Comparando Aspectos de Desempenho do Protocolo SNMP com Diferentes Estratégias de Gateways Web Services

Ricardo Lemos Vianna, Tiago Fioreze, Lisandro Zambenedetti Granville, Maria Janilce Bosquiroli Almeida, Liane Margarida Rockenbach Tarouco

Universidade Federal do Rio Grande do Sul - Instituto de Informática
Av. Bento Gonçalves, 9500 - Bloco IV - Porto Alegre, RS – Brasil
{rvianna, tfioreze, granville, janilce, liane}@inf.ufrgs.br

Abstract. *Web Services gateways can be used to include SNMP devices into a Web Services-based management architecture. This paper presents an evaluation of bandwidth consumption and response time of two different Web Services to SNMP gateway strategies. The evaluation intends to verify the feasibility of using Web Services closer to the network devices management interface. In addition, it has been verified the impact associated with the data compression and encryption of the Web Services communication traffic. The collected results have been then compared with the associated SNMP traffic.*

Resumo. *Gateways Web Services podem ser usados para incluir dispositivos SNMP em uma arquitetura de gerenciamento baseada em Web Services. Neste artigo é apresenta a avaliação de consumo de banda e tempo de resposta de duas estratégias para gateways Web Services para SNMP. A avaliação foi realizada para se verificar a possibilidade de usar Web Services próximos à interface de gerenciamento dos dispositivos de rede. Também foi verificado o impacto da compressão e criptografia dos dados necessários à comunicação com os gateways e os resultados obtidos foram comparados com o tráfego SNMP associado.*

1. Introdução

Nas últimas décadas, grandes esforços foram realizados na tentativa de contornar problemas relacionados ao gerenciamento de redes. Um desses problemas é a grande diversidade de dispositivos de rede existentes atualmente. Como cada um desses dispositivos contém particularidades de manipulação, a tarefa de configuração manual (i.e. através da interação direta do operador de rede com o dispositivo gerenciado) torna-se difícil em alguns casos. Devido a tal diversidade e aos problemas provenientes disso, protocolos de gerenciamento foram desenvolvidos com o intuito de tornar a atividade de gerenciamento de redes mais factível e menos propensa a erros de configuração. A definição do SNMP (*Simple Network Management Protocol*) [Case et al. 1990] foi o passo chave nessa direção. Então, usuários e fornecedores de dispositivos de rede começaram a construir suas soluções de monitoração baseadas em SNMP, as quais impulsionaram o conhecimento em relação às redes gerenciadas. Mais adiante, as arquiteturas RMON [Waldbusser 2000] e RMON2 [Waldbusser 1997] aumentaram o poder dos processos de monitoração baseados em SNMP, melhorando o nível de detalhes sobre o uso da rede e os serviços rodando sobre a mesma. Entretanto, novos problemas surgiram e o protocolo SNMP não tem se mostrado unânime como solução

para os mesmos. Por exemplo, não existe uma arquitetura largamente aceita para delegação de *scripts* de gerenciamento, embora o IETF tenha proposto a MIB Script [Levi and Schoenwaelder 1999]. Questões de segurança ainda estão abertas, apesar do SNMPv3 [Case et al. 2002] ter sido criado para abordar tais questões. Outro fator que conta contra o SNMP é o fato de muitos administradores de redes não confiarem no mesmo para gerenciar a configuração dos dispositivos de rede, optando muitas vezes no desenvolvimento de ferramentas próprias. Essa atitude vai contra o trabalho que vem sendo desenvolvido pelo IETF, no contexto do grupo de trabalho SNMPCONF, em relação à utilização do SNMP para configuração [MacFaden et al. 2003].

Uma nova tecnologia, denominada Web Services (WS) [Curbera et al. 2002], tem se mostrado bastante promissora, despertando o interesse da indústria e dos pesquisadores na área de gerenciamento de redes. Embora os WS tenham sido desenvolvidos originalmente para suportar processos de *e-commerce*, eles podem ser usados como uma ferramenta de integração em diversos sistemas, tais como redes de suporte a *grids* [Goth 2002], inteligência artificial [Preece and Decker 2002] e, recentemente, gerenciamento de redes. Nesse último, WS parecem ter um potencial para resolver alguns dos problemas de gerenciamento de rede apresentados anteriormente. Devido à utilização de protocolos Web, o tráfego WS cruza *firewalls* na Internet com mais facilidade que o SNMP. A gama de ferramentas disponíveis para lidar com WS e o suporte nativo a WS oferecidos por plataformas de desenvolvimento de softwares são maiores que as facilidades disponíveis no universo SNMP. Além disso, WS podem ser implementados através do protocolo SOAP (*Simple Object Access Protocol*) [Mitra 2003], o qual pode operar sobre serviços de transporte seguros, implementados através de protocolos como HTTPS, SMTP encriptado e FTP seguro. Além disso, enquanto o SNMP é usado quase somente na comunicação das estações de gerenciamento com os dispositivos de rede (apesar dos esforços do SNMPv2), WS poderiam ser usados para melhorar a comunicação entre diferentes plataformas de gerenciamento ou mesmo para construir complexos sistemas baseados em serviços de gerenciamento mais simples. Provavelmente, um dos ganhos mais interessantes obtidos com WS é a integração mais fácil do gerenciamento de redes com outros processos críticos, como *e-business*.

Embora esse novo cenário de gerenciamento possa se apresentar fascinante à primeira vista, a introdução de WS no gerenciamento de redes deve ser cautelosa e melhor analisada. Questões de desempenho nos elementos gerenciados e o consumo de banda da rede com o tráfego imposto pelo uso de WS poderiam impedir uma solução de gerenciamento efetiva. Também não é adequado acreditar que o suporte a WS será encontrado em todos os elementos do processo de gerenciamento de rede. Por exemplo, dispositivos baseados em SNMP certamente ainda restarão nas futuras redes. Dessa forma, acreditamos que o cenário de gerenciamento baseado em WS mais comum será aquele onde as tecnologias estabelecidas de gerenciamento (como SNMP) coexistirão com WS. Nesse cenário, também acreditamos que WS serão mais freqüentemente encontrados próximos à interface de administração de rede, enquanto que SNMP será, obviamente, encontrado nas interfaces dos dispositivos (Figura 1). Uma questão importante surge aqui: onde, nos futuros sistemas, estará a linha entre WS e as tecnologias estabelecidas de gerenciamento?

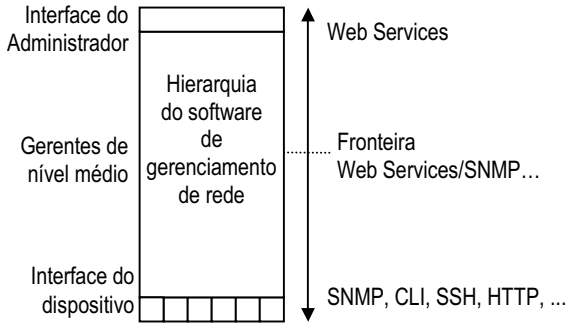


Figura 1. Hierarquia do software de gerenciamento de rede

Neste artigo, novas medições de desempenho foram realizadas para verificar a viabilidade dos WS em gerenciamento. Essas medições estendem um trabalho anterior [Neisse et al. 2004] que se deteve especialmente na avaliação do tráfego gerado na rede, onde foram estabelecidas comparações com o SNMPv1. Assim como no trabalho anterior, novamente utilizou-se variações com compressão e criptografia de dados nas medições realizadas. Os novos aspectos de desempenho observados neste artigo incluem o consumo de banda da rede e o tempo de resposta para as operações realizadas. Entretanto, agora os resultados de desempenho dos *gateways* são comparados com as versões 1 e 3 do protocolo SNMP (SNMPv1 e SNMPv3).

O restante deste artigo está organizado como segue. A seção 2 apresenta trabalhos relacionados e discute sobre WS e *gateways* para SNMP atualmente propostos. A seção 3 introduz duas abordagens para *gateways* WS para SNMP. A seção 4 apresenta o cenário de avaliação e os resultados obtidos. Finalmente, a seção 5 finaliza este artigo apresentando as conclusões e possíveis trabalhos futuros.

2. Web Services e *Gateways* para SNMP

Web Services (WS) podem ser descritos como um conjunto de componentes independentes, os quais são disponibilizados na Internet utilizando protocolos Web (ex.: HTTP, SMTP e FTP) e que recebem invocações de serviços de clientes. Os clientes de um WS podem ser aplicações de usuários finais ou mesmo outros WS. Nesse último caso, um WS pode requisitar operações de outro WS, permitindo construir uma hierarquia de requisições. Além disso, através de mecanismos conhecidos como orquestração e coreografia de WS, é possível a criação de WS bastantes complexos baseados na invocação de outros mais simples. A arquitetura geral de WS (Figura 2) é composta por três elementos principais: Registro, Provedor de serviços e Cliente.

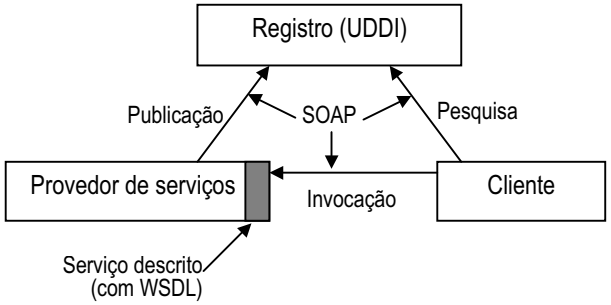


Figura 2. Arquitetura geral de Web Services

Para implementar os elementos dessa arquitetura, as principais tecnologias utilizadas, atualmente, são: UDDI (*Universal Description, Discovery, and Integration*), que tem a função de atuar como o registro dos WS; WSDL (*Web Services Description Language*), um padrão para descrição dos WS; e SOAP, um protocolo baseado em XML utilizado para comunicação entre os elementos da arquitetura WS. Analisando mais detalhadamente, UDDI [Bellwood et al. 2002] é um serviço de diretório emergente que trabalha como um repositório de dados para registrar e armazenar descrições de WS e informações de localização. O próprio registro UDDI é implementado como um WS, cujos serviços oferecidos são o cadastro e a pesquisa no repositório de dados. Dessa forma, clientes exploram o repositório UDDI, pesquisando por operações e WS disponíveis. Uma vez encontrado o WS apropriado, o cliente contata esse WS, invocando suas operações. Uma das informações que podem ser encontradas no registro UDDI é a localização do arquivo de descrição do WS. De posse desse arquivo, o cliente pode aprender os detalhes dos serviços oferecidos pelo WS e criar formas de acesso aos mesmos em tempo de execução. WSDL [Chinnici et al. 2003] é usada para este propósito, ou seja, descrever as informações necessárias para a realização de acesso a um determinado serviço. Todas as comunicações entre cliente e UDDI, WS e UDDI, e cliente e WS são executadas através do protocolo SOAP [Mitra 2003].

Para usar a arquitetura de WS no gerenciamento de redes, processos de tradução devem ser introduzidos. Esses processos são necessários para traduzir as informações de gerenciamento obtidas através dos protocolos estabelecidos em informações oferecidas via WS. Uma maneira comum de implementar esses processos de tradução é utilizando *gateways* de protocolos nos sistemas de gerenciamento.

Yoon-Jung Oh et al. [Oh et al. 2002] definem *gateways* XML para SNMP e três métodos de tradução interativa: baseadas em DOM (*Document Object Model*), em HTTP e em SOAP. Nas traduções baseadas em DOM, um gerente com suporte a XML chama uma interface DOM residente no *gateway*. Tais chamadas são traduzidas em operações SNMP entre o *gateway* e o dispositivo alvo. Na tradução baseada em HTTP, o *gateway* recebe expressões XPath e XQuery codificadas por um gerente com suporte a XML. Essas expressões são então traduzidas para requisições SNMP. Esse método de tradução é especialmente interessante porque a filtragem de informação pode ser executada diretamente no *gateway*, reduzindo o conjunto de informações de gerenciamento entre o gerente com suporte a XML e o *gateway*, embora uma sobrecarga de processamento seja introduzida. Finalmente, na tradução baseada em SOAP, o *gateway* oferece serviços mais sofisticados, que são acessados pelo gerente com suporte a XML. Nesses serviços, o gerente pode pesquisar informações com XPath ou prosseguir com consultas complexas através de expressões XQuery.

Strauss e Klie [Strauss and Klie 2003] propuseram um *gateway* XML para SNMP similar ao método de tradução de Yoon-Jung Oh et al. O *gateway* aceita mensagens HTTP com expressões XPath na URL. As expressões são então verificadas e traduzidas para mensagens SNMP. DOM é usado para acessar os documentos XML dentro dos *gateways*, reduzindo os dados transferidos entre o gerente e o *gateway*. Em operações de escrita, mensagens POST são traduzidas para requisições SNMP SetRequest. *Traps* SNMP são suportadas dentro do *gateway* através de um *buffer* para *traps* que é acessado pelo gerente. Nesse processo, mensagens POST são enviadas pelo *gateway* para receptores (*listeners*) HTTP nos gerentes baseados em XML.

Em um trabalho anterior, foi implementado um sistema [Neisse et al. 2003] que, dado um arquivo SMI (*Structure of Management Information*) de uma MIB (*Management Information Base*), cria automaticamente *gateways* XML para SNMP. Os *gateways* criados consultam informações nos dispositivos e geram documentos XML, que são enviados de volta para o gerente, onde são analisados através de *parsers*. Como no trabalho anterior de Strauss and Klie, a tradução é executada com a ajuda da ferramenta `smidump` [Strauss 2003], a qual gera uma versão XML de arquivos SMI.

Como verificado, *gateways* são criados para acessar dispositivos baseados em SNMP e exportar informações de gerenciamento em documentos XML (WS reais, baseados em SOAP, dificilmente são usados). Adicionalmente, a descrição de WS através de WSDL e seu registro em UDDI não são abordados nesses trabalhos. Na próxima seção, serão apresentadas duas abordagens para *gateways* SNMP, que melhor exploram as facilidades introduzidas pela arquitetura de WS. Após isso, também serão apresentados os resultados obtidos com a avaliação de tais abordagens quanto à banda consumida e ao tempo de resposta. Tais resultados são comparados com os resultados obtidos pelo uso dos protocolos SNMPv1 e SNMPv3.

3. *Gateways* WS para SNMP em Nível de Protocolo e Objeto

Neste trabalho, dois níveis diferentes de *gateways* WS para SNMP são considerados: *gateways* em nível de protocolo e *gateways* em nível de objeto [Neisse et al. 2004]. Primeiro, o *gateway* em nível de protocolo mapeia diretamente as primitivas SNMP para operações WS (ex.: Get, GetNext, Set). Um *gateway* em nível de objeto, por sua vez, oferece operações que refletem a estrutura das informações de gerenciamento. Nesse caso, GetIfTable é um exemplo de uma operação oferecida por um *gateway* em nível de objeto. Nesta seção, serão apresentados: (a) um *gateway* WS para SNMP em nível de protocolo e (b) um sistema que, dado um arquivo de MIB de entrada, automaticamente cria novos *gateways* WS para SNMP em nível de objeto.

3.1. O *Gateway* WS para SNMP em Nível de Protocolo

O *gateway* WS para SNMP em nível de protocolo fornece operações que são mapeamentos diretos das primitivas SNMP. Um gerente baseado em WS requisita informações de gerenciamento acessando o *gateway* WS para SNMP através de mensagens SOAP, sendo que, no protótipo atual, as mensagens SOAP trafegam usando HTTP ou HTTPS. Já os servidores que hospedam os *gateways*, recebem do gerente, a identificação da operação a ser acessada (ex.: Get ou Set) e uma lista dos parâmetros relacionados ao SNMP (o endereço do dispositivo alvo, uma comunidade SNMP válida e OIDs SNMP). Com essas informações, a operação apropriada é invocada dentro do *gateway* WS e o dispositivo alvo é acessado via SNMP.

Foram implementadas as operações Get, GetNext e Set, que geram, para cada requisição do gerente, exatamente uma requisição SNMP do *gateway* para o dispositivo alvo, e exatamente uma resposta do dispositivo alvo para o *gateway*. Após as informações SNMP serem obtidas do dispositivo, o *gateway* monta uma mensagem SOAP com tais informações e envia essa mensagem de volta para o gerente.

As operações implementadas permitem verificar, como será apresentado na próxima seção, o impacto de um *gateway* em nível de protocolo sobre a rede gerenciada. Posteriormente, mais operações poderiam, obviamente, ser incluídas (ex.:

GetBulk). O *gateway* WS para SNMP em nível de protocolo foi implementado como um *script* PHP4 usando, para o suporte ao SOAP, a biblioteca NuSOAP [Ayala 2004]. A Figura 3 apresenta as operações fornecidas pelo *gateway* em nível de protocolo (as operações foram implementadas como funções em PHP4).

```
function Get     ($ip, $community, $oid)
function GetNext ($ip, $community, $oid)
function Set     ($ip, $community, $oid, $newvalue)
```

Figura 3. Operações do *gateway* WS para SNMP em nível de protocolo

Em um uso simples desse *gateway*, suas operações (Get, GetNext e Set) foram descritas em WSDL e registradas em um repositório UDDI. Um gerente baseado em WS, procurando por WS de gerenciamento, pode pesquisar no UDDI e descobrir a localização dos *gateways* disponíveis. É importante notar que, com esta abordagem, o gerente baseado em WS ainda deve estar ciente dos OIDs SNMP para, corretamente, requisitar as instâncias dos objetos ao *gateway*. A vantagem dessa abordagem, entretanto, é o fato de que, toda vez que novos objetos passam a ser suportados por um agente SNMP, o *gateway* WS para SNMP não precisará ser alterado. A desvantagem, por outro lado, vem do fato do gerente ser obrigado a conhecer os objetos SNMP suportados em cada dispositivo gerenciado, mesmo na presença de UDDI. Além disso, o gerente ainda precisa lidar com OIDs SNMP, pois tais *gateways* não possibilitam operações de uso mais fácil do que aquelas encontradas atualmente usando SNMP.

3.2. O Sistema de Criação de *Gateways* WS para SNMP em Nível de Objeto

Um *gateway* WS para SNMP em nível de objeto, diferentemente do apresentado anteriormente, "conhece" os objetos de MIB suportados pelo dispositivo alvo, e apresenta tais objetos como operações de WS. Por exemplo, uma operação GetIfTable é uma operação que obtém a tabela completa de interfaces, enquanto que SetAdminStatus é uma operação que muda, no dispositivo alvo, o estado administrativo de uma das interfaces de rede disponíveis. Uma vantagem do *gateway* em nível de objeto é que ele não apenas consulta e expõe as informações como um WS, mas também possui somente as operações permitidas para serem executadas sobre o dispositivo alvo. Dessa forma, um gerente baseado em WS não apenas pesquisa o UDDI em busca de WS de gerenciamento geral, mas procura por WS especializados para dispositivos específicos. Portanto, o gerente baseado em WS não é forçado a manter uma lista das capacidades de cada dispositivo: isto está informado no *gateway* utilizado.

Essa abordagem, entretanto, perde flexibilidade quando o agente SNMP do dispositivo alvo é alterado (tanto para incluir como para remover objetos). Nesse caso, o WS associado precisa, de fato, ser refeito para refletir as mudanças do agente SNMP. Sendo assim, é preciso uma maneira eficiente para criar *gateways* WS para SNMP em nível de objeto. Para tal, foi desenvolvido um sistema que, dado um arquivo de MIB SMI, cria um novo WS. A Figura 4 apresenta a arquitetura que suporta a criação de novos *gateways* WS para SNMP em nível de objeto. O primeiro passo na criação do *gateway* é a transferência do arquivo de MIB (codificado em SMIv1 ou SMIv2) do gerente baseado em WS para o servidor Web, através de HTTP/HTTPS.

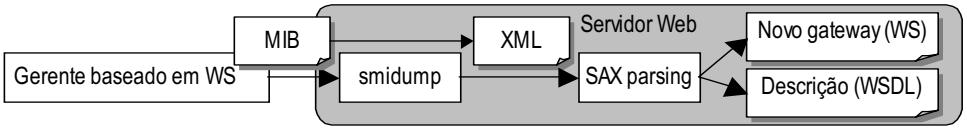


Figura 4. Criação de um novo *gateway* WS para SNMP em nível de objeto

Internamente no servidor, a ferramenta `smidump` verifica a MIB passada e, na ausência de inconsistências, gera um XML temporário (que é a versão XML da MIB). No próximo passo, é feito um *parsing* do documento XML temporário para construir o novo *gateway*. Cada nó da árvore da MIB original é transformado em operações WS. Nessas operações, está incluído código para contatar, via SNMP, o dispositivo alvo. O dispositivo alvo e sua string de comunidade são tratados dentro do código como parâmetros, cujos valores serão posteriormente passados, quando a operação for invocada pelo gerente. O WS recém criado é armazenado em um diretório padrão no servidor Web e disponibilizado para ser invocado logo após sua criação. O arquivo de MIB original recebido também é armazenado em outro diretório padrão, para fins de documentação, bem como o documento XML intermediário gerado pela ferramenta `smidump`. Ao mesmo tempo em que a etapa de *parsing* cria o código do novo WS, ela também cria o documento WSDL que descreve o WS criado. Além disso, o gerente baseado em WS que requisitou a criação de um novo serviço, opcionalmente, informa a URL do repositório UDDI onde espera que o WS criado seja registrado.

Com o processo de criação de *gateways*, novas MIBs podem facilmente ser adicionadas ao ambiente de gerenciamento baseado em WS. Para que esse processo funcione corretamente, entretanto, os arquivos de MIB devem ser definidos corretamente de acordo com SMIv1 e SMIv2. Porém, não é raro se encontrar arquivos de MIB com problemas de definição. Nesse caso, o *gateway* WS para SNMP não será criado, e a correspondente mensagem do `smidump` descrevendo os erros encontrados será enviada de volta para o gerente baseado em WS. É importante notar que o sistema de criação de *gateway* propriamente dito não é um WS, mas um conjunto de *scripts* PHP4 rodando no mesmo servidor Web que hospeda os *gateways* recém criados.

Quando uma requisição é disparada pelo gerente baseado em WS, o *gateway* WS a ser usado é determinado na URL passada na requisição HTTP/HTTPS. O dispositivo alvo e a comunidade SNMP são passados como argumentos para as operações do WS. Cada nó do arquivo de MIB original é traduzido para operações do WS. Para objetos escalares, uma operação Get é sempre construída. Ela utiliza mensagens GetRequest SNMP para obter, do agente SNMP, o objeto requisitado. Uma operação Set pode ser criada se este for um objeto de escrita. Nesse caso, a operação usa uma mensagem SNMP SetRequest para alterar o valor de um objeto no dispositivo alvo. A Figura 5 mostra as operações criadas para o objeto sysLocation da MIB-II.

```
function GetSysLocation ($ip, $community)
function SetSysLocation ($ip, $community, $newvalue)
```

Figura 5. Operações para o objeto da MIB-II sysLocation

Uma operação Get também será criada para objetos de tabelas. Uma operação Set, entretanto, somente será criada para objetos de tabela que são de escrita. Uma diferença importante entre operações para objetos escalares e de tabelas é a identificação das instâncias dos objetos. No caso de objetos escalares, existe apenas uma instância. Mas,

para objetos de tabela, diversas instâncias podem existir. Além disso, tais instâncias têm que ser determinadas para terem seus valores obtidos através de operações Get, ou para permitir que seus conteúdos sejam modificados por operações Set. A Figura 6 apresenta as operações criadas para o objeto ifAdminStatus do grupo interface da MIB-II.

```
function GetIfAdminStatus ($ip, $community, $index=-1)
function SetIfAdminStatus ($ip, $community, $index, $newvalue)
```

Figura 6. Operações para o objeto da MIB-II ifAdminStatus

O parâmetro `index` é opcional em operações Get. Se não for fornecido, a operação retorna todas as instâncias associadas. Do contrário, somente a instância identificada é retornada. No caso de operações Set, o parâmetro `index` é obrigatório.

Tendo sido apresentados os dois tipos de *gateways* WS para SNMP e suas particularidades de implementação e funcionalidade, na próxima seção serão apresentados os resultados das medições realizadas neste trabalho.

4. Avaliando o Desempenho: SNMP versus *Gateways*

Na seção anterior, *gateways* WS para SNMP foram apresentados. Esses *gateways* permitem o gerenciamento de dispositivos baseados em SNMP utilizando WS. Entretanto, o uso de WS via *gateways* introduz, inevitavelmente, inconvenientes que poderiam impedir um efetivo gerenciamento de rede baseado em WS. O desempenho fim-a-fim entre gerente e dispositivo é diretamente afetado pela sobrecarga de processamento introduzido pelo suporte ao SOAP, necessário no gerente e no *gateway*. Sendo assim, nesta seção, são analisados e comparados aspectos de desempenho, tanto dos *gateways* quanto dos protocolos SNMPv1 e SNMPv3. Os aspectos de desempenho avaliados foram o consumo de banda e o tempo de resposta, além do tráfego total entre o gerente e o dispositivo gerenciado. A banda consumida foi calculada através da divisão do tráfego gerado pelo tempo médio de resposta. Para avaliar o tempo médio de resposta, criou-se um *script* que repetia 50 vezes a medição do tempo de resposta, sendo que entre uma medição e outra foi inserido um intervalo de 1 segundo.

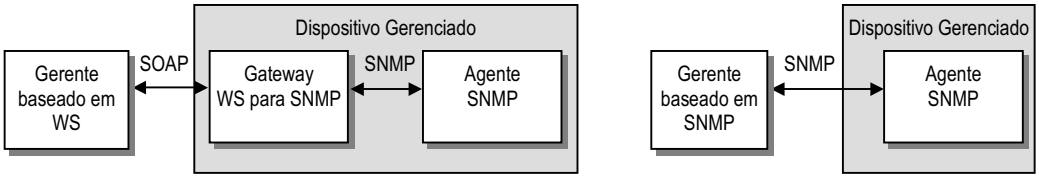
Para realizar a avaliação dos aspectos de desempenho, uma MIB de teste foi criada. Essa MIB é composta exclusivamente por um objeto do tipo tabela, que contém apenas uma coluna do tipo inteiro. Dessa forma, foram feitas medições de desempenho na recuperação dessa tabela, com o seu tamanho variando de 10 até 110 linhas.

4.1. Cenários de Avaliação

As medições foram feitas considerando dois tipos principais de cenários, conforme ilustrado na Figura 7. No cenário (a), analisou-se o desempenho dos dois tipos de *gateways* apresentados anteriormente. Já no cenário (b), foram avaliadas duas versões do protocolo SNMP: SNMPv1 e SNMPv3.

Como o objetivo no cenário (a) era concentrar a avaliação apenas no desempenho da comunicação gerente-*gateway*, o dispositivo gerenciado hospeda, além do agente SNMP, os *gateways* em um servidor HTTP interno a esta máquina. Isso objetiva minimizar a influência, principalmente em termos de tempo de resposta, que seria causada pela comunicação entre *gateway* e agente SNMP, caso estivessem localizados em máquinas diferentes, o que se supõe ser o cenário real de aplicação dos

gateways. Tanto o gerente quanto o dispositivo gerenciado são PCs interligados diretamente via um cabo cruzado, cujas configurações são mostradas na Figura 8.



(a) (b)

Figura 7. Cenários de avaliação

Os aspectos de desempenho foram avaliados considerando a troca de mensagens entre um gerente e *gateways* WS para SNMP (em nível de protocolo e objeto), conforme Figura 7a, e entre um gerente baseado em SNMP e o agente SNMPv1 e SNMPv3, conforme Figura 7b. Para verificar o desempenho dos *gateways* e do protocolo SNMP em função da quantidade de objetos consultados, foi testado o retorno de todas as linhas da MIB de teste citada anteriormente. Nesse caso, foram observadas mensagens SNMP com apenas um objeto no campo Variable Bind List, que é um dos usos mais freqüentes de mensagens SNMP quando da recuperação de tabelas via operações GetNext. Além disso, embora operações Set estejam disponíveis nos *gateways*, foram somente verificadas operações do tipo Get. Considerando que o tráfego de configuração (que usa Set) consome menos recursos da rede que o tráfego de monitoração (que usa Get), operações do tipo Get causariam impacto mais significativo na rede. Nesse ponto, torna-se importante acrescentar duas novas características ao cenário de avaliação: a compressão e a criptografia das mensagens. Os mecanismos empregados para garantir essas novas características são apresentados nas subseções seguintes.

	Gerente	**Dispositivo Gerenciado**
Processador	AMD Athlon XP 2400+	Intel Pentium 4 2.40GHz
Memória RAM	1GB	256MB
Sistema Operacional	Fedora Core 2 (2.6.5-1.358)	Red Hat 9 (2.4.20-8)
Servidor Web	Apache (2.0.49-4)	Apache (2.0.40-21)
PHP	4.3.8-2.1	4.2.2-17
SNMP	NET-SNMP (5.1.1-2)	NET-SNMP (5.0.9-2.90.1)

Figura 8. Configurações dos PCs utilizados

4.2. Aumentando a Segurança com Criptografia de Dados

Considerando a importância da segurança em qualquer sistema computacional, é interessante adicionar essa característica aos *gateways*, ainda mais considerando que mensagens de gerenciamento podem cruzar diferentes domínios administrativos. Nesse caso, a avaliação do impacto no desempenho com o uso de criptografia sobre as mensagens SOAP e SNMP necessita ser igualmente investigado.

Neste trabalho, a maneira empregada para permitir criptografia de dados nos *gateways* é através do uso de um protocolo seguro para transporte das mensagens SOAP. Como a biblioteca NuSOAP utiliza HTTP para transporte de dados, a versão segura desse protocolo, o HTTPS, foi usada nas comunicações entre o gerente e o *gateway*. Para avaliar a utilização da criptografia no protocolo SNMP, usou-se o SNMPv3, com suporte a autenticação e criptografia habilitado.

4.3. Reduzindo o Tráfego com Compressão de Dados

Esperando reduzir o tráfego gerado pelo protocolo SOAP, um processo de compressão de dados também foi incluído e teve seu impacto no desempenho igualmente avaliado. Em tal processo, as mensagens SOAP são compactadas em ambas as direções: do gerente para o *gateway* (na invocação do serviço) e do *gateway* para o gerente (no recebimento da resposta). Para implementar esse processo de compressão, o algoritmo ZLIB [Deutsch and Gailly 1996] foi usado. O suporte a esse algoritmo pode ser facilmente encontrado na linguagem PHP (usada nas implementações do gerente baseado em WS e dos *gateways*).

4.4. SNMP versus *Gateway* em Nível de Protocolo

Na primeira verificação, o interesse é avaliar quanto tráfego é gerado e quanto de banda é consumida pelas mensagens SOAP, além do tempo de resposta, quando o *gateway* em nível de protocolo é usado como mapeamento direto de mensagens GetNext. Esses resultados foram comparados aos obtidos pelo SNMPv1 e SNMPv3.

Nesse caso, o gerente percorre a tabela da MIB de teste do dispositivo alvo requisitando, sucessivamente, a operação GetNext no *gateway* em nível de protocolo. Cada requisição da operação GetNext no *gateway* gera uma requisição GetNext SNMP do *gateway* para o dispositivo alvo. Dessa forma, é o gerente que controla como a MIB é percorrida. Portanto, sendo **n** o número de linhas na tabela, a recuperação de todas instâncias de uma coluna desta tabela exigirá **(n+1)** pares solicitação-resposta SOAP. A Figura 9 mostra os resultados da avaliação de desempenho do *gateway* em nível de protocolo, comparando com o SNMPv1 e SNMPv3. Na comunicação entre o gerente e o *gateway*, foi também avaliado o uso de compressão e de criptografia de dados.

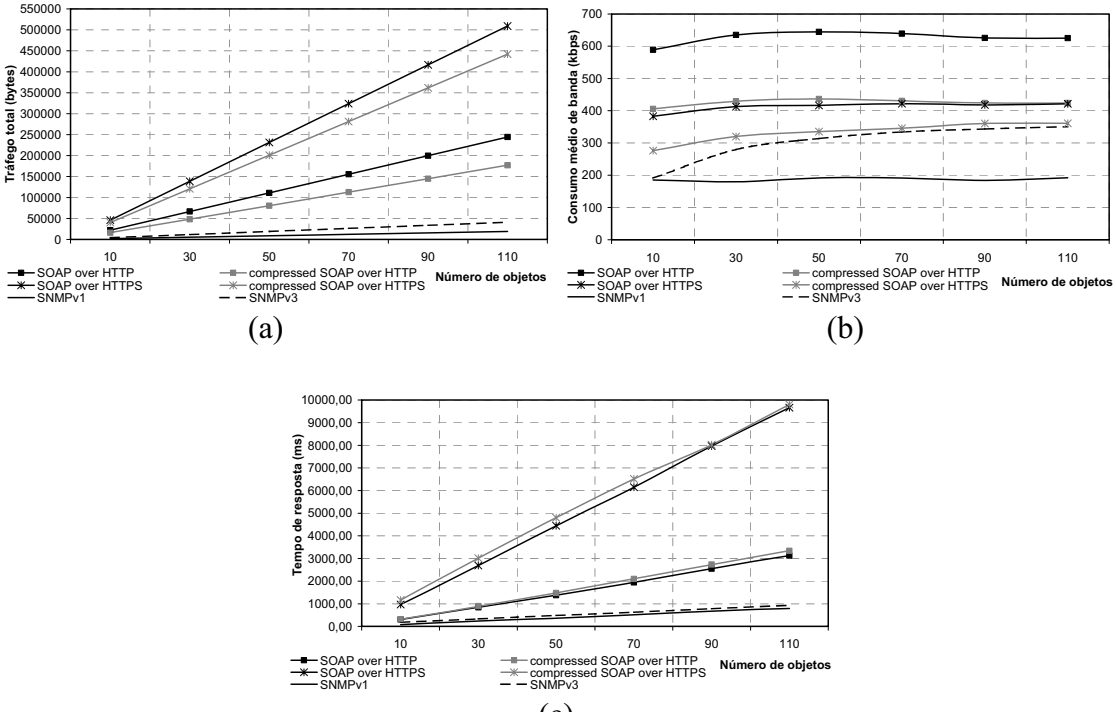


Figura 9. SNMP versus *Gateway* em Nível de Protocolo

Pode-se observar no gráfico da Figura 9a que, quanto maior o número de objetos a serem recuperados, maior será a diferença entre o tráfego SOAP e SNMP (independente do uso, ou não, de compressão e criptografia). O mesmo vale para o tempo de resposta (Figura 9c). Ou seja, o tráfego gerado pelo SNMPv1 e SNMPv3, bem como os respectivos tempos de resposta, são sempre menores que aqueles obtidos com o uso desse tipo de *gateway*. Já no caso do consumo de banda (Figura 9b), pode-se observar que esse consumo se mantém aproximadamente constante em função do aumento do número de instâncias recuperadas. Isso se deve ao fato de que a razão entre o tráfego gerado pela consulta e o tempo gasto para executar a mesma se mantém aproximadamente constante em função do tamanho da tabela recuperada. Por fim, observa-se também que o SNMPv1 obteve os melhores resultados, em todos os aspectos avaliados.

4.5. SNMP versus *Gateway* em Nível de Objeto

Na segunda verificação, o interesse está focado em avaliar o quão factível é o uso das operações do *gateway* em nível de objeto. Todas as operações em tais *gateways* são implementadas através de uma seqüência de mensagens SNMP GetNext. Nesse caso, o caminhamento na MIB é coordenado pelo *gateway*, e não pelo gerente (como acontecia no *gateway* em nível de protocolo). Diferentemente do que ocorre no *gateway* em nível de protocolo, no *gateway* em nível de objeto o número de mensagens SOAP e SNMP não será o mesmo. Para uma tabela de **n** linhas, continuarão a ser geradas **(n+1)** pares solicitação-resposta SNMP. Porém, apenas **1** par solicitação-resposta SOAP será gerado, pois o gerente solicita o serviço correspondente à recuperação da coluna inteira da tabela, e recebe do *gateway* uma única mensagem com todas as linhas da coluna solicitada. Os resultados do *gateway* em nível de objeto são apresentados na Figura 10.

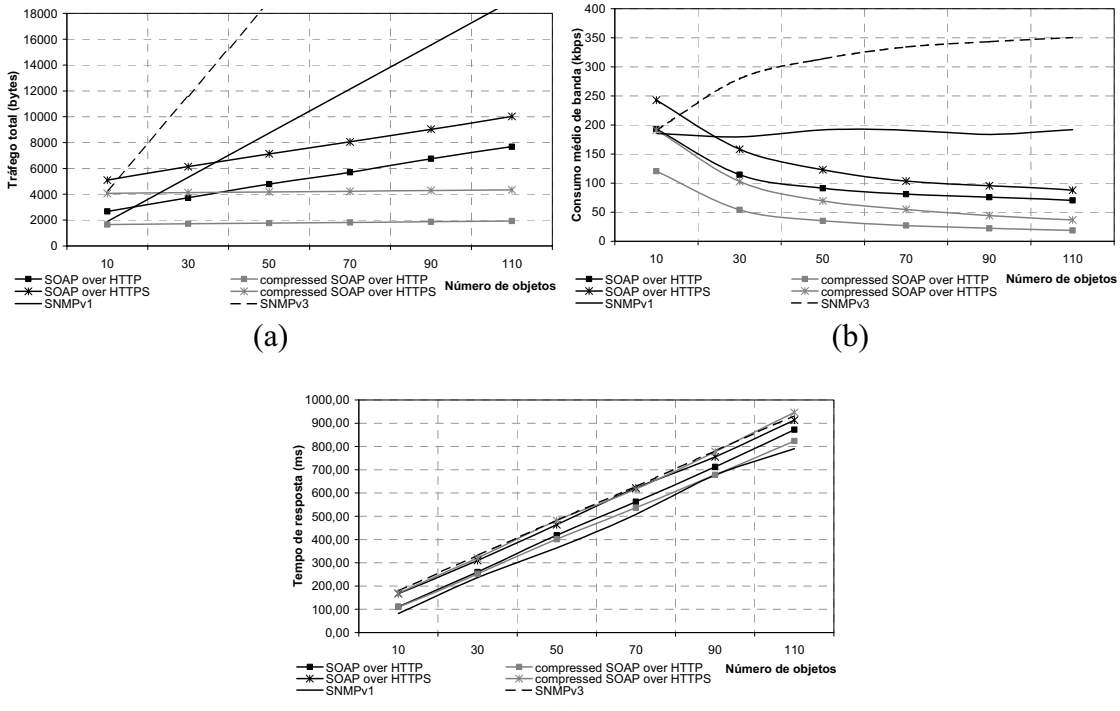


Figura 10. SNMP versus *Gateway* em Nível de Objeto

Para todos os casos (com e sem o uso de compressão e criptografia de dados), o tráfego gerado pelo *gateway* em nível de objeto cresce a uma taxa consideravelmente mais baixa que o SNMPv1 e o SNMPv3 (Figura 10a), em função do aumento do número de instâncias recuperadas. Esse comportamento decorre da vasta interação do SNMP requerida na comunicação entre o gerente e o dispositivo gerenciado, enquanto que, na comunicação entre o gerente e o *gateway*, apenas duas mensagens SOAP são trocadas: a requisição da operação no WS e sua resposta. Como conseqüência disso, existe, para cada variação do *gateway*, um ponto, a partir do qual, o tráfego WS passa a ser menor que correspondente SNMP. Em relação ao tempo de resposta (Figura 10c), pode-se concluir que este cresce de forma linear, seguindo uma taxa aproximadamente igual tanto para o *gateway* (com suas variações) quanto para o SNMP (em ambas as versões). Além disso, a diferença no tempo de resposta entre o *gateway* e o SNMP é bastante baixa, ficando, na avaliação realizada, em aproximadamente 100ms. Como o tempo de resposta do *gateway* é aproximadamente igual ao do SNMP, mas o tráfego gerado é menor (a partir de um certo momento), a banda consumida pelo *gateway* em nível de objeto é menor que àquela consumida pelo SNMP (Figura 10b).

5. Conclusões e Trabalhos Futuros

Neste artigo, duas abordagens para *gateways* WS para SNMP foram apresentadas: em nível de protocolo e de objeto. Os *gateways* em nível de protocolo são bastante simples e fornecem um mapeamento das primitivas do SNMP para operações de WS (como GetNext e Set). Os *gateways* em nível de objeto são mapeamentos das estruturas da MIB para operações de WS (como GetIfTable). Nesse último caso, foi utilizado o sistema desenvolvido por Neisse et al. [Neisse et al. 2004], o qual automaticamente gera novos *gateways* a partir de um arquivo de MIB. O *gateway* em nível de protocolo obriga o gerente baseado em WS a continuar lidando com OIDs. Ou seja, não existe uma nova facilidade oferecida pelos *gateways*. Ambos os *gateways* foram comparados com duas versões do SNMP (v1 e v3), levando-se em conta três aspectos de desempenho: tráfego total, banda consumida e tempo de resposta. Acrescentou-se também na comparação, a utilização de criptografia e compressão nas mensagens de gerenciamento.

Conforme verificado nas Figuras 9a e 9c, o tráfego e o tempo de resposta do *gateway* em nível de protocolo são maiores que o tráfego e o tempo de resposta SNMP em todos os casos. Isso decorre do fato do número de mensagens SNMP e SOAP geradas ser igual. Portanto, como cada mensagem individual SOAP é sempre maior e mais demorada que a correspondente SNMP (mesmo com o uso de compressão de dados), o desempenho do *gateway* será sempre pior. O mesmo ocorre em relação ao consumo de banda (Figura 9b), onde o *gateway* apresentou um consumo bem maior que o SNMPv1. Por outro lado, o SNMPv3 teve desempenho, nesse aspecto, bastante próximo ao do *gateway* em nível de protocolo no seu melhor caso (SOAP sobre HTTPS com compressão), conforme o aumento na quantidade de objetos recuperados. Sendo assim, para uma pequena quantidade de consultas, o *gateway* em nível de protocolo, poderia ser utilizado já que, nesses casos, o tempo de resposta e o tráfego gerado são pequenos, apesar de serem maiores que o correspondente SNMP.

O *gateway* em nível de objeto foi o que apresentou os melhores resultados na maioria das medições realizadas. Como pode ser observado na Figura 10a, para cada

tipo de variação (com ou sem compressão/criptografia) do *gateway* em nível de objeto, existe um ponto a partir do qual o tráfego SNMPv1 ultrapassa o tráfego SOAP. Para SOAP sobre HTTP, esse ponto situa-se em torno de 17 objetos recuperados. Quando a criptografia é utilizada, via HTTPS, esse ponto desloca-se para 37 objetos. Observa-se também o efeito causado pelo uso de compressão de dados, reduzindo os dois valores anteriores para, respectivamente, 10 e 22 objetos. Esse desempenho superior do *gateway* se explica pelo fato do número de mensagens SOAP e SNMP geradas ser diferente. São sempre apenas duas mensagens SOAP (uma solicitação e uma resposta) contra um número crescente (em função do tamanho da tabela) de mensagens SNMP. No caso do tempo de resposta, o que se pode verificar é a pequena diferença entre o desempenho do *gateway* e do SNMPv1 e SNMPv3 (aproximadamente 100ms no pior caso). Observa-se pelo gráfico da Figura 10c, que o tempo de resposta aumenta a uma mesma taxa para todos os casos, sendo que o uso de criptografia e compressão de dados não alterou esse tempo de forma significativa. Já em relação à banda consumida, pode-se visualizar que a mesma, no caso do *gateway* em nível de objeto, diminui em função do aumento no número de objetos recuperados, conforme a Figura 10b. Isso acontece porque o tempo de resposta cresce a uma proporção maior que o crescimento do tráfego. Portanto, a razão entre o tráfego e o tempo de resposta diminui em função do aumento na quantidade de objetos recuperados.

Como trabalho futuro, atualmente está em desenvolvimento um novo tipo de *gateway*, denominado *gateway* em nível de serviço [Fioreze et al. 2005]. Tal *gateway* visa criar um grau maior de abstração na manipulação com MIBs, permitindo com isso uma interação mais simples e com o mesmo nível de controle sobre as MIBs. Além disso, ainda restam outros aspectos de desempenho a serem avaliados, como, por exemplo, consumo de CPU e memória, bem como outros algoritmos de compressão, além do ZLIB utilizado neste trabalho. A questão das notificações em um ambiente de gerenciamento baseado em WS foi abordada em um outro trabalho [Lima et al. 2006], onde foram comparados os desempenhos das notificações Web Services, usando a especificação *WS-Notification*, e das *traps* SNMP.

Referências

Ayala, D. (2004) “NuSOAP - Web Services Toolkit for PHP”, http://dietrich.ganx4.com/nusoap/, December.

Bellwood, T., Cl´ement, L., Ehnebuske, D., Hately, A., Hondo, M., Husband, Y. L., Januszewski, K., Lee, S., McKee, B., Munter, J. and von Riegen, C. (2002) “UDDI Version 3.0”, UDDI Spec Technical Committee Specification.

Case, J., Fedor, M., Schoffstall, M. and Davin, J. (1990) “A Simple Network Management Protocol (SNMP)”, RFC 1157, IETF.

Case, J., Mundy, R., Partain, D. and Stewart, B. (2002) “Introduction and Applicability Statements for Internet Standard Management Framework”, RFC 3410, IETF.

Chinnici, R., Gudgin, M., Moreau, J. and Weerawarana, S. (2003) “Web Services Description Language (WSDL) Version 1.2 Part 1: Core Language”, W3C Working Draft, W3C.

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N. and Weerawarana, S. (2002) “Unraveling the Web Services Web: an Introduction to SOAP, WSDL, and UDDI”,

In: IEEE Internet Computing, Vol. 6, Issue 2, p. 86-93.

Deutsch, P. and Gailly, J. (1996) "ZLIB Compressed Data Format Specification Version 3.3", RFC 1950, IETF.

Fioreze, T., Granville, L. Z., Almeida, M. J. B. and Tarouco, L. M. R. (2005) "Comparing Web Services with SNMP in a Management by Delegation Environment", In: 9th IFIP/IEEE International Symposium on Integrated Network Management (IM 2005).

Goth, G. (2002) "Grid Services Architecture Plan Gaining Momentum". In: IEEE Internet Computing, Vol. 6, Issue 4, p. 11-12.

Levi, D. and Schoenwaelder, J. (1999) "Definitions of Managed Objects for the Delegation of Management Scripts", RFC 2592, IETF.

Lima, W., Alves, R. S., Vianna, R. L., Almeida, M. J. B., Tarouco, L. M. R. and Granville, L. Z. (2006) "Evaluating the Performance of SNMP and Web Services NotificationsAccepted", In: 10th IFIP/ IEEE Network Operations and Management Symposium (NOMS 2006). (to be published)

MacFaden, M., Partain, D., Saperia, J. and Tackabury, W. (2003) "Configuring Networks and Devices with Simple Network Management Protocol (SNMP)", RFC 3512, IETF.

Mitra, N. (2003) "SOAP Version 1.2 Part 0: Primer", W3C Recommendation, W3C, http://www.w3.org/TR/2003/REC-soap12-part0-20030624/, December.

Neisse, R., Granville, L. Z., Almeida, M. J. B. and Tarouco, L. M. R. (2003) "A Dynamic SNMP to XML Proxy Solution", In: 8th IFIP/IEEE International Symposium on Integrated Network Management (IM 2003), p. 481-484.

Neisse, R., Vianna, R. L., Granville, L. Z., Almeida, M. J. B. and Tarouco, L. M. R. (2004) "Implementation and Bandwidth Consumption Evaluation of SNMP to Web Services Gateways", In: 9th IFIP/IEEE Network Operations and Management Symposium (NOMS 2004), p. 715-728.

Oh, Y., Ju, H., Choi, M. and Hong, J. (2002) "Interaction Translation Methods for XML/SNMP *Gateway*", In: 13th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 2002), p. 54-65.

Preece, A. and Decker, S. (2002) "Intelligent Web Services", In: IEEE Intelligent Systems, Vol. 17, Issue 1, p. 15-17.

Strauss, F. (2003) "libsmi - a Library to Access SMI MIB Information", http://www.ibr.cs.tu-bs.de/projects/libsmi/, August.

Strauss, F. and Klie, T. (2003) "Towards XML Oriented Internet Management", In: 8th IFIP/IEEE International Symposium on Integrated Network Management (IM 2003), p. 505-518.

Waldbusser, S. (1997) "Remote Network Monitoring Management Information Base Version 2 Using SMIv2", RFC 2021, IETF.

Waldbusser, S. (2000) "Remote Network Monitoring Management Information Base", RFC 2819, IETF.