int i = 0; i < szHash[0].Count; i++) // i → Consumidor
  if ((i mod 2048) is true)) LoadArray(LogConsumidor); // Search has arrived at half of the log
  MATCH_Flag = false;
  for (int j = 1; j < num_Logs; j++) // j → Gerador [1..n]
     // Search go back, it means that have packets out of order.
     index[j]=((off_set[j]-OFFSET)>0)?(off_set[j]-OFFSET):0;
     // Load another Log. Search arrived at half of the log.
     if ((index[j] mod 2048 is true)) LoadArray(LogGerador,Num_Log); // More than one Generator log!
     int _forwardseek = 0;
     // Seek forward until the limit of _FORWARD_SEEK variable
     while (index[j] < szHash[j].Count AND index[j] < szHash[0].Count AND _forwardseek <= _FORWARD_SEEK)
        if (szHash[i] == szHash[j])
           MATCH_Flag = true; int DiffTime = Calc. diffence time between packtes;
           Compare SLA value w/ DiffTime and set KPI value;
           Store KPI value and IPDest Address; break;
        endif;
        _forwardseek++; // next index of Generator vector
    end_while; // KPI value, Thresholds: {-2,-1,0,1,2}
    if (!MATCH_Flag) Store KPI value and IPDest Address;
  end_for;
```

Figura 9. Algoritmo de processamento de KPI

O teste foi executado medindo os fluxos no servidor (ingresso) e clientes (egresso). O *hash* dos pacotes foi calculado em tempo real, sendo inicialmente salvo em memória e, concorrentemente, gravado no arquivo de *log* a cada quatro mil pacotes (10 segundos de vídeo). Os registros dos *logs* foram usados para calcular o KPI de atraso (*delay*), através da comparação dos valores de tempo e o KPI de perda de pacotes (*packet loss*), observando-se a existência ou não do mesmo valor *hash* nos *logs* comparados. Com esta metodologia, é possível examinar os resultados por fluxo individual ou agregado. Para observar o comportamento dos fluxos do serviço de vídeo de um SLA, o indicador KPI para atraso foi ajustado com valor limite de alerta (*Upper Warning Threshold*) de **50ms** e de erro (*Upper Error Threshold*) de **90ms**. O KPI para atraso de pacotes, foi estipulado com limite de erro (*Upper Error Threshold*) de 10%.

Foram analisados os fluxos individuais para cada computador e para o fluxo agregado, de vídeo. Os resultados do KPI de atraso (delay) são apresentados na tabelas 1, 2 e 3. O percentual define a quantidade de pacotes em conformidade com os limiares definidos no KPI atraso, sendo: ***Normal* < 49ms**; ***Warning* > 50ms** e ***Error* > 90ms**. Cada tabela apresenta o resultado para uma configuração diferente do balde de fichas no roteador.

Tabela 1 – Resultado da simulação de vídeo, taxa de 4800Kbps no roteador, KPI atraso

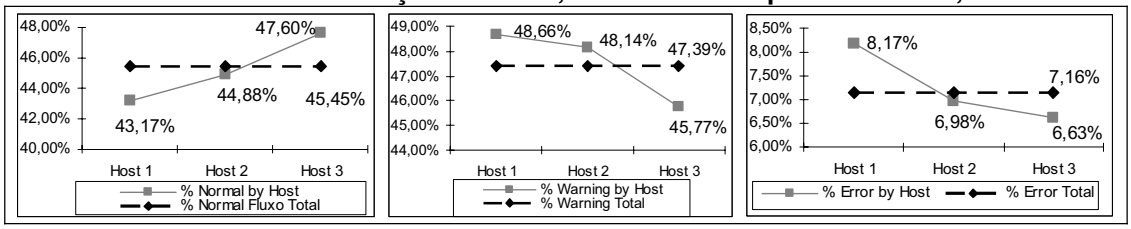


Tabela 2 – Resultado da simulação de vídeo, taxa de 4200Kbps no roteador, KPI atraso

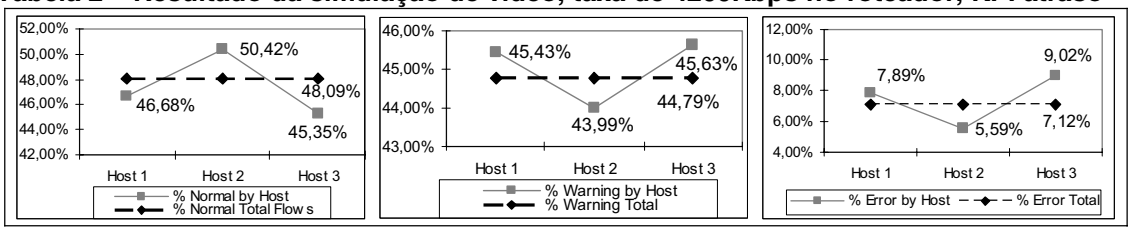
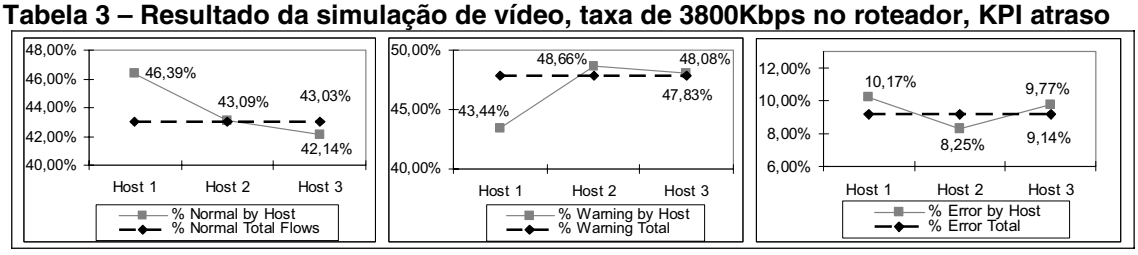


Tabela 3 – Resultado da simulação de vídeo, taxa de 3800Kbps no roteador, KPI atraso



Deve-se observar que o cenário avaliado incluiu apenas três fluxos e um único salto. Mesmo assim, observa-se que existe variação nos valores de KPI dos fluxos individuais em relação ao agregado, tanto para o KPI de atraso quanto para perda de pacotes. Cenários mais complexos certamente levarão a uma variância mais significativa do comportamento dos fluxos.

6. Conclusão

Este artigo observou o uso de medição ativa e passiva para avaliação de SLA. A monitoração de redes usando uma medição ativa com sondas mostrou-se ineficiente para a obtenção de indicadores de KPI para tráfego sensível ao atraso, como vídeo. Como alternativa, esse artigo analisou a estratégia passiva, utilizando etiquetas calculadas com algoritmo de *hash*. Os resultados mostraram que essa abordagem é satisfatória tanto para avaliar fluxos individuais quanto o tráfego agregado.

Foi observada que na avaliação de fluxo agregado, este pode não refletir corretamente o comportamento de fluxos individuais, questão que deve ser levada em consideração na escolha da ferramenta de monitoração de SLA. O software usado no trabalho precisa ser otimizado para atingir um desempenho melhor, também limitado pelo hardware utilizado. A utilização da biblioteca *Winpcap* possibilitou a captura dos pacotes ao nível de *kernel*, mas a seleção dos dados a serem codificados utilizando *hash* foi executada ao nível de aplicação. Apesar disso, a estratégia se mostrou viável para operação em enlaces de até 100 Mbps. Para velocidades superiores, da ordem de Gigabit, seria necessário utilizar um hardware dedicado e levar as demais operações do sistema para o nível do *kernel*.

Algumas áreas precisam ser aprimoradas nesse trabalho, muitas delas já apontadas nos trabalhos do PSAMP. Entre elas, a procura de um protocolo padronizado para configuração remota e coleta dos nós de ingresso e egresso. Estudos adicionais devem ser feitos para avaliar a velocidade de coleta a fim de estimar com que freqüência o *log* dos nós pode ser efetivamente liberado. Igualmente, um estudo mais aprofundado é necessário em relação ao algoritmo do coletor, uma vez que a quantidade de comparações a ser executada em uma rede com vários fluxos monitorados pode ser extensa. Finalmente, estudos podem ser feitos para utilização de técnicas amostrais para cálculo dos KPIs.

6. Referências

Paxson, V., Mahdavi, J., Adams, A., Mathis, M. (1998) "An Architecture for Large-Scale Internet Measurement," IEEE Communications, Vol. 36, No. 8, Aug.

Almes, G., Kalidindi, S., Zekauskas, M. (1999a) "A One-way Delay Metric for IPPM", IETF Network Working Group, RFC2679.

Almes, G., Kalidindi, S., Zekauskas, M. (1999b) "A One-way Packet Loss Metric for IPPM", IETF Network Working Group, RFC2680, September.

Kalidindi, S., Zekauskas, M. J. (1999) "Surveyor: An Infrastructure for Internet Performance Measurements", INET Internet Network, California, USA.

Georgatos, F. et al. (2001) "Providing active measurements as a regular service for ISP'S". RIPE NCC. In Passive & ActiveMeasurement (PAM), Amsterdam, April.

Duffield, N., Greenberg, A., Grossglauer, M., Rexford, J. (2005) "A Framework for Packet Selection and Reporting", IETF PSAMP Working Group, Internet-Draft, Jul.

Zseby, T., Molina, M., Duffield, N., Niccolini, S., Raspall, F. (2005) "Sampling and Filtering Techniques for IP Packet Selection", PSAMP, Internet-Draft, Jul.

Duffield, N., Grossglauser, M., (2000) "Trajectory sampling for direct traffic observation", Proceedings of the conference on SIGCOMM '00. Aug.

SLA Management Handbook TeleManagement Forum. (2004) ''SLA Management Handbook, Volume 4, Enterprise Perspective'', G045. The Open Group.

ITU-T Recommendation G.108. (1999) "Application of the E-model: A planning guide". Sep/1999.

ITU-T Recommendation G.107. (2005). "The E-Model, a computational model for use in transmission planning". Mar/2005.

Trajectory Sampling. AT&T Inc., R&D. (2006) URL: http://www.research.att.com/~trajsamp/ . Acessado em 2006.

Mathis, M. e Heffner, J. (2007) "Fragmentation Considered Very Harmful", Work in Progress, Jan. 2007.

WinPcaP Project. (2006) URL: http://www.winpcap.org.

Ross, A., Biham, E. (1996) "Tiger: A Fast New Hash Function", IWFSE: International Workshop on Fast Software Encryption, LNCS, Cambridge.

Jenkins, B. (1997) "Algorithm Alley", Dr. Dobb's Journal, Sep. URL http://burtleburtle.net/bob/hash/doobs.html.

Mulvey, Bret. (2006) "Hash Functions page". URL: http://bretm.home.comcast.net/

Biham, E. (2006) "Test Results of Tiger". URL: http://www.cs.technion.ac.il/~biham/Reports/Tiger. Acessado em 2006.