UDP-FLEX

Monitor, Direcionador e Multiplicador de Fluxo de pacotes UDP integrado à plataforma V-PRISM

Nilson Luís Damasceno

Sistemas Multimídia – 2020.1 - Trabalho Final

(relatório preliminar - 2)

Introdução

O fluxo de pacotes é uma característica essencial em qualquer rede de computadores baseada no protocolo IP. A plataforma V-PRISM, baseada no protocolo IP, permite a criação de comunicação entre sensores multimídia virtuais (VMS) onde cada dispositivo recebe pacotes IP da rede. Os pacotes, uma vez recebidos por um VMS, podem ser decodificados, transformados e/ou armazenados. Além disso, os VMS podem ser origem de pacotes, sejam eles gerados pelo próprio VMS ou simplesmente resultantes da transformação dos pacotes recebidos.

Numa rede física tradicional, os dispositivos que integram uma rede não costumam se conectar diretamente. Normalmente, existem equipamentos dedicados para realizar a interconexão desses dispositivos, tais como *switches* e roteadores. Por conta disso, esses equipamentos de interconexão acabam tendo acesso a todos os pacotes que trafegam na rede. Isso os capacita a realizar o monitoramento direto do volume de tráfego, como total de pacotes e bytes transmitidos, taxas de transmissão em bytes ou pacotes por segundo, como também realizar o monitoramento indireto de falhas na rede ou nos dispositivos através da ausência total de um fluxo esperado ou a redução do seu volume.

A plataforma V-PRISM não prevê a criação de equipamentos de interconexão, uma vez que um dos objetivos dessa plataforma é permitir a criação de cadeias de dispositivos em sequência, onde cada um desses pode aplicar alguma operação sobre os elementos presentes no fluxo de pacotes que recebe, gerando um fluxo de saída com os elementos transformados, ou gerando um novo fluxo contendo elementos obtidos através dessa operação.

Uma característica da plataforma que é fundamental para este trabalho é o fato de que toda a comunicação entre dispositivos VMS se dá através do protocolo UDP, possibilitando a criação de um componente de monitoramento especializado nesse protocolo de transporte.

Objetivo e Requisitos

O objetivo geral deste trabalho é projetar e desenvolver um componente de software que atenda aos requisitos a seguir.

O componente deve ser capaz de:

1. ser executado como um programa regular sobre um sistema operacional;
2. ser executado como VMS do V-PRISM (um container Docker);
3. ser executado como *daemon* auxiliar dentro de um container VMS;
4. receber fluxos de pacotes UDP em uma porta de entrada;
5. enviar cada pacote recebido para **N** portas de saída UDP, onde $N \in [0, 32]$;
6. enviar os pacotes UDP recebidos com o menor atraso possível;
7. utilizar recursos computacionais mínimos durante a execução;
8. contabilizar o número de bytes e de pacotes recebidos;
9. enviar as informações contabilizadas para um coletor externo;
10. aceitar a especificação das portas de saída através de linha de comando ao iniciar;
11. permitir o acréscimo e remoção de portas de saída durante a operação mediante comandos fornecidos durante a execução;
12. especificamente receber comandos para o acréscimo e remoção de portas de saída através do mecanismo “*subscribe*” utilizando o protocolo MQTT.

Descrição do UDP-Flex

O software UDP-FLEX é um programa versátil, cujo código executável ocupa apenas 59Kbytes quando compilado para a plataforma Linux 64 bits. A Figura 1 apresenta os diversos parâmetros que possibilitam grande flexibilidade no seu uso. O UDP-FLEX já está integrado ao framework V-PRISM, tanto como VMS quanto como *daemon* em um container VMS. Durante o processo de integração com foram realizados testes usando ferramentas do Linux ("ls", "top" e "iperf") que possibilitaram a verificação da baixa utilização de recursos e apesar de um elevado *throughput* utilizando a interface *localhost*: 100Mbps sem perda de pacotes (pacotes de 1K) e 1 Gps com perda de até 10 pacotes a cada 1000 pacotes.

```
-----------------------------------
UDP-Flex for V-PRISM      version 0.9
-----------------------------------
Usage: ./udp_flex [ options ] [remote_host [ remote_port ]]
remote_host               - destination host
remote_port               - destination udp port - Default = 5000
Forward mode options:
  -lp local_port          - local udp port. Default = 5000
  -id ID_String           - used to identify this instance. Default = UPX_001
  -ts time silence        - maximum time silence seconds (heartbeat). Default = 60
  ====== Secondary destination
  -sh secondary_host      - ip of secondary remote host.
  -sp secondary_port      - udp port of secondary remote host  - Default = 5000
  ====== MQTT server
  -mh mqtt_host           - mqtt host
  -mp mqtt_port           - mqtt port  - Default = 1883
  -mt mqtt_topic          - mqtt topic
  ====== Collector instance
  -ch collector_host      - ip of remote collector.
  -cp collector_port      - udp port of remote collector. Default = 6001
Collect mode options:
  -R REST_URL             - url of V-PRISM REST database collector.
  -P collect_port         - listen port in collect mode. Default=6001
Common options:
  -h                      - prints this help
  -D                      - debug. print on stderr
```

Figura 1- Opções para ativação do UDP-Flex via linha de comando

Modos de execução: forwarding e collecting

Para garantir uma performance mais elevada, o componente UDP-Flex deve ser executado em um dos seguintes modos: **modo *forwarding*** e **modo *collecting***. A Figura 2 ilustra como essas diferentes instâncias se relacionam.

A existência de dois modos distintos de execução se justifica como forma de evitar que o componente responsável pela transmissão dos pacotes tenha de consumir tempo ou recursos para enviar dados de monitoramento através de um protocolo de coleta complexo, que exija, por exemplo, o uso de TCP e, consequentemente, o uso de temporizadores, janelas de transmissão etc. Assim, as instâncias executando no modo *forwarding* consomem a maioria dos ciclos da CPU realizando apenas a tarefa para qual foram projetados.

Já, a(s) instância(s) executando em modo *collecting*, recebem as informações de monitoramento das instâncias em modo *forwarding*, eventualmente gerando um arquivo de *log* ou,

eventualmente, encaminhando as informações de monitoramento para coletores externos. Para integração com o V-PRISM, instâncias em modo *collecting* podem enviar as informações recebidas para um serviço WEB-REST oferecido pela plataforma.

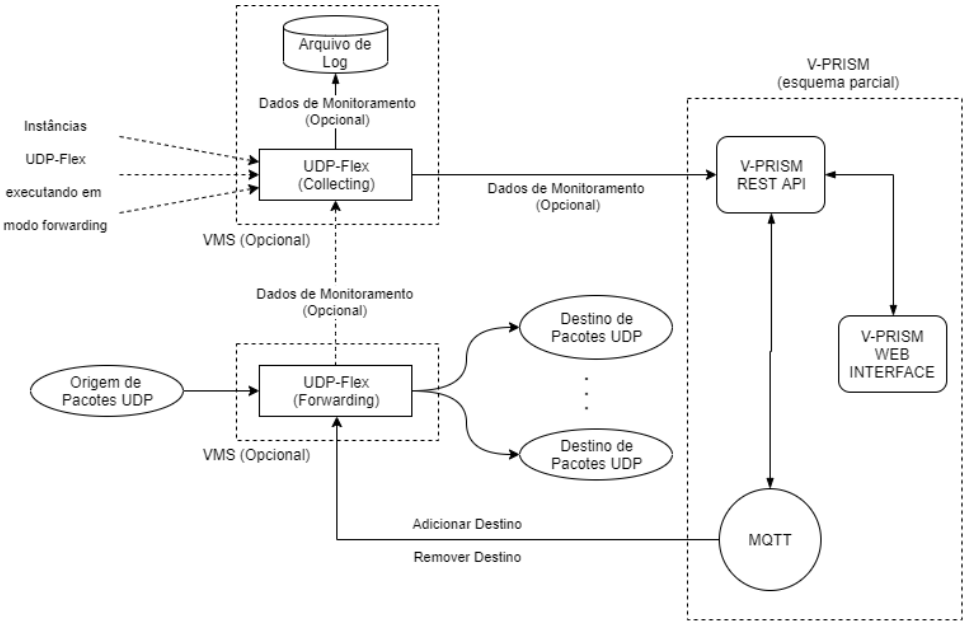

Figura 2 - Diagrama esquemático para utilização do UDP-Flex

Executando no modo *forwarding*, o UDP-Flex:

- implementa a função de transporte entre os pacotes recebidos na entrada e todas a(s) saída(s) programadas;
- opcionalmente, contabiliza o tráfego recebido para depois enviar, usando UDP, os dados contabilizados para uma instância do próprio componente que deve estar rodando em modo *collecting*;
- possui um código "ID" que serve para identificá-lo univocamente no mundo. Isso permite que um único componente em modo *collecting* possa concentrar toda a coleta, recebendo dados coletados por todos os outros componentes identificados rodando em modo forwarding na mesma rede e enviando esses dados recebidos para um coletor externo.

Heartbeat

Quando o UDP-Flex está operando em modo *forwarding*, ele envia um pacote com as estatísticas coletadas sempre que houve algum tráfego recebido neste período. Porém, se não houver tráfego recebido, não existiria nenhuma informação de tráfego a ser enviada. No entanto, do ponto de vista da plataforma, essa ausência de informações sobre tráfego em um dispositivo também pode significar que houve uma falha de comunicação em alguma parte da infraestrutura. Para evitar essa ambiguidade, foi implementado o mecanismo ***heartbeat***, que faz cada instância do UDP-Flex executando em modo *forwarding* enviar um pacote de monitoramento pelo menos a cada X segundos (parâmetro X configurável), tendo recebido tráfego na entrada ou não. Por conta deste mecanismo, a plataforma pode inferir que a ausência

de informações de monitoramento oriundas de algum dispositivo, realmente significa que houve alguma falha na infraestrutura.

Disparo de Gradiente

O UDP-Flex também implementa um mecanismo adicional para reduzir o tempo necessário para que o painel de controle da infraestrutura detecte alguma variação brusca no fluxo de pacotes recebidos por um componente. Tipicamente, o tempo determinado para o envio de dados de monitoramento costuma ser da ordem de minutos. Isso significa que, se ocorrer alguma falha na infraestrutura, podem se passar vários minutos até que seja notado que houve uma variação no volume de dados em trânsito. Por outro lado, se um novo elemento A começa a encaminhar tráfego na direção de um outro elemento B, podem se passar alguns minutos até que este novo tráfego de entrada seja notificado pelo B para o painel de controle da infraestrutura.

Para reduzir os tempos de retardo para notificação de eventos no painel de controle da infraestrutura, o UDP-Flex implementou um mecanismo experimental – chamado de Disparo de Gradiente - que envia uma informação de monitoramento atualizada sempre que detecta uma variação da taxa de recepção de dados. Na implementação atual, o disparo ocorre sempre que for detectada uma variação na taxa de bytes recebidos (calculada a cada segundo) for maior do que 10. O algoritmo funciona utilizando os seguintes elementos:

- uma instância do UDP-Flex em modo *forwarding* não recebe nenhum fluxo, logo sua taxa de recepção é zero;
- uma instância começa a receber seus primeiros pacotes e após 1 segundo, calcula a taxa de recepção A. Como A é maior do que zero, então um pacote de monitoramento é enviado imediatamente;
- a cada segundo, a instância apura a taxa de bytes por segundo B e compara com a taxa A apurada na última vez que um pacote de monitoramento foi enviado. Se (max(A,B) / max(min(A, B), 1e-10)) >= 10, então um pacote de monitoramento é transmitido. Se a taxa não variar num fator de 10 (varia de 1Mbps para 100kbps, por exemplo), então o pacote de monitoramento só será enviado no próximo *heartbeat*.
- Note que se houve um problema na infraestrutura e a instância deixar de receber pacotes, então é provável que a expressão gere um valor maior do que 10 após poucos segundos. Isso forçará a transmissão de um pacote de monitoramento, reduzindo o tempo para que o problema seja detectado no painel de controle.

Configuração via MQTT

Como requisito para integração com a plataforma V-PRISM, o UDP-Flex é capaz de receber comandos via MQTT para adicionar e remover portas de saída. Como a comunicação com o servidor MQTT se dá através do protocolo TCP, optou-se por introduzir o mecanismo de *threads* na implementação do UDP-Flex quando executa em modo forwarding. Contudo, providências foram tomadas para garantir que a tarefa de encaminhar dos pacotes recebidos para as portas de saída sofresse o menor impacto possível com a implementação dessa funcionalidade, evitando a introdução de atrasos adicionais na propagação dos pacotes.

Atualmente, a implementação aceita dois comandos via MQTT:

- Adicionar uma nova porta de saída e
- Remover uma porta de saída.

Em ambos os casos, o comando consiste numa mensagem enviada para o tópico que é monitorado pela instância conforme a opção “-mt” na linha de comando que a inicia. Esse comando obedece ao seguinte padrão:

- IP destino
- Porta destino
- ID da instância
- Letra do Comando: A para adicionar e R para remover

A semântica das operações é a óbvia, sendo que, no momento:

- não é possível o mesmo destino (IP + porta) ser porta de saída mais de uma vez.
- Não há problema caso não exista a porta de saída especificada.

Detalhes da implementação

A versão atual da implementação foi desenvolvida utilizando a linguagem C, padrão C11. O compilador utilizado foi o “gcc version 7.5.0” sobre o sistema operacional Ubuntu 18.04. Todo o desenvolvimento foi realizado para uma plataforma de 64 bits. O resultado da compilação (em 03/07/2020) produziu um arquivo executável com menos de 59KB.

Com o objetivo de simplificar a utilização, todo o projeto é composto de 6 arquivos de texto, sendo: (a) o primeiro deles o Makefile; (b) todo o código fonte desenvolvido especificamente para o componente está contido em apenas um arquivo C (main.c); (b) 4 arquivos adicionais que implementam o acesso ao MQTT devem ser compilados conjuntamente para produção do executável.

A versão mais recente do código fonte do programa pode ser encontrada no site: https://github.com/midiacom/alfa/tree/master/virtual-nodes/vms/udp_flex

Conclusão

A implementação do UDP-Flex atendeu a todos requisitos do projeto. O componente já está complemente integrado à plataforma V-PRISM.