Avaliação de Suporte de Notificações utilizando SNMP e Web Services em uma Arquitetura de Correlação de Eventos Distribuída

Evandro Della Vecchia Pereira, Lisandro Zambenedetti Granville, Maria Janilce Bosquiroli Almeida, Liane Margarida Rockenbach Tarouco

Instituto de Informática – Universidade Federal do Rio Grande do Sul
Caixa Postal 15064 – 90501-970 Porto Alegre, RS

{edvpereira, granville, janilce, liane}@inf.ufrgs.br

Abstract. *Currently, Web Services for network management have been an intensive field of investigation. However, the investigations carried up to do not properly address the issue related to event notifications. The Simple Network Management Protocol, which is the widely deployed and accepted solution, has an interesting but also limited support for event notification through its trap messages. In this paper we present a distributed event correlation architecture that allowed us to closely investigate the use of Web Services as a tool in network management considering the specific case of notification support. This study complements the investigations under development showing aspects of Web Services for network management that were unknown in the field of event notification.*

Resumo. *Atualmente, os Web Services têm sido amplamente tema de pesquisa no campo de gerenciamento de redes de computadores. Contudo, não há pesquisas propriamente relacionadas à notificação de eventos. O protocolo SNMP (Simple Network Management Protocol), que é a solução mais aceita e utilizada, possui suporte à notificação de eventos através de suas mensagens trap. Porém, esse suporte é limitado. Neste artigo, apresentamos uma arquitetura de correlação de eventos distribuída, que permite a investigação do uso de Web Services como ferramenta no gerenciamento de redes, considerando o caso específico de suporte à notificação de eventos. Este estudo complementa as investigações em andamento, mostrando aspectos dos Web Services no gerenciamento de redes, que eram desconhecidos no campo de notificação de eventos.*

1. Introdução

Atualmente, há um grande interesse, da comunidade de gerenciamento de redes, nas pesquisas relacionadas com o uso de *Web Services* (WS) como ferramenta de gerenciamento. Segundo Schönwälder et al. [Schönwälder et al., 2003], os *Web Services* representam uma revolução no gerenciamento de redes, porque seu uso sugere a substituição do framework SNMP [Case et al., 1990], que é atualmente largamente aceito e utilizado. Contudo, mesmo sendo revolucionário, o uso de *Web Services* no gerenciamento de redes pode não ser viável em sistemas reais que possuem dispositivos dependentes apenas do SNMP.

Com o intuito de integrar dispositivos que dependem exclusivamente do SNMP com sistemas de gerenciamento baseado em WS, algumas medidas são propostas. Uma medida

comum é baseada no uso de *gateways* que mapeiam (traduzem) mensagens, informações e/ou serviços SNMP para operações WS. Estes *gateways* são baseados em WS implementados através do uso do protocolo SOAP (Simple Object Access Protocol), que geralmente utiliza o protocolo HTTP como transporte. Essa ligação SOAP/HTTP é facilmente realizada devido ao grande número de ferramentas de desenvolvimento e suporte disponíveis atualmente.

A utilização de SOAP sobre HTTP implica uma comunicação síncrona, onde o cliente utilizando uma API, invoca uma operação WS que é executada em um servidor localizado em um computador remoto. Geralmente, o cliente fica bloqueado, aguardando por uma resposta do servidor. Observando o tráfego na rede, pode-se observar que uma mensagem SOAP primeiramente é enviada do cliente ao servidor para realizar a chamada à operação WS, e uma mensagem final é enviada do servidor ao cliente, contendo os resultados da operação. Essas duas mensagens lembram as mensagens *request/response* do SNMP, indicando então que interações cliente/servidor utilizando WS são semelhantes às interações agente/gerente SNMP. Deste modo, *gateways* SNMP/WS podem ser construídos através do uso da ligação SOAP/HTTP, mapeando as requisições WS do cliente para uma requisição SNMP (*Get*, *GetNext*, *Set*, etc.), e mapeando as respostas SNMP (*GetResponse*) para WS ao servidor. Contudo, a natureza síncrona da ligação SOAP/HTTP não é apropriada para mensagens de notificação SNMP (*traps*), e muito pouca (se alguma) investigação nesse tema tem sido realizada.

Nesse contexto, ligações SOAP alternativas podem ser utilizadas para suprir o suporte de notificações SNMP em *gateways* SNMP/WS. Este artigo apresenta uma investigação do uso de SOAP sobre SMTP para entrega de mensagens de notificação SNMP aos gerentes baseados em WS. Para isso, foi definida uma arquitetura de notificação distribuída (que é capaz de correlacionar notificações de eventos) composta por *gateways* de notificação SNMP/WS e encaminhadores de notificações baseados em políticas. Como sugere o nome, os *gateways* de notificação mapeiam notificações SNMP para chamadas WS assíncronas, enquanto os encaminhadores correlacionam e encaminham notificações, baseados em políticas definidas pelo administrador da rede. A solução proposta foi avaliada quanto ao volume de tráfego gerado na rede e o retardo de transmissão, bem como foram discutidos os prós e contras da utilização de WS como tecnologia para notificação de eventos da rede. As contribuições principais deste artigo são duas: a observação da ligação de SOAP com outro protocolo além do HTTP, no contexto de gerenciamento de redes, e a implementação de uma arquitetura de notificações baseada em políticas. Embora notificações baseadas em políticas estejam presentes na solução proposta (como pode ser visto na arquitetura apresentada na seção 3), este artigo está focado na avaliação das diferenças entre SNMP e a ligação SOAP/SMTP para notificação de eventos.

O restante deste artigo está organizado como segue. A seção 2 apresenta os trabalhos relacionados, revisando os conceitos de *Web Services* para o gerenciamento de redes, gerenciamento baseado em políticas e correlação de eventos. A seção 3 apresenta a arquitetura de correlação de eventos, enquanto a seção 4 apresenta o protótipo da implementação do sistema. A seção 5 mostra resultados do uso da arquitetura proposta em uma rede de teste, e a seção 6 encerra o artigo com conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Nesta seção são revisados *Web Services* para o gerenciamento de redes, gerenciamento de redes baseado em políticas (Policy Based Network Management - PBNM) e correlação de eventos.

2.1. Web Services para Gerenciamento de Redes

Desenvolvido e padronizado pelo *World Wide Web Consortium* (W3C) [Consortium, 2004], *Web Services (WS)* são componentes de aplicação independentes, baseados na Web, que outras aplicações podem localizar e utilizar. WS variam de simples a complexos e prometem flexibilidade e computação distribuída na Internet [Roy and Ramanujan, 2001].

No contexto de gerenciamento de redes, G. Pavlou et al. [Pavlou et al., 2004] define que os WS possuem grande semelhança com CORBA. WS suportam interações de chamadas remotas dos tipos solicitação (*request*), resposta (*response*) e erro (*error*).

Outras investigações têm sido realizadas com a intenção de observar aspectos relacionados ao desempenho dos WS no gerenciamento de redes. Jeroen van Sloten et al. [v. Sloten et al., 2004] investigam como *gateways* SNMP/WS podem reduzir o consumo de banda. Nos nossos trabalhos de pesquisa temos avaliado *gateways* SNMP/WS no mapeamento de mensagens originais SNMP para operações WS [Neisse et al., 2004] [Fioreze et al., 2005].

No geral, é possível afirmar que as investigações atuais observam o tráfego dos WS e o desempenho associado quando há a tradução e integração (utilizando algum enfoque) de operações/informações SNMP em um ambiente de gerenciamento de rede que permite a utilização de WS. Se dividirmos o SNMP em mensagens *request/reply* e mensagens assíncronas (*traps*), as mensagens *request/reply* são muito mais usadas para o gerenciamento do que as *traps*. Então, não é surpresa perceber que investigações de WS e SNMP são relacionadas a mensagens SNMP *request/reply* e, até onde os autores deste artigo sabem, investigações específicas relacionadas a notificações não foram realizadas até hoje. Este artigo tem como objetivo apresentar uma investigação desse tema, mostrando os resultados coletados, obtidos através da observação dos tráfegos SNMP e SOAP/SMTP gerados pela implementação de nossa arquitetura de correlação de eventos distribuída.

2.2. Gerenciamento Baseado em Políticas (Policy Based Network Management - PBNM)

No contexto de gerenciamento de redes, políticas são uma maneira de controlar o comportamento da rede. Na prática, o gerenciamento baseado em políticas também tem a intenção de reduzir a complexidade e diversidade das interfaces de gerenciamento atuais (ex.: SNMP, TELNET/CLI, etc.). Isso é possível porque arquiteturas PBNM permitem ao administrador da rede definir os comportamentos da rede através de políticas que são traduzidas (pela arquitetura) para ações de configuração nos dispositivos da rede. Desta maneira, administradores não precisam se preocupar mais com a diversidade das interfaces de gerenciamento, sendo que esta variedade é tratada pela arquitetura PBNM.

A definição de políticas requer o uso de linguagens de políticas. Muitas linguagens têm sido propostas na última década (ex.: Ponder [Damianou et al., 2001] e PDL [Lobo et al., 1999]). Basicamente, os modelos de política adotados nessas linguagens permitem a definição de políticas baseadas em eventos-condições-ações ou baseadas em

condições-ações. Embora o modelo eventos-condições-ações pareça ser mais completo, o IETF (*Internet Engineering Task Force*) tem trabalhado na definição de arquiteturas, protocolos e modelos de informação para modelos condições-ações[1].

A linguagem de políticas por si só não é suficiente para a aplicação de PBNM: uma arquitetura PBNM também é necessária. Uma das arquiteturas PBNM mais aceitas é a definida pelo IETF, que é composta por quatro módulos chave: ferramenta de políticas, repositório de políticas, pontos de decisão de políticas (*policy decision point* - PDP) e pontos de aplicação de políticas (*policy enforcement point* - PEP). A ferramenta de políticas é usada pelo administrador da rede para editar as políticas armazenadas no repositório de políticas. As políticas são armazenadas para que sejam reusadas tanto na aplicação de políticas quanto na definição de uma nova política. As traduções de políticas, para ações de configuração em dispositivos, são realizadas pelos PDPs. Finalmente, os PEPs são os elementos nos dispositivos gerenciados, que efetivamente aplicam a política (ex.: disciplinas de filas e interfaces são PEPs comuns em roteadores).

Essa maneira de operação, chamada *provisioning*, é apropriada para redes que suportam *Diffserv*. A mesma arquitetura do IETF, no entanto, pode operar de acordo com o modelo *outsourcing*, apropriado para redes que suportam *IntServ* e usam o protocolo RSVP (ReSource reServation Protocol) [Braden et al., 1997] como um mecanismo de sinalização para alocação de recursos de rede. O IETF está definindo o protocolo COPS (Common Open Policy Service) [Durham et al., 2000]) para suporte ao modelo *outsourcing*, bem como o protocolo COPS-PR (COPS for Policy Provisioning) [Chan et al., 2001] para suporte ao modelo *provisioning*. Além disso, o IETF tem trabalhado em modelos de informação tais como o PCIM (Policy Core Information Model) e PCIMe (PCIM extension) [Moore, 2003].

Uma política pode ser definida de duas maneiras [Westerinen et al., 2001]:

- um objetivo definido, curso ou método de ação para guiar e determinar decisões atuais e futuras. Políticas são implementadas ou executadas em um contexto particular (como políticas definidas em uma unidade organizacional);
- políticas como um conjunto de regras para administrar, gerenciar e controlar o acesso a recursos da rede.

No contexto da arquitetura de correlação de eventos proposta, a segunda maneira será explorada. Com as políticas definidas pelo(s) administrador(es) da rede, a arquitetura é capaz de gerenciar os eventos, correlacionando-os e enviando-os para um encaminhador (gerente intermediário) pré-definido.

Uma política pode ser usada para modificar o comportamento de um sistema. Separando as políticas dos gerentes que as interpretam, pode-se modificá-las para que haja mudança do comportamento e estratégia do sistema de gerenciamento, sem ter que atualizar o código de implementação do gerente. A arquitetura proposta pode alterar o comportamento do sistema, desabilitando políticas ou trocando políticas antigas por novas, sem ter que reinicializar o sistema [Lupu and Sloman, 1999].

2.3. Correlação de Eventos e *Traps* SNMP

Uma notificação é a informação dada por um evento, relacionado a situações consideradas anormais. Notificações de eventos enviados por dispositivos de rede são essenciais

[1]Freqüentemente é dito que eventos estão implicitamente no modelo condições-ações, conceito que muitos pesquisadores discordam no entanto.

para alertar administradores sobre problemas relacionados à rede. Porém, em muitos casos, múltiplos eventos relacionados ao mesmo problema podem ser recebidos por uma estação gerente. Isto pode ocorrer, por exemplo, porque o intervalo definido para o envio de um evento seja muito pequeno. Dessa forma, o *software* de visualização do gerente pode ficar sobrecarregado mostrando muitos eventos relacionados ao mesmo problema. Outro cenário que aumenta a quantidade de eventos é quando uma única falha gera várias notificações diferentes. Por exemplo, se um roteador está inoperante e um servidor alcançado através deste roteador está funcionando normalmente, os agentes de monitoração de ambos podem enviar notificações indicando que os dois estão inoperantes ou inacessíveis. A lista abaixo mostra algumas situações comuns relacionadas a possíveis sobrecargas de eventos [de Castro and Nogueira, 1998]:

- Um agente pode enviar várias notificações relacionadas à mesma falha;
- Uma falha intermitente pode produzir um novo evento cada vez que a falha é detectada;
- Um dispositivo contendo falha pode fazer com que seja gerado um evento cada vez que o dispositivo é requisitado (ex.: uma consulta);
- Uma única falha pode ser detectada por diferentes agentes;
- Uma falha em um dispositivo pode afetar outros dispositivos da mesma rede.

Neste cenário, uma redução no número de mensagens, através da correlação de eventos, é necessária para que os processos de análise e resolução de problemas sejam realizados de forma mais rápida e eficiente. Algumas técnicas podem ser usadas para a correlação de eventos, tais como: raciocínio baseado em regras, raciocínio baseado em modelos, raciocínio baseado em casos, *codebook*, modelo de grafo de transição de estados e máquina de estados finita. Detalhes sobre estas técnicas podem ser encontradas em Zupan e Medhi [Zupan and Medhi, 2003].

Além de usar técnicas de correlação de eventos, é necessário saber o comportamento dos eventos em cada rede. De acordo com Melo et al. [Melo et al., 2000], as seguintes relações de eventos podem ser encontradas em redes IP através do SNMP.

- ***Start/stop***: os eventos são apresentados em pares, um indica quando determinada condição é válida e outro indica quando não é mais válida. Como exemplo, temos as *traps* RMON [Waldbusser et al., 2003], a mudança de estado nas *traps* SNMP e eventos de monitoração no gerente SNMP;
- ***Storm***: seqüência de vários eventos do mesmo tipo, em um determinado período de tempo;
- **Fluxo de eventos correlacionados**: quando a incidência de um evento está associada com a incidência de outro evento;
- **Fluxo de eventos independentes**: não há nenhuma relação significativa entre os eventos.

A arquitetura proposta realiza a correlação de eventos que seguem os comportamentos *storm* e fluxo de eventos correlacionados. Ambos são analisados de acordo com as políticas definidas pelo administrador. Ou seja, se um dos comportamentos ocorrer e não houver política definida para os tipos de eventos recebidos pelo encaminhador (gerente), nenhuma correlação é feita.

Como mencionado na introdução, provavelmente o mecanismo de notificação mais utilizado é o envio de *traps* SNMP. No entanto, este mecanismo possui muitas limitações

(desvantagens) conhecidas, tais como falta de confirmação de recebimento e limitação no tamanho da informação carregada pelas mensagens de notificação. Uma completa substituição das *traps* SNMP não é possível, porque isto envolveria a atualização de IOS de dispositivos ou até mesmo uma completa substituição dos dispositivos, o que não é viável na maioria das redes em operação.

Embora os WS sejam apontados como uma arquitetura revolucionária [Schönwälder et al., 2003] para o gerenciamento de redes (poderia então substituir as *traps* SNMP), acredita-se que uma solução mista, combinando SNMP com WS, poderia ser a solução mais apropriada para a obtenção das vantagens do uso de WS no gerenciamento de redes e continuar usando dispositivos compatíveis apenas com o SNMP. As próximas seções apresentam a solução proposta e um protótipo da implementação para a realização da correlação de eventos baseada em WS sem excluir os dispositivos que suportam (apenas) o SNMP. Nós utilizamos elementos SNMP desta arquitetura para avaliar os WS como uma ferramenta de notificação, cujos resultados são apresentados na seção 5.

3. Arquitetura de Correlação de Eventos Baseada em Web Services

O objetivo da arquitetura de correlação de eventos proposta é correlacionar eventos de uma forma hierárquica (Figura 1). Os eventos (*traps* SNMP) são gerados pelos dispositivos da rede e enviados aos *gateways* SNMP/WS. Em cada *gateway*, as *traps* são convertidas para mensagens SOAP/SMTP e enviadas a um encaminhador (gerente intermediário - GI). Cada GI correlaciona os eventos e possivelmente envia a correlação a um próximo GI. Este processo se repete até que os eventos correlacionados cheguem ao gerente superior(GS).

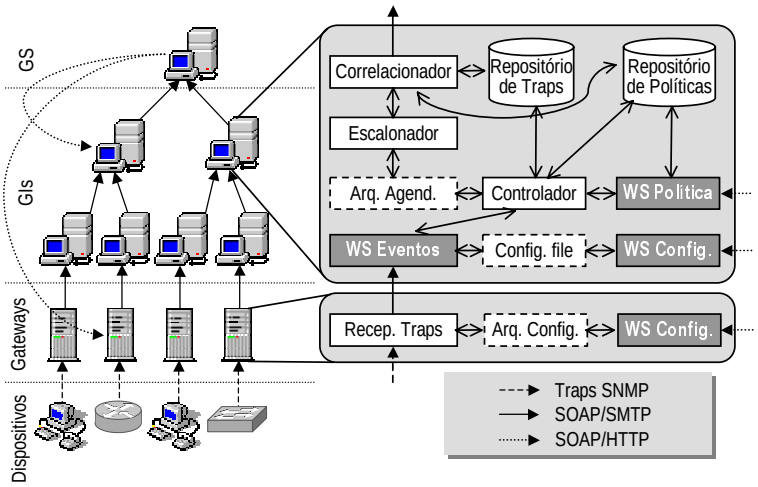

Figura 1: Árvore de Correlação e Estruturas do Gateway e GI

Além de receber os eventos correlacionados no topo da arquitetura, o GS é o elemento responsável em configurar os *gateways* e encaminhadores (GIs) através de WS de configuração, que, neste caso, utilizam mensagens SOAP/HTTP. A ligação do SOAP com HTTP foi utilizada porque ações de configuração são tipicamente realizadas com o tipo de interação *request/reply*, possibilitando uma resposta no ato. As subseções seguintes apresentam os detalhes da arquitetura com enfoque especial nas interações assíncronas, tipicamente requeridas por um suporte de notificação de eventos.

3.1. Tradução de Traps SNMP para Mensagens SOAP/SMTP

Devido a algumas limitações, *traps* devem ser tratadas dentro de domínios administrativos. *Traps* SNMP dificilmente passam por *firewalls* de borda, mas geralmente podem alcançar outras máquinas na rede local. Na arquitetura proposta, uma vez que uma *trap* é gerada na rede local, ela deve ser mapeada (traduzida) para mensagens baseadas em SOAP sobre SMTP. Depois de mapeadas, as *traps* (agora na forma de mensagens SOAP/SMTP) podem ser encaminhadas com garantia de entrega, porque SMTP utiliza TCP como protocolo de transporte. Além disso, o tráfego SOAP/SMTP pode passar por *firewalls* de borda mais facilmente, o que permite que os elementos da arquitetura proposta possam ser colocados em diversos domínios administrativos diferentes.

Para o mapeamento de *traps* para mensagens SOAP/SMTP, os dispositivos da rede e *gateways* devem estar configurados de forma apropriada. No dispositivo de rede, o agente SNMP deve estar configurado para enviar suas *traps* para um determinado *gateway*. Isso é feito através das opções de configuração no agente SNMP do dispositivo. No *gateway*, um arquivo de configuração (remotamente gerenciado graças ao elemento **WS Config**) lista os dispositivos de onde podem ser recebidas *traps* (através do elemento **Receptor de Traps**), e os GIs para onde as mensagens SOAP/SMTP devem ser encaminhadas.

3.2. Recepção e Correlação de Eventos

Cada GI possui um elemento interno **WS Eventos**, que é um servidor SMTP responsável por receber os eventos em mensagens SOAP/SMTP. O elemento **WS Eventos**, que possui alguns parâmetros de operação (apresentados na próxima seção) especificados em um arquivo de configuração local, extrai o conteúdo SOAP da mensagem SMTP e chama uma operação no elemento **Controlador**, que irá tratar a notificação recém recebida. Então, o Controlador procura, no repositório de políticas local, as políticas que poderiam tratar tal notificação. Todas políticas relacionadas à notificação são avaliadas para verificar se as ações associadas poderiam ser executadas. Se nenhuma política relacionada ao evento recebido é encontrada no repositório, então a notificação é encaminhada a um determinado GI pré-configurado (abordagem otimista) ou é descartada (abordagem pessimista). A seleção da abordagem a ser utilizada é baseada na configuração interna de cada GI.

Geralmente, uma política recém avaliada necessita ser avaliada novamente pouco tempo depois, devido ao aspecto temporal descrito na política. Desta forma, eventos temporais precisam ser notificados. Por exemplo, se uma regra de correlação de uma política determina que, se um evento X for seguido por um evento Y em menos de 30 segundos, a ação Z deve ser executada. Quando a notificação do evento X avaliar a política, deve haver uma maneira de reavaliar esta política depois de 30 segundos se nenhuma notificação de evento Y for recebida. Ou seja, deve haver um evento relógio notificando o *timeout* de 30 segundos. Esses eventos relacionados com tempo são utilizados na arquitetura proposta através do agendamento de política. O **Arquivo de Agendamento** armazena informações de agendamento, e o seu conteúdo é gerenciado pelo elemento Controlador.

Uma vez que a notificação é recebida e associada a uma política, essa notificação é armazenada no repositório de *traps*. Esse repositório possui uma tabela de *traps* e as políticas associadas (que estão sob avaliação). Essa informação é utilizada quando uma reavaliação da política é solicitada pelo elemento **Escalonador** (que monitora o Arquivo de Agendamento). A reavaliação da política é realizada pelo elemento **Correlacionador**, que verifica todas notificações que foram recebidas e estão relacionadas à política sob avaliação.

Se o conjunto de eventos torna as condições da política verdadeiras, então as ações da política são executadas.

Embora diversas diferentes ações possam ser executadas, foi utilizada neste trabalho apenas uma, que é o encaminhamento de uma correlação de eventos para um GI ou GS (indicando qual endereço de e-mail deste gerente). Isto é realizado quando o Correlacionador chama outro GI através de uma nova mensagem SOAP/SMTP. Após a execução das ações, todas *traps* armazenadas, relacionadas à política avaliada, são removidas do repositório de *traps*.

O elemento **WS Política**, enquanto armazena políticas recém recebidas no repositório de políticas local, também chama o elemento Controlador, avisando que há novas políticas disponíveis. Para cada política nova, o Controlador analisa seu conteúdo com a intenção de identificar suas condições temporais. Essa informação é então utilizada para atualizar o Arquivo de Agendamento, que armazena o período de tempo que as políticas armazenadas no repositório devem ser ativadas.

3.3. Árvores de Notificação Dinâmicas

As *traps* enviadas dos dispositivos aos *gateways* e as mensagens SOAP/SMTP encaminhadas de um GI de mais baixo nível a um de nível superior, formam uma árvore de notificação, onde os dispositivos são as folhas, os *gateways* são os nodos intermediários inferiores, GIs são os nodos intermediários acima dos *gateways* e o GS é o nodo raíz.

Tipicamente, políticas de rede não abrangem apenas aspectos de rede. Como mencionado anteriormente, elas abrangem também aspectos temporais. Isso quer dizer que políticas podem ser agendadas para serem ativadas e então avaliadas em períodos de tempo específicos. Dessa forma, é possível que os eventos correlacionados sejam encaminhados a diferentes GIs, dependendo do dia da semana e horário atuais. Um exemplo dessa abordagem é quando um gerente A envia eventos correlacionados ao gerente B das 7h01min às 8h e envia ao gerente C das 8h01min às 9h. Então, a abordagem de correlação baseada em políticas tem a capacidade de definir árvores de correlação dinâmicas, sendo que as árvores podem mudar seu formato no decorrer do tempo, devido às condições temporais.

4. Implementação do Protótipo

Foi implementado um protótipo do sistema que segue a arquitetura de correlação proposta. As subseções abaixo mostram como foi feita a implementação de cada elemento, de forma *bottom-up*.

4.1. Dispositivos de Rede e Agentes SNMP

Nós utilizamos três diferentes “geradores”de trap: um computador com sistema operacional Windows, um computador com sistema operacional Linux e um roteador Cisco. Todos tiveram seus agentes SNMP correspondentes ativados e configurados para enviar suas *traps* a determinados *gateways*. Neste caso, os *gateways*, do ponto de vista dos agentes SNMP, são gerentes responsáveis por receber e tratar as *traps* SNMP.

4.2. Implementação do Gateway

Os gateways foram implementados utilizando o *toolkit* NET-SNMP [Sourceforge.net, 2004], a linguagem PHP e a biblioteca PEAR::SOAP, em ambiente

Linux. Para receber as *traps* enviadas pelos dispositivos gerenciados, o *deamon snmptrapd* do *toolkit* NET-SNMP foi utilizado. Uma vez que a *trap* é recebida, o *deamon* realiza a tradução (mapeamento) para o formato XML, de acordo com parâmetros pré-definidos na inicialização do *deamon*. Depois de traduzida para o formato XML, a *trap* é gravada em um arquivo temporário e o script gateway.php é executado. Este *script* realiza uma chamada de operação WS utilizando SOAP sobre SMTP, passando como parâmetro a *trap* no formato XML armazenada no arquivo temporário. Na chamada de operação WS, as seguintes informações são necessárias para a realização da comunicação *gateway*/gerente:

- **E-mail destino**: como foi utilizado SMTP para transferência dos eventos após passarem pelos *gateways*, o e-mail do gerente destino, que pode ser um GI ou o GS, deve ser determinado;
- **Operação Web Service**: como diferentes operações WS podem ser solicitadas no gerente, há a necessidade de determinar qual operação é a responsável pelo tratamento da chamada feita pelo *gateway*.

O e-mail destino e a operação WS são definidas no arquivo de configuração do *gateway*. Embora eles possam ter diferentes valores em *gateways* diferentes, como configuração padrão a operação WS é sempre definida como "TrapHandler". E o e-mail destino é configurado como nome_gerente@dominio_gerente.

4.3. Implementação dos Gerentes Intermediário (GI) e Superior (GS)

O GI também foi implementado em ambiente Linux. Um servidor SMTP é responsável por receber notificações WS através de mensagens SOAP/SMTP, enquanto um servidor HTTP (Apache) é responsável por receber políticas enviadas pelo GS. No GI, ambos repositórios, o de *traps* e o de políticas, foram implementados através de bases de dados MySQL.

Três *scripts* principais implementam os elementos WS Eventos, WS Config e WS Política. O WS Config e WS Política são chamados através de mensagens SOAP/HTTP, enquanto o WS Eventos (que possui a operação WS TrapHandler) é o único chamado através mensagens SOAP/SMTP. Os demais elementos (Controlador, Escalonador e Correlacionador) também foram implementados em *scripts* PHP. O Escalonador é constantemente chamado para avaliar as políticas armazenadas. O Escalonador utiliza o *crontab* do Linux para auxiliar no agendamento de avaliação de políticas.

A implementação do GS é mais simples (possui menos elementos), pois ele é quem recebe os eventos já correlacionados e os armazena em um repositório de *traps*, para que um *software* de gerenciamento possa consultar. Para isso, são utilizados o elemento WS Eventos, já mencionado anteriormente, e uma base de dados MySQL. Há também um repositório de políticas global implementado com MySQL, que armazena todas políticas do sistema. O que há de diferente no GS são os elementos **Controlador de Políticas Global** e **Controlador de Políticas GI**. O primeiro é responsável pela inclusão, consulta, alteração e exclusão das políticas armazenadas no repositório de políticas global. O segundo é responsável pelas mesmas funções, porém nos repositórios de políticas locais (em cada GI). A Figura 2 mostra como são realizadas as comunicações entre os elementos. Pode ser visto na figura que há uma interface de *software* gerente, onde foram utilizados arquivos texto passados como parâmetro aos *scripts* PHP. Para facilitar o uso da ferramenta, uma interface gráfica está sendo estudada.

A transferência das políticas (realizada pelo elemento Controlador de Políticas GI) é feita através de um *script* PHP, que recebe como parâmetro um arquivo em formato XML.

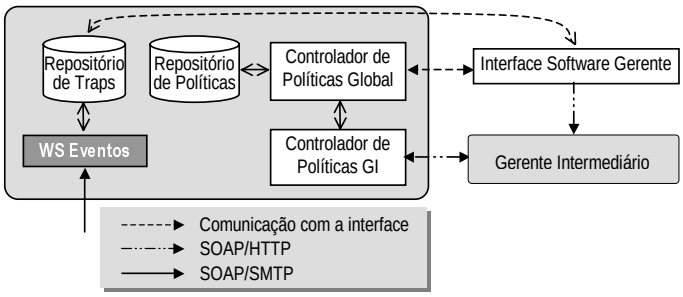

Figura 2: Gerente Superior (GS)

Atualmente, este arquivo é editado manualmente, mas como mencionado anteriormente, estamos estudando uma interface gráfica que possibilite a geração desse arquivo após a seleção dos parâmetros escolhidos na tela. Para a representação das políticas, não foi utilizada nenhuma das linguagens de políticas já existentes. Optou-se pela criação de uma linguagem com formato semelhante a linguagens de programação, tais como PHP e C. Conforme a linguagem criada, foi definido um formato XML no qual as informações da política são enviadas para que o *parser* localizado no GI possa interpretá-las e armazená-las. A Figura 3 mostra um exemplo de política utilizando a linguagem e a conversão para o formato XML.

```
Política: 1
Nome: Teste

Regra: 1
Nome: NOCDown
Condições:
if (trapType == 1 or trapType ==2) and (hostName == Hubble or hostName == Noc)
    trapType = 3
    hostName = Hubble

Regra: 2
Nome: ManagerDefinition
if (timeOfDay >= 10:00 and timeOfDay <= 14:00)
    manager = gerente@metropoa.tche.br
else
    manager = gerente@labcom.inf.ufrgs.br
```

```
<?xml version="1.0" ?>
  <Policy>
    <ID>1</ID>
    <Name>Teste</Name>
    <Rule>
      <ID>1</ID>
     <Name>NOCDown</Name>
     <Conditions>
       <Group>
         <trapType>==1</trapType>
         <operator>or</operator>
         <trapType>==2</trapType>
       </Group>
       <operator>and</operator>
       <Group> ... </Group>
     </Conditions>
     <Actions>
       <trapType>3</trapType>
       <hostName>Hubble</hostName>
     </Actions>
   </Rule> ...
```

Figura 3: Exemplo de política na linguagem criada e trecho do arquivo XML equivalente

5. Avaliação e Resultados

A arquitetura proposta apresenta dois pontos de observação: o uso de políticas (PBNM) e gerenciamento de redes baseado em notificações com Web Services. Neste momento, a preocupação maior está no segundo ponto, porque, como discutido na Introdução, não há estudos no impacto da utilização de Web Services como suporte de notificações. Conseqüentemente, nós concentramos nossas observações iniciais em verificar o volume de tráfego gerado pelas mensagens SNMP entre os dispositivos e *gateways*, e o tráfego SOAP/SMTP correspondente entre *gateways* e gerentes intermediários, bem como o retardo de transmissão para ambos os casos.

5.1. Rede e Ambiente de Teste

Para verificar o volume de tráfego gerado pelas mensagens SNMP e WS, foi realizada a monitoração *links* de uma rede de teste (Figura 4). Nesse ambiente de teste, as *traps* foram geradas por um computador denominado Hubble. Essas *traps* foram enviadas a um *gateway* denominado Labcom. Depois de mapear as *traps* SNMP, o Labcom começou a notificar, através de SOAP/SMTP, o gerente intermediário denominado Noc.

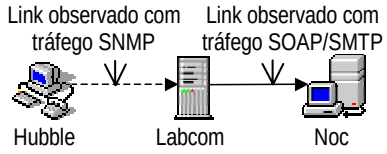

Figura 4: Rede de teste

Na máquina Hubble foi realizado o controle do envio de *traps* SNMP através de um *script bash* simples. Esse *script* inclui quantas variáveis inteiras forem passadas por um parâmetro de entrada (por linha de comando). As *traps* informavam um evento *link up*, e a lista de variáveis descreviam as interfaces hipotéticas da máquina Hubble que voltavam a operar.

5.2. Consumo de Banda SNMP x SOAP/SMTP

O tráfego SNMP inclui um cabeçalho regular e uma lista de variáveis adicionais. Inicialmente foi gerada uma *trap* sem variáveis adicionais. O tamanho da *trap* e o volume de tráfego WS gerado para transferir a mensagem SOAP/SMTP correspondente foram verificados. Depois, o número de interfaces falsas foi progressivamente aumentado até 90. A Figura 5 mostra o volume de tráfego gerado pelas mensagens SNMP e WS. Ressaltamos que a proporção do tráfego gerado pelas mensagens SOAP/SMTP pelas mensagens SNMP decresce na media que a quantidade de variáveis aumenta. Com nenhuma variável adicional o tráfego SOAP/SMTP é cerca de 52 vezes maior que o tráfego SNMP. Com 90 variáveis adicionais, a proporção cai para cerca de 5,7. Com mais variáveis, a proporção muda pouco, ou seja, a proporção não fica abaixo de 5 vezes.

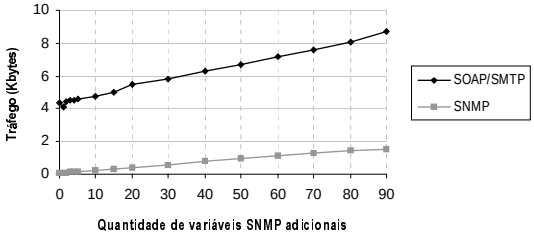

Figura 5: Tráfego SNMP x SOAP/SMTP

As mensagens SNMP geram muito menos tráfego que as mensagens SOAP/SMTP, como pode ser observado. Para mensagens com menos variáveis a diferença desse volume de tráfego é considerável. Com o aumento do número de variáveis, o volume de tráfego também aumenta. O fato importante observado é que, além do volume de tráfego de mensagens SOAP/SMTP aumentar conforme aumenta o volume de tráfego de mensagens SNMP, a diferença entre o volume de tráfego dos dois aumenta mais conforme aumenta o número de variáveis transportadas na *trap*.

A observação é crítica porque outras investigações comparando SNMP com WS [Neisse et al., 2004] [Fioreze et al., 2005] mostraram que mesmo o tráfego gerado por WS sendo maior que o tráfego gerado por SNMP, quando poucas variáveis estão presentes, o tráfego WS cresce menos que o tráfego SNMP quando mais variáveis estão presentes, levando a uma situação que eventualmente o tráfego WS para um número maior de variáveis seria menor que o tráfego SNMP associado. Mas essa observação é correta para mensagens *request/reply*, que não é o caso de nossa investigação.

5.3. Retardo de Transmissão

Utilizando o mesmo intervalo de variáveis adicionais mencionado na subseção anterior, uma medição do retardo de transmissão foi realizada, para comparar a diferença do tempo necessário para a transmissão de *traps* SNMP e mensagens SOAP/SMTP equivalentes. Para a realização desta medida, os relógios das máquinas Hubble, Labcom e Noc foram sincronizados. No momento de envio de cada *trap*, o horário da máquina Hubble foi gravado. Na máquina Labcom, o arquivo temporário utilizado para gravar a *trap* recebida, foi verificado (um dos campos da *trap* é o horário local). No *script* de geração da mensagem SOAP/SMTP, comandos adicionais foram colocados antes da chamada da API responsável por enviar a mensagem ao servidor SMTP, permitindo a gravação do horário corrente. E na máquina Noc, o mesmo procedimento foi realizado, no *script* de recepção da mensagem SOAP/SMTP, após a recepção da mensagem e antes de gravar no repositório.

O que pôde ser observado é que, independente do volume de tráfego gerado (dentro do intervalo observado), não houve variação significativa. A diferença verificada foi menor que 1 segundo. Todas *traps* SNMP foram transmitidas em menos de 1 segundo e todas mensagens SOAP/SMTP foram transmitidas em cerca de 49 segundos. Cabe ressaltar que nesses 49 segundos está incluso o tempo necessário para o estabelecimento de conexão TCP, já que o SMTP utiliza esse protocolo como transporte, e não está incluso o tempo necessário para encerramento da conexão. No momento que os dados da mensagem são lidos, o horário é gravado.

Assim como medido no volume de tráfego gerado, a diferença no tempo de transmissão de mensagens SOAP/SMTP ficou em torno de 50 vezes maior que a transmissão de *traps* SNMP. Porém, esta diferença não mudou com o aumento do número de variáveis adicionais, ao contrário do volume de tráfego. Este é um fator importante, para ajudar na decisão do uso de uma dessas duas tecnologias, quando o tempo é considerado importante no sistema em questão.

6. Conclusões e Trabalhos Futuros

Um aspecto importante do uso de WS para o gerenciamento de redes é que eles são fáceis de implementar, de manter e são bons para a interconexão de diferentes sistemas. Outro importante aspecto é a garantia de entrega das mensagens de notificação, tornando o sistema mais confiável. SOAP sobre HTTP para o gerenciamento de redes tem sido investigado ultimamente, e pode ser considerado relativamente bem conhecido. Por outro lado, pouco é conhecido sobre o gerenciamento de redes utilizando SOAP sobre SMTP. Há duas razões principais para esse fato: o suporte para SOAP sobre HTTP é bem mais utilizado que SOAP sobre SMTP, e SMTP é apropriado para mensagens assíncronas como as *traps*, mas as *traps* são bem menos utilizadas no SNMP do que outras mensagens síncronas tais como *GetRequest* and *GetResponse*.

Neste artigo, foi apresentada uma arquitetura distribuída para correlação de eventos, onde *traps* SNMP são mapeadas para mensagens assíncronas SOAP/SMTP. Tal mapeamento é realizado por *gateways* que encaminham notificações aos GIs, responsáveis pela correlação dos eventos baseada em políticas (definidas pelo operador da rede). Essa arquitetura foi implementada, e neste artigo a atenção foi focada na avaliação do volume de tráfego gerado e o retardo de resposta pelos tráfegos SNMP e SOAP/SMTP, quando feitas notificações de um número crescente de falsas interfaces que deixaram de funcionar.

O volume de tráfego gerado pelas mensagens WS é muito maior que o volume de tráfego gerado pelas mensagens SNMP. Essa é uma importante observação porque investigações anteriores focaram em mensagens SNMP síncronas, mostrando que em algumas situações o tráfego WS pode ser menor que o tráfego SNMP. Neste artigo, foi mostrado, contudo, que não somente o tráfego WS é maior que o tráfego SNMP, mas também que a diferença entre os dois cresce cada vez mais, se mais variáveis estão presentes. No entanto, a proporção do tráfego SOAP/SMTP sobre o tráfego SNMP diminui conforme o número de variáveis aumenta.

Esse fato pode levar à conclusão que notificações baseadas em WS não valem a pena. Isso é verdade se considerado apenas o aspecto de volume de tráfego gerado. Contudo, WS possuem facilidades de integração ausentes no SNMP. E ainda, notificações baseadas em WS podem cruzar diferentes domínios administrativos porque WS são baseados em protocolos Web, enquanto o tráfego SNMP é geralmente restrito a redes locais. Por outro lado, esses fatos não ajudam se o tráfego de notificações imposto pelos WS tornar a rede congestionada. Na opinião dos autores, há aspectos que devem ser avaliados na escolha de uma das tecnologias para a notificação de eventos, tais como volume de tráfego gerado, retardo de transmissão, facilidade de uso e facilidade/possibilidade de integração.

Trabalhos futuros nessa direção incluem mais investigações sobre como a correlação de eventos pode reduzir o volume de tráfego associado aos WS. Enquanto pensa-se que os WS são requeridos para integração de notificações inter-domínio, também acredita-se que o volume de tráfego gerado é um aspecto importante. Nesse cenário, reduzindo o volume de tráfego e ainda utilizando uma tecnologia para integração como os WS, poderia-se ter uma ferramenta mais apropriada para notificações.

Referências

Braden, R., Zhang, L., Berson, S., Herzog, S., and Jamin, S. (1997). Resource reservation protocol (rsvp) – version 1 functional specification. Request for Comments: 2205. IETF.

Case, J., Fedor, M., Schoffstall, M., and Davin, J. (1990). A simple network management protocol (snmp). IETF RFC 1157.

Chan, K., Seligson, J., Durham, D., Gai, S., McCloghrie, K., Herzog, S., Reichmeyer, F., Yavatkar, R., and Smith, A. (2001). *COPS Usage for Policy Provisioning (COPS-PR): RFC 3084*. IETF.

Consortium, W. W. W. (2004). Web services activity. http://www.w3c.org/2002/ws/.

Damianou, N., Dulay, N., Lupu, E., and Sloman, M. (2001). The ponder policy specification language. *Lecture Notes in Computer Science*, 1995:18–22. in Workshop on Policies for Distributed Systems and Networks (Policy 2001).

de Castro, T. and Nogueira, J. (1998). An alarm correlation system for sdh networks. In *Proceedings Telecommunications Symposium - SBT/IEEE International ITS '98*. Pages 492 - 497, Vol. 2.

Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R., and Sastry, A. (2000). *The COPS (Common Open Policy Service) Protocol: RFC 2748*. 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 (a ser publicado)*.

Lobo, J., Bhatia, R., and Naqvi, S. (1999). A policy description language. In *AAAI/IAAI*, pages 291–298.

Lupu, E. C. and Sloman, M. (1999). Conflicts in policy-based distributed systems management. *IEEE Transactions on Software Engineering*, 25(6):852–869.

Melo, E., Vieira, E., and Sari, S. (2000). Tratamento de eventos. *Boletim bimestral sobre tecnologia de redes*, 4(4). Disponível em <http://www.rnp.br/newsgen/0007/art10.html>. Acesso em 08 Jun.

Moore, B. (2003). Policy core information model (pcim) extensions. Request for Comments: 3460, Updates RFC 3060. IETF.

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 sevices gateways. In *IEEE/IFIP Network Operations and Management Symposium, NOMS*, pages 715–728.

Pavlou, G., Flegkas, P., Gouveris, S., and Liotta, A. (2004). On management technologies and the potential of web services. *IEEE Communications, special issue on XML-based Management of Networks and Services*, 42(7):58–66.

Roy, J. and Ramanujan, A. (2001). Understanding web services. *IEEE Computer Society - ITPro*, pages 69–73.

Schönwälder, J., Pras, A., and Martin-Flatin, J. P. (2003). On the future of internet management technologies. *IEEE Communications Magazine, Vol. 41, No. 10*, pages 90–97.

Sourceforge.net (2004). The net-snmp project home page. http://net-snmp.sourceforge.net/.

v. Sloten, J., Pras, A., and v. Sinderen, M. J. (2004). On the standardisation of web service management operations. *EUNICE 2004 - Proceedings of the 10th Open European Summer School and IFIP WG6.3 Workshop*, pages 143–150.

Waldbusser, S., Kalbfleisch, C., and Romascanu, D. (2003). *Introduction to the Remote Monitoring (RMON) Family of MIB Modules: RFC 3577*. IETF.

Westerinen, A., Schnizlein, J., Strassner, J., Scherling, M., Quinn, B., Herzog, S., Huynh, A., Carlson, M., Perry, J., and Waldbusser, S. (2001). Terminology for policy-based management, request for comments (rfc) 3198. The Internet Engineering Task Force (IETF).

Zupan, J. and Medhi, D. (2003). An alarm management approach in the management of multi-layered networks. In *Proceedings 3rd IEEE Workshop on IP Operations and Management (IPOM)*. Pages 77 - 84.