UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
DEPARTAMENTO DE INFORMÁTICA APLICADA
INF01002 – Protocolos de Comunicação

SEMESTRE 2025/1
TRABALHO PRÁTICO: EXPLORANDO P4 PARA ANÁLISE DE DESEMPENHO

Crash Course sobre In-Band Network Telemetry (INT)

Esta seção se destina a introduzir o mecanismo de In-Band Network Telemetry [1], um hot topic em pesquisa que surgiu com o advento de redes programáveis. INT permite a coleta de informações a partir dos switches de uma rede, e pode ser utilizado por exemplo para identificar problemas de desempenho e coletar métricas sobre o funcionamento da rede. A ideia geral é que os pacotes dos end-hosts sejam monitorados obrigatoriamente pelos próprios switches.
A Figura 1 e a descrição a seguir apresentam um exemplo resumido do funcionamento do sistema INT:

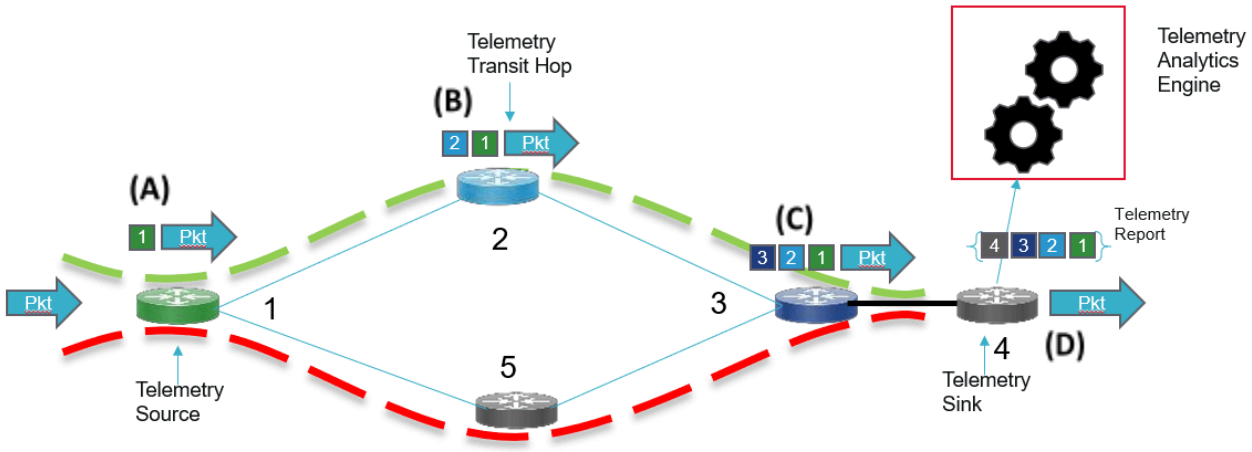

Figura 1 – Workflow de funcionamento resumido da In-Band Network Telemetry

1) Todo pacote inserido na rede recebe necessariamente uma *tag* com um cabeçalho INT caso este não esteja presente (passo A). Um exemplo de cabeçalho INT é ilustrado na Figura 2. Esta verificação de existência do cabeçalho INT deve ser feita pelo switch e inserido obrigatoriamente pelo switch caso não esteja presente. A inclusão de um cabeçalho pode ser feita usando a função `setValid` no cabeçalho desejado (desde que seu nome esteja já incluso no *deparser*). Isto garante que end-hosts não irão burlar o sistema não inserindo o cabeçalho INT.

2) A cada salto, o switch deverá ler todos os cabeçalhos. Para ler um número arbitrário de cabeçalhos, será necessário fazer um controle de quantos cabeçalhos já foram colocados. Isto será um dado importante no cabeçalho INT, que será usado por uma primitiva do parser.

3) Em seguida, o switch irá criar um cabeçalho filho ao cabeçalho INT (passos B & C). Este cabeçalho filho terá seu próprio **`header type`**. Este cabeçalho filho conterá dados sobre a condição do switch que deverão ser carregados pelo programa P4. Os dados a serem usados serão escolhidos pelo programador. O cabeçalho filho do switch atual tem que ser colocado após todos os cabeçalhos filhos anteriores que, por sua vez, serão colocados depois do cabeçalho pai.

4) Quando estivermos no salto imediatamente anterior ao destino do pacote (passo D), é possível: (i) deixar os cabeçalhos serem recebidos pela aplicação destino e interpretados por ela própria; (ii) criar uma cópia do pacote (clone) e: remover os cabeçalhos INT do primeiro pacote, fazendo assim com que o pacote seja entregue de forma transparente para a aplicação destino, sem que esta saiba que o pacote foi monitorado e, para o segundo, remover todos os dados do *payload* do pacote e enviar somente o cabeçalho para algum outro host, responsável pelo monitoramento global da rede (*telemetry analytics engine*).

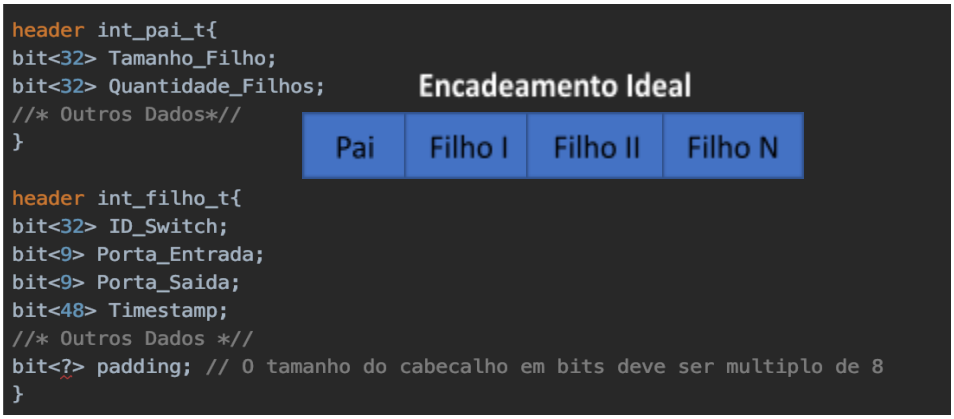

Figura 2 – Exemplo de cabeçalhos INT

ESPECIFICAÇÃO DO TRABALHO

Apresentadas as informações contextuais acima, e assumindo uma rede totalmente programável via SDN/P4, projete e desenvolva um mecanismo que permita – via *In-Band Network Telemetry* [1] – depurar precisamente problemas de desempenho em uma aplicação (*i.e.*, conjunto de fluxos) de interesse.

A ideia básica é que cada pacote desses fluxos, ao ingressar na rede, tenha um cabeçalho de telemetria adicionado. **A cada salto, incluindo o primeiro e o último, devem ser coletadas e armazenadas (nos cabeçalhos INT) as "condições" de rede** (*e.g.*, *timestamp*, *delay* do salto, tamanho da fila, porta de entrada e saída e fluxos competidores) **observadas pelo pacote**, além de **identificadores** (*e.g.*, **ID do dispositivo**). Ao chegar no end-host destino, o programa receptor deverá identificar os cabeçalhos adicionados no pacote e informar as condições de rede capturadas por ele.

Requisitos Funcionais

1. Verificação da presença do cabeçalho INT pai e inclusão deste cabeçalho nos pacotes em que ele não esteja presente.
2. Criação e inserção, a cada hop, de um cabeçalho INT filho.
3. Preenchimento dos campos dos cabeçalhos filhos com as informações dos switches

    a. O subconjunto **mínimo** de métricas a serem coletadas é (mas métricas adicionais são desejáveis):
        i. Timestamp de entrada;
        ii. Porta de entrada;
        iii. Porta de saída;
        iv. ID do switch.
4. Ao chegar no host destino, o host deverá ser capaz de, por software (no arquivo **`receive.py`**), extrair as informações do pacote e separá-las do *payload* original do pacote.
    a. E.g., se foram inseridos 20 bytes de cabeçalhos INT, o **`receive.py`** deverá separar os 20 bytes INT do *payload* original e exibi-los de maneira distinta.

Getting Started

1. Faça o download da máquina virtual com o ambiente P4 já previamente configurado e disponibilizado para os alunos;
2. Baixe o zip da pasta TP-Protocolos (disponível no Moodle da disciplina) e descompacte-o no local de sua preferência; acesse então, a pasta "TP-Protocolos/skeleton/TP-skel";
3. Inicialize o Mininet usando "**`make`**" e teste a conectividade básica entre hosts:
    a. Ao abrir o Mininet, use o comando "**`xterm h1 h2`**";
    b. Em um dos terminais, digite **`ifconfig`** e descubra o IP deste host. Logo após, execute o arquivo **`receive.py`**;
    c. Use o outro terminal para executar um "**`send.py <ip> <mensagem>`**" sendo **`<ip>`** o IP do primeiro terminal e mensagem uma string. Se o pacote e a mensagem aparecerem corretamente no **`receive.py`**, então há conectividade entre os dois hosts (se houver perda da conectividade durante o processo de desenvolvimento, significa que algo não foi feito da forma correta, e é sugerido sempre refazer esse teste para não se perder);
4. Edite o arquivo **`basic.p4`** para incluir o seu mecanismo de INT;
5. Edite o arquivo **`receive.py`** para receber e ler o seu cabeçalho INT;
6. Implemente seu mecanismo de INT de forma que este funcione para topologias genéricas, e não somente para a topologia proposta para este trabalho.

Requisito bônus [1] (**vale 1,0 ponto extra na nota do trabalho**): À medida que cada salto adiciona seus dados de monitoramento no pacote, o tamanho do pacote aumenta e pode chegar ao ponto de poder ultrapassar o MTU (tamanho máximo de um pacote que pode ser transmitido), causando a fragmentação ou a perda do pacote. Assim, um switch deve adicionar um novo cabeçalho de telemetria no pacote apenas se após a adição deste o tamanho do pacote não for superior ao MTU. Para a resolução do requisito, vamos assumir que o valor do MTU é de 1500 bytes. Desta forma, para atender esse requisito é necessário alterar o **`basic.p4`** para:

1. Obter o tamanho do pacote na chegada (disponível nos metadados fornecidos pela arquitetura: standard_metadata).
2. Calcular o tamanho do cabeçalho INT que será adicionado (o tamanho de um cabeçalho, que não contém nenhum campo varbit, é constante e representado pela soma dos tamanhos dos seus campos).
3. Somente adicionar o cabeçalho INT se o tamanho do pacote resultante não ultrapassar o MTU.
4. Um novo campo deve ser criado no cabeçalho INT_PAI para indicar se houve estouro do valor do MTU. Esse campo pode ser do tipo bit<1> iniciando com o valor zero e deve ser setado para 1 se houver estouro do MTU em algum switch.

Ao exibir os dados INT coletados, o `receive.py` deve ser alterado para indicar se foi possível coletar os dados de telemetria de todos os switches do caminho ou se houve estouro do MTU.

Requisito bônus [2] (**vale 1,0 ponto extra na nota do trabalho**): A sua implementação INT atual presume que o programa que recebe o pacote estará preparado para extrair o cabeçalho INT e manipulá-lo corretamente. Isso é uma premissa forte, uma vez que necessitaria que as aplicações responsáveis pelos fluxos monitorados soubessem fazer essa manipulação (aplicações como Netflix, YouTube, e-mail). Para fazer um INT realmente transparente, precisaremos remover os cabeçalhos INT adicionados e exportá-los para uma aplicação de monitoramento antes de entregá-lo ao destino. Assim, no último salto, ao invés de entregar o cabeçalho de telemetria junto ao pacote para o programa receptor, o cabeçalho de telemetria deverá ser removido do pacote e enviado a uma aplicação de monitoramento que está executando em um controlador centralizado. Essa aplicação de monitoramento deverá, em tempo real, apresentar o desempenho da aplicação escolhida (*e.g.*, *delay*, tamanho de filas, etc). Para isso, você deverá, no último switch antes de chegar no host receptor, criar uma cópia/clone do pacote; (a) Para o pacote original, remover, por meio do arquivo `basic.p4`, os cabeçalhos INT antes de entregar o pacote ao host destino, para que o `receive.py` não enxergue nenhuma alteração entre o *payload* enviado pelo `send.py` e o recebido; (b) Para o pacote copiado, remover todo o *payload* e entregar o pacote para a aplicação de monitoramento, que desta forma irá ver somente os cabeçalhos INT.

1. Para essa funcionalidade, o `receive.py` deverá estar rodando também em outro host que não o receptor.
2. Edite novamente o `basic.p4` para que seja direcionado, **inteiramente pelos switches da rede**, um clone do pacote para o host onde esteja rodando o `receive.py`, que deverá exibir as informações INT recebidas.
3. O `receive.py` do receptor terá que enxergar apenas o pacote, enquanto o `receive.py` do monitor (telemetry engine) terá que enxergar apenas os cabeçalhos INT.

OBSERVAÇÕES IMPORTANTES

- O trabalho poderá ser realizado em grupos de 2 ou 3 integrantes;
- A topologia a ser monitorada já está presente na pasta do trabalho;
- Qualquer fluxo entre dois hosts da topologia deverá ser monitorável pelo seu programa.

AVALIAÇÃO

Deverá ser entregue o código fonte do mecanismo, documentado, e um relatório contendo:

- Descrição da lógica de funcionamento do mecanismo;
- Resultados de experimentos conduzidos (incluindo *screenshots* de telas e explicações), que permitam observar que o mecanismo foi capaz de implementar uma lógica válida de análise baseada no mecanismo solicitado (INT);
- Descrição dos eventuais problemas que você encontrou durante a implementação e como estes foram resolvidos (ou não);
- Breve descrição dos passos tomados para implementação de cada requisito funcional.

A nota do trabalho está condicionada à apresentação/demonstração do mesmo. **Todos** os componentes do grupo precisarão, impreterivelmente, comparecer à apresentação, pois serão inquiridos a respeito de diferentes aspectos da realização do trabalho.

Prazos: A data da entrega do trabalho é dia 25/06/2025 às 10:30. As demonstrações acontecerão no mesmo dia, no horário da aula, em formato remoto síncrono.

REFERÊNCIAS

[1] **In-bandNetworkTelemetry (INT)**, WorkingDraft.
https://github.com/p4lang/p4-applications/blob/master/docs/INT_v2_1.pdf
[2] **P4 Cheat Sheet:** https://github.com/p4lang/tutorials/blob/master/p4-cheat-sheet.pdf
[3] **P4$_{16}$ Language Specification:**
https://p4.org/wp-content/uploads/2024/10/P4-16-spec-v1.2.5.pdf
[4] **Bmv2 Simple Switch documentation:**
https://github.com/p4lang/behavioral-model/blob/main/docs/simple_switch.md
[5] **Python Scapy**: https://scapy.readthedocs.io/en/latest/build_dissect.html