Universitat de Girona
Escola Politècnica Superior

Grau en Enginyeria Informàtica

PROJECTE FINAL DE GRAU

Desenvolupament d'una eina basada en eBPF/XDP per al monitorage del rendiment d'una xarxa

---

Autor:
Gerard Vila Martínez

Tutors:
Jordi Paillisse Vilanova
Lluís Fàbrega Soler

RESUM

Convocatòria:
Juny 2023

Departament :
Arquitectura i Tecnologia de computadors

Introducció

El monitoratge del rendiment d'una xarxa té cada vegada més importància en les xarxes actuals. L'obtenció de paràmetres estadístics que mesuren l'ocupació dels enllaços i el retard i pèrdues de paquets en un camí (retard mitjà, variació del retard, i percentatge de paquets perduts), permet conèixer l'estat de la xarxa i la detecció de mal funcionaments. A partir de l'anàlisi d'aquestes dades i de tècniques de predicció (basades en machine learning) els sistemes de gestió poden prendre decisions per aconseguir un ús eficient dels recursos de la xarxa.

Per altra banda, l'aparició de tecnologies com P4 (Programming Protocol-independent Packet Processors) o eBPF/XDP (extended Berkeley Packet Filter/eXpress Data Path) per a la programació de l'anomenat pla de dades (Programmable Data Plane o PDP), és a dir, per a la programació del comportament de dispositius de xarxa (commutadors, routers, tallafocs, NAT, etc.), permeten definir d'una manera més flexible com aquests dispositius processen i reenvien els paquets, i alhora obren la possibilitat de construir eines de monitoratge (i també en altres àmbits) més potents que les existents (observació de més variables, càlculs estadístics fets en el mateix dispositiu, processament més ràpid, etc.). En concret, eBPF és un conjunt d'instruccions i un entorn d'execució dins del kernel de Linux, que permet la programació del kernel en temps d'execució. eBPF, a més, es pot utilitzar per programar l'XDP, una capa de xarxa del kernel que processa els paquets ben a prop de la interfície de xarxa aconseguint, per tant, un processament més ràpid dels paquets.

Objectius

L'objectiu d'aquest projecte és desenvolupar una eina basada en eBPF/XDP per al monitoratge del rendiment d'una xarxa, concretament per obtenir paràmetres que mesurin el temps d'encuament i pèrdues de paquets en la cua de sortida d'un enllaç d'un dispositiu de xarxa, i també el retard i les pèrdues de paquets en un camí.

Per portar a terme aquest treball es seguiran els següents passos:

- Estudiar la tecnologia eBPF/XDP
- Estudiar els conceptes relatius al rendiment d'una xarxa i als paràmetres de mesura.
- Definir els requisits de l'eina de monitoratge.
- Construir l'eina de monitoratge.
- Construir les eines necessàries per a calcular els valors del rendiment.
- Fer proves per verificar el seu funcionament.

Disseny

En una xarxa de commutació de paquets un paquet és portat des d'una estació origen a una estació destí seguint un camí a través de la xarxa formada per una seqüència d'enllaços i nodes (commutadors). El retard d'un paquet és el temps que tarda a viatjar de l'estació origen a l'estació destí; més concretament, és el temps que passa

des que s'envia el primer bit a l'entrada de la xarxa fins que arriba el darrer bit a la sortida de la xarxa.

Aquest retard és la suma dels temps de processament, els temps d'encuament, els temps de transmissió i els temps de propagació, en els nodes i enllaços del camí.

Llavors, el programa que es vol fer ha de ser capaç d'aconseguir el moment d'entrada i de sortida d'un paquet. A partir d'aquests valors, podrem obtenir el temps que el paquet ha estat dins del node, aquest temps resultant correspon al temps de processament sumat al temps d'encuament. Això es calcularà per cada paquet individualment que passi pel node.

Així doncs, el programa ha de ser capaç de capturar els paquets, injectant codi al kernel de Linux. Els paquets capturats també han de ser accessible des de fora del kernel, des del programa per, seguidament, guardar-nos els resultats en un arxiu.

Com es veu en la Fig. 1, el codi principal s'encarrega d'injectar les funcions de captura en l'entrada i la sortida, i farà la recollida de paquets que s'hagin capturat.

Per tant, la funció de captura identifiquen el paquet i el guarden en un *mapa hash*. Pel que fa als paquets entrants, es guardarà amb marca de temps de quan ha entrat el paquet. En canvi, la captura a la sortida, es comprovarà si s'ha capturat a l'entrada, si és així, s'actualitzarà el paquet amb el moment de sortida, altrament es guarda com un paquet nou.

Després, el codi principal té accés al *mapa hash*, permeten extreure els paquets en el mateix moment que es capturen, i guardar-los en un arxiu.

Com que es necessita que els paquets tinguin la capa del protocol IP, per poder fer les proves es fa ús d'eines que generin trànsit d'aquests paquets. En aquest cas, s'ha utilitzat tant l'eina *ping*, que genera trànsit de paquets *icmp*, com l'eina *iperf*, que genera tants paquets *tcp* com *udp*.

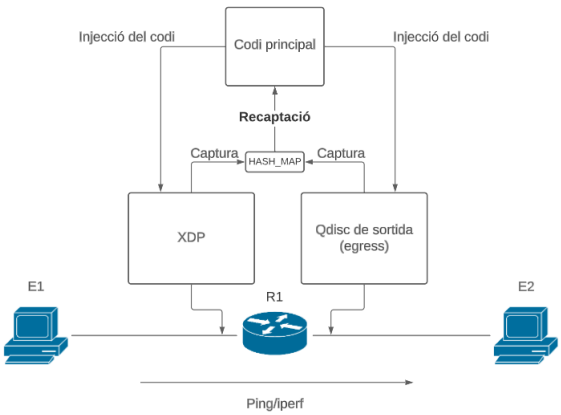

FIGURA 1: L'estructura del programa en l'escenari

Doncs, per poder fer tot això es crea un escenari de xarxa, aquest escenari s'ha fet amb màquines virtuals, cosa que limita a quatre interfícies de xarxa. Així que, s'ha creat un escenari on, el node que captura el trànsit tingui totes les interfícies possibles, les quatre (vegeu la Fig. 2).

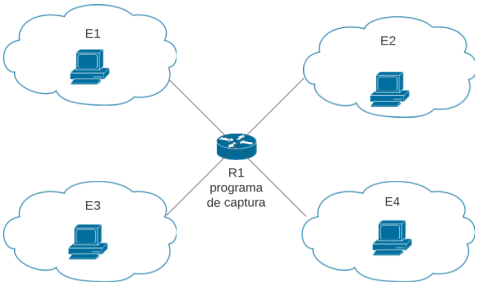

FIGURA 2: Escenari de treball 2

Implementació

Pel que fa al codi, el programa principal ha de ser capaç d'injectar unes funcions de captura al kernel. Per això, fa ús de l'eina Bcc, per la captura de paquets entrants, i de l'eina TC(*traffic control*), per la captura de paquets sortints.

La captura de paquets entrants es fa al bloc *xdp* del kernel. Aquest bloc hi accedim amb l'eina Bcc. En aquest punt, en rebre un paquet, s'accedeix a la capa IP del paquet, si en té, i es guarda el paquet amb una marca de temps de quan ha entrat.

La captura de paquets sortints es fa al bloc *qdisc* del kernel, usant l'eina TC. També, en arribar-hi un paquet, s'accedeix a la capa IP del paquet. Aquest l'identifiquem i es busca al mapa de paquets guardats. Si es troba, s'hi afegeix la marca de temps de sortida al paquet en el mapa, si no, es guarda el paquet amb el moment de sortida.

Els paquets es guarden en un *mapa hash* (vegeu la Fig. 1). Aquest mapa fa servir una clau per identificar els paquets, per això, es pot fer servir tres camps per identificar unívocament un paquet, l'adreça IP, el camp identificador i el número de fragment.

Llavors, els paquets queden guardats en un mapa que, fent servir l'eina Bcc hi tenim accés. Per tant, el programa principal té accés a les dades que es capturen i les podem recol·lectar.

Així doncs, quan el programa principal observi un paquet capturat, se'l guardarà si s'ha capturat a l'entrada i la sortida o si és un paquet perdut.

Llavors, els paquets recol·lectats es guarden en un arxiu JSON.

Corresponent a les pèrdues, els paquets que es capturin s'han de mantenir temporalment mentre no es capturi a la sortida o passi prou temps que es consideri que s'ha perdut. Per tant, s'ha determinat empíricament que el temps a esperar un paquet sigui de 0.5 segons.

Resultats

Per obtenir els resultats, s'engega el programa en l'escenari descrit anteriorment (vegeu la Fig. 2). Usant l'eina *iperf* es genera trànsit de paquets *tcp* entre l'estació E1 i l'estació E2. Durant l'enviament de paquets, s'engega el programa al router R1. Durant un segons, capturem els paquets que s'envien, capturant-ne així aproximadament 11000 paquets. Aquests queden guardats en un arxiu JSON. Posteriorment,

utilitzant un petit programa, mostrem en un histograma els retards dels paquets (vegeu la fig. 4) i també els retards asonseguits, com es mostra a la Fig 3.

```
user@host:~/eBPF$ python3 calculate.py output2.json
Màxim i mínim
1945.5710000000001 µs i 4.837 µs
Mitjana de retard
187.47311701766264  µs
```

FIGURA 3: L'execució del programa de processar els paquets

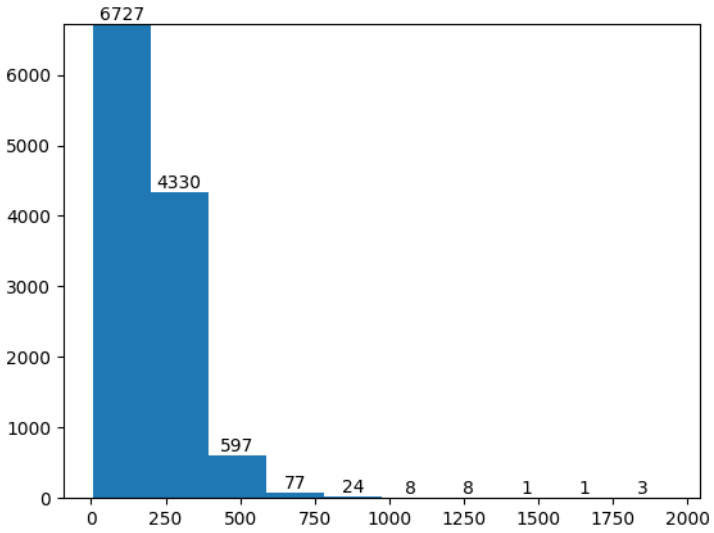

FIGURA 4: L'histograma del retard dels paquets capturats

Resum i conclusions

Per resumir, s'ha construït una eina que permet mesurar el temps d'un paquet dins del kernel. Aquesta eina s'ha fet usant l'eBPF/XDP i TC per injectar el codi de captura al kernel de Linux. S'han creat dues funcions per fer les captures de paquets.

Aquest programa ha sigut provat en un escenari en un router, tant amb dues interfícies de xarxa com amb quatre. S'ha provat amb paquets *icmp*, *tcp* i *udp* usan les eines *ping* i *iperf*.

També s'ha provat el programa en un escenari en el qual hi ha pèrdues de paquets. S'ha construït un programa que permet mostrar els retards en un histograma, i també mostra el retard mitjà.