Minimização da Intervenção Humana para Detectar e Solucionar Anomalias Rede de Computadores

Alexandre Amaral[1], Ana Paula Malheiro[2]

[1]Instituto Federal de Santa Catarina (IFSC)
CEP 89254-430 – Jaraguá do Sul – SC – Brasil

[2]Instituto Federal Catarinense (IFC)
Caixa Postal 2016 – Camboriú – SC – Brasil

alexandre.amaral@ifsc.edu.br, anapaula@ifc-camboriu.edu.br

Abstract. *This paper proposes a system to detect and apply corrective actions when anomalous events occur in the network. Management goals are defined through the metrics and the self-healing property of the autonomic computing is used, empowering the system to perform corrective actions without human intervention. NEMES (Network Metric Specification) a domain-specific language was developed to build the metrics. The system uses IP flows to reduce the volume of data to be processed, allowing its use in large-scale networks. Tests performed in a real environment have shown the effectiveness and potential of the proposed system to assist in the network management.*

Resumo. *Neste trabalho é proposto um sistema para detectar e aplicar ações corretivas na ocorrência de eventos anômalos na rede. Os objetivos da gerência são definidos através de métricas e a propriedade de autorreparo da computação autonômica é utilizada para que o sistema execute ações corretivas sem a intervenção humana. NEMES (Network Metric Specification), uma linguagem de domínio específico foi desenvolvida para a escrita das métricas. O sistema utiliza fluxos IP que reduz o volume de dados a serem processados, permitindo sua utilização em redes de grande escala. Testes realizados em um ambiente real demonstraram a eficácia e o potencial do sistema proposto para auxiliar no gerenciamento de rede.*

1. Introdução

Esforços têm sido realizados pela comunidade cientifica a fim de desenvolver mecanismos que auxiliem no monitoramento e na segurança da rede. Algumas soluções do tipo NIDS (*Network-based Intrusion Detection System*) têm sido propostas. Porém, [Bhuyan *et al.* 2014] apresentaram recentemente diversos fatores que limitam a aplicação desses mecanismos para funcionamento em tempo real em redes de grande escala. A inspeção de todos os pacotes transmitidos pela rede requer um alto processamento e espaço de armazenamento. Os alarmes emitidos na ocorrência de um evento anômalo requer uma análise manual do administrador de rede para que o problema seja reparado. A aplicação dos NIDS em um escopo mais amplo de gerenciamento não tem sido endereçada. Isto inclui o monitoramento de serviços e usuários, identificação de pontos de gargalo na rede, além de não fornecerem um meio para definição e monitoramento de KPIs (*Key Performance Indicator*) específicos, como *uptime*/*downtime* de um recurso de rede.

Nesse trabalho é apresentado um sistema para detectar eventos anômalos na rede e aplicar ações corretivas, com funcionamento em tempo real. Os objetivos da gerência são definidos através de métricas, e o sistema aplica a propriedade de autorreparo da computação autonômica [Magalhães e Silva 2013], a fim de que as ações corretivas e/ou preventivas sejam executadas sem a intervenção humana. NEMES (*Network Metric Specification*), uma linguagem de domínio específico foi desenvolvida para a escrita das métricas. Diferentemente das propostas que utilizam como fonte de dados os pacotes IP, o sistema proposto utiliza os fluxos IP. Esta escolha visa reduzir o volume de dados a serem processados e a aplicação do sistema em redes de grande escala. Testes realizados em um ambiente real demonstraram a eficácia e o potencial desse sistema para auxiliar no gerenciamento de rede.

2. Sistema Proposto

A Figura 1 apresenta os principais componentes do sistema. Os fluxos IP coletados dos dispositivos ativos na rede são processados pelo *FE (Feature extractor)*. O *FE* obtêm as propriedades dos fluxos (*e.g.*, endereço IP de origem, número de pacotes) enviando os para o componente *ME* (*Metric processor)*. Utilizando as métricas lidas da *KB* (*Knowledge base*), as propriedades dos fluxos são analisadas pelo *ME,* com o objetivo de identificar os fluxos anômalos ou indesejáveis. As métricas contêm as condições das propriedades dos fluxos considerados anômalos e as ações que devem ser executadas caso eles ocorram na rede. Quando um fluxo é considerado anômalo a ação especificada na métrica é executada pelo componente *AP* (*Action processor*), e o resultado de sua execução é armazenado na *KB.*

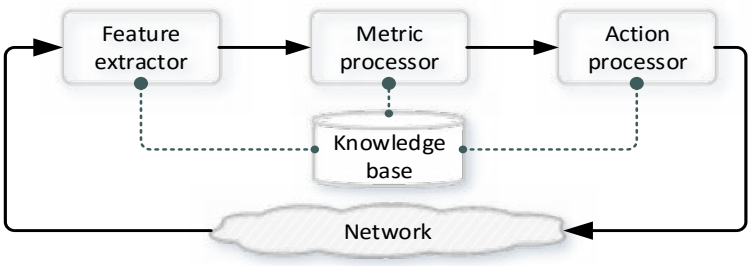

Figura 1. Principais componentes do sistema.

Neste trabalho, uma métrica é uma condição definida pelo administrador que caso ela ocorra na rede uma ação é disparada. A Figura 2 apresenta uma métrica para detecção de um ataque *Ping of Death*. Este ataque consiste no envio de fluxos ICMP com pacotes maiores do que 65535 bytes. Tais características são definidas na *condition* da métrica. Na ocorrência desse ataque, a métrica especifica duas ações a serem realizadas: *execute* e *notify*. O método *execute* aplica uma ação corretiva e/ou preventiva. No exemplo, um *script* será executado com o propósito de bloquear o ataque. Simultaneamente, uma notificação será enviada para o administrador de rede através do método *notify*. O meio de comunicação utilizado no exemplo é o e-mail. Entretanto, o sistema está sendo desenvolvido como uma interface que permita a utilização de outros meios, tais como SMS (*Short Message Service*) e *log*.

```
{
  "metric": {
     "id": "030511",
     "status": "active",
     "description": "Detecção de ataques Ping of Death",
     "condition": {
        "and":[
           {"gt": {"sizepackets": 65535}},
           {"eq": {"proto": "ICMP"}}
         ]
     },
     "action":{
         "methods":[
            {
               "name":"execute",
               "process":"/action/scripts/blockPingOfDeathAttack.sh"
            },
            {
               "name":"notify",
               "mean":"email",
               "msg":"Ping of Death detectado. Fonte %srcip% e o(s) destino(s) %dstip%."
            }
         ]
      }
   }
}
```

Figura 2. Exemplo de uma métrica para detectar um ataque *Ping of Death*.

Uma linguagem de domínio específico, denominada NEMES (*Network Metric Specification*) foi desenvolvida para a criação das métricas. NEMES utiliza a notação JSON como mostra a Figura 2. A escolha do JSON dentre outras linguagens (*e.g.*, XML), se por algumas razões. O JSON é mais leve, mais fácil para leitura e escrita por humanos e por estar sendo amplamente utilizado nos últimos anos [Yang 2012]. Antes de serem disponibilizadas na base de conhecimento, as métricas escritas pelo administrador de rede são processadas pelo compilador NEMES, desenvolvido com o propósito de realizar a análise léxica, análise sintática e análise semântica.

2.1. Aplicação do autorreparo

Quando a condição de uma métrica é atendida o componente *AP (Action Processor)* é acionado. Através deste componente, o sistema proposto implementa o método *execute* com o objetivo de aplicar a propriedade de autorreparo da computação autonômica. No contexto do gerenciamento de redes, uma gama de ações pode ser definida pelo administrador para fins de autorreparo. O bloqueio de acesso e a finalização de uma sessão de usuário pelo abuso dos recursos de um servidor são procedimentos que podem ser especificados, a fim de que os serviços prestados pela rede continuem operando sem danos.

3. Estudo de Caso

Testes em um ambiente real de rede foram realizados com o propósito de validar o sistema proposto. Para este fim, um *testbed* foi montado, como ilustra a Figura 3. Fluxos IP são coletados do gateway/firewall através do protocolo NetFlow v9 a cada cinco minutos. O sistema foi desenvolvido com a linguagem Java. Para a base de conhecimento foi utilizada o banco de dados Neo4J [Neo4J 2016].

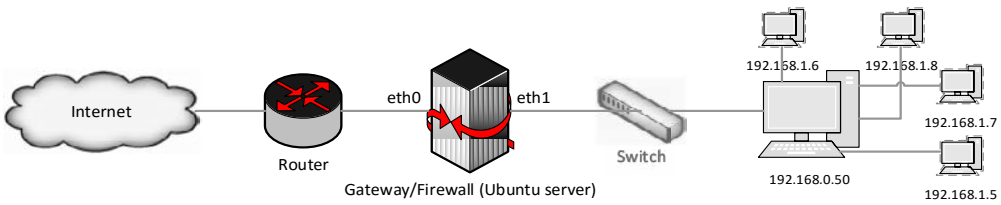

Figura 3. Testbed para realização dos testes.

Nos experimentos, as métricas foram criadas para a detecção de *Land attack*, *Smurf attack*, detecção de acesso não permitido a diversos sites proibidos pela instituição, incluindo redes sociais e de conteúdo adulto. Para geração dos ataques *Land attack* e *Smurf attack* foi utilizada a ferramenta Hayane [Hyenae 2016] lançados a partir das máquinas virtuais hospedadas na máquina `192.168.0.50`. Para aplicação das ações de autorreparo foi utilizado o firewall Iptables instalado na máquina Gateway/Firewall.

Os testes realizados demonstraram a eficácia do sistema para detectar e aplicar as correções especificadas nas métricas. As ações testadas correspondem ao método *execute* e *notify*. Em todos os testes foram utilizados *scripts* para bloquear os fluxos indesejados via firewall. O método *notify* foi testado utilizando o e-mail e o armazenamento textual do resultado da aplicação da ação em arquivos de *log*.

4. Conclusão

Nesse trabalho foi apresentado um sistema para detecção de atividades anômalas na rede. Através de uma linguagem de domínio específica denominada NEMES, métricas podem ser escritas para que o sistema identifique atividades que vão além dos ataques e ações maliciosas endereçadas pelas soluções atuais. A utilização da propriedade de autorreparo da computação autonômica permite que o sistema monitore, detecte as ações especificadas pelas métricas e execute ações para solucioná-las. Os testes demonstraram a capacidade do sistema de tomar ações sem a intervenção humana, auxiliando o administrador na árdua tarefa do gerenciamento de rede. Como trabalhos futuros, desejamos explorar outras propriedades da computação autonômica, realizar mais testes e utilizar o sistema de forma distribuída.

5 Referências

Bhuyan, M.H. Bhattacharyya, D.K. e Kalita, J.K. (2014) “Network Anomaly Detection: Methods, Systems and Tools”. Communications Surveys & Tutorials, IEEE , vol.16, no.1, p. 303-336, 2014.

Hyenae. Disponível em: http://sourceforge.net/projects/hyenae/

Magalhães J. P. e Silva L. M., "Self-healing Performance Anomalies in Web-based Applications," Network Computing and Applications (NCA), 2013 12th IEEE International Symposium on, Cambridge, MA, pp. 81-88, 2013.

Neo4j. “The World's Leading Graph Database”. Disponível em: http://www.neo4j.org/

Yang Y. “Impact data-exchange based on XML”, Computer Science & Education (ICCSE), 7th International Conference. p.1147-1149, 2012.