UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
GERÊNCIA E APLICAÇÕES EM REDES
RELATÓRIO DO TRABALHO FINAL

LAURA SOARES - 00245174
PIETRA FREITAS - 00242285

INTRODUÇÃO

Gerenciamento de redes como microsserviço

Atualmente, arquiteturas de gerenciamento de redes costumam seguir modelos estáticos em termos de topologia e protocolos utilizados. Porém, a infraestrutura das redes modernas permite a implementação de modelos muito mais dinâmicos, como Virtualização de Funções de Redes (NFV), infraestruturas baseadas em nuvem, entre outros. O objetivo principal deste trabalho é propor uma aplicação e arquitetura de gerenciamento que acompanhe essa dinâmica e que seja capaz tanto de gerenciar quanto de fazer uso desses novos tipos de serviços em redes.

Estratégia de implementação: usando Docker

Docker é o software utilizado neste trabalho para a implementação de micro-serviços. Diferente dos serviços de virtualização tradicionais, que implementam um sistema operacional completo e isolado, containers Docker possuem recursos isolados mas que utilizam bibliotecas em comum com o host onde está sendo executado. Isso possibilita o "empacotamento" de uma aplicação dentro de um container, o que agiliza o processo de portabilidade. No contexto deste trabalho, se propõe a implementação de containers contendo tanto a aplicação a ser gerenciada quando a arquitetura de gerenciamento sendo utilizada.

Sobre a arquitetura de gerenciamento utilizada: SNMP

Simple Network Management Protocol é um protocolo de gerenciamento utilizado para coletar e organizar informações sobre dispositivos em rede. Essas informações podem ser alteradas pela estação de gerenciamento (gerente) para modificar o comportamento dos dispositivos gerenciados (agentes). O protocolo é não

orientado à conexão e foi projetado para ser o mais simples possível, causando um overhead mínimo ao tráfego da rede.

Docker images utilizadas

Para a aplicação agente do SNMP, foi utilizada a imagem snmpd disponível em [1], escolhida dentre outras implementações por sua popularidade e facilidade de uso. Para a aplicação do serviço a ser monitorado, utilizou-se um container docker para o Firefox, disponível em [2]. A imagem permite o ajuste de vários parâmetros, e é acessada através de qualquer browser utilizando o IP do host e a porta do serviço, sem necessidade de instalação.

ESTRATÉGIAS DE IMPLEMENTAÇÃO

Aplicações cliente

Para fazer a configuração dos clientes a serem monitorados, foi utilizado a funcionalidade de swarm disponibilizada pelo Docker. Para isso, o seguinte arquivo docker-compose foi utilizado:

```
1  version: "3.3"
2
3  services:
4      snmpd:
5          image: polinux/snmpd
6          ports:
7          - "161:161/udp"
8          deploy:
9              replicas: 2
10             restart_policy:
11                 condition: none
12
13     firefox:
14         image: jlesage/firefox
15         shm_size: '64m'
16         ports:
17         - "5800:5800"
18         volumes:
19         - "/docker/appdata/firefox:/config:rw"
20         deploy:
21             replicas: 2
22             restart_policy:
23                 condition: none
```

Para garantir o funcionamento da aplicação conforme a especificação deste trabalho, foi necessário o uso de diretivas de *deployment*. ‘replicas’ é utilizado para

garantir que a quantidade de instâncias do serviços (snmpd e firefox) é igual à quantidade de nodos no swarm. O Docker por padrão tenta alocar uma réplica por nodo no swarm. Caso isso não seja possível uma segunda tentativa é feita em outro nodo, e, dessa maneira, é possível que um mesmo nodo acabe com duas réplicas do mesmo serviço. Para que isso não aconteça, foi adicionada a diretiva 'restart_policy' com o valor de *none*. Assim, caso não seja possível colocar uma réplica do serviço em um nodo específico do swarm, o deployment é abortado.

Idealmente, seriam utilizadas diretivas de 'placement', onde são especificadas 'constraints' e 'preferences' sobre o nodo em que cada container de serviço seria colocado.

Servidor e aplicação de gerenciamento

A aplicação do gerente foi simplificada ao máximo para facilitar a execução do trabalho. A cada cinco segundos, é feita uma chamada do servidor para o cliente, através de snmp walk, acessando a MIB SysUpTime.

Na aplicação original era utilizada a MIB IfInOctets, para medição do tráfego de entrada do host gerenciado. Porém, essa MIB é privada, e só pode ser acessada de outros IPs se configurado corretamente no arquivo snmpd.conf do agente. Para que fosse possível executar toda a aplicação através de um script automatizado, decidiu-se usar a MIB SysUpTime, cujo acesso é público.

Scripts de automação

O shell script utilizado é interativo e realiza desde a criação do swarm à execução da aplicação de medição desenvolvida.

ANÁLISE DOS RESULTADOS OBTIDOS

O estado final da aplicação desenvolvida corresponde aos requisitos da definição do trabalho, porém apenas em cenários muito simplificados. Como está, o container com o serviço snmpd retorna dados de tráfego de todo o host de origem, e não apenas do container com a aplicação firefox. Portanto, para medição correta, apenas esses dois containers devem estar sendo executados no host.

Várias abordagens foram testadas sem sucesso para uma possível correção. Idealmente, seria declarado no docker-compose do swarm o serviço firefox com acesso somente a uma rede interna, enquanto o serviço snmpd teria acesso tanto à essa rede interna quanto à rede default do swarm em overlay. Para que a comunicação entre

os dois containers fosse feita corretamente, a rede interna estaria em modo bridge para tráfego interno em um mesmo host. Porém, não é possível fazer o deploy de uma compose file com uma rede declarada em modo bridge, já que em um swarm não há garantia que dois containers estarão em um mesmo host. Alternativas para superação desse problema não foram encontradas.

Sobre a aplicação do gerente, a ideia inicial era fazer a execução dentro de um container próprio, dentro da rede snmpd_default utilizada pelo swarm. Com esse objetivo a aplicação original foi simplificada, pois várias bibliotecas utilizadas (como a tkinter e a matplotlib) não puderam ser portadas para dentro do container. Porém, adicionar um container à rede do swarm requer acesso privilegiado, e não foram encontrados meios de executar isso automaticamente com um script.

ANÁLISE CRÍTICA DA APLICABILIDADE DO TRABALHO

Atualmente o protocolo SNMP por si só tem dado lugar a serviços mais modernos e melhor adaptados para o gerenciamento de redes. Esses novos protocolos (como Netconf e YANG) resolvem problemas como a falta de segurança e as interfaces pouco intuitivas da arquitetura SNMP.

Especificamente dentro de um ambiente de containers, existem APIs capazes de coletar informações de desempenho de cada serviço sem ter que fazer novas instalações, e sem se limitar a dados de rede. Um exemplo é o software de código aberto Prometheus [3], utilizado para monitoramento de outros containers.

Em adição a isso, durante a elaboração deste trabalho, uma grande quantidade de mão-de-obra foi necessária para configurar um container de serviço ao container com o snmpd. Escalar isso para múltiplos serviços não apresenta um custo benefício favorável se comparado às outras soluções disponíveis. No caso de múltiplos containers de serviço serem de fato implementados, o tráfego de gerenciamento necessário (apesar de mínimo segundo a implementação original do protocolo SNMP) tende a crescer. Junto com o crescimento desse tráfego, cresce também o problema já conhecido da dificuldade de fazer o manejo desses dados, o que aumenta a complexidade da aplicação de gerenciamento.

Acredita-se, portanto, que apesar de ter fornecido valiosas experiências de aprendizado e um estudo mais profundo dos conceitos abordados na disciplina, a aplicação desenvolvida tem baixa aplicabilidade em cenários reais.

CONCLUSÃO

Neste trabalho foi proposta uma aplicação para gerenciamento de redes utilizado o protocolo de gerenciamento SNMP, junto com um ambiente de containers implementados utilizando Docker. Por se tratar de um estudo acadêmico, e também de uma proposta inovadora, muito pouco pode ser encontrado na literatura existente que facilitasse a elaboração da proposta. Apesar disso, o resultado final obtido atende os requisitos da especificação do trabalho para cenários super simplificados.

REFERÊNCIAS

[1] polinux/snmpd - Docker Hub

[2] jlesage/firefox - Docker Hub

[3] Prometheus - Monitoring system & time series database, Collect Docker metrics with Prometheus | Docker Documentation

Docker (software) – Wikipédia, a enciclopédia livre

Simple Network Management Protocol - Wikipedia

Enterprise Container Platform | Docker

http://cric.grenoble.cnrs.fr/Administrateurs/Outils/MIBS/?oid=1.3.6.1.2.1.2.2.1.10

Aula 05 - Arquitetura de Gerenciamento da Internet