MARIONNET

Laboratoire de réseaux virtuels TCP/IP

***Jean-Vincent Loddo*[1]**

***Luca Saiu*[2]**

1. Membre permanent (LCR, 2003-)

2. Thèse et ATER (LCR, 2007-2012)

Logiciel libre (GNU GPL)

http://www.marionnet.org

Motivations

Pour l'étudiant en réseaux/informatique :

- permettre le travail personnel et le travail à distance sur des réseaux d'ordinateurs grandeur nature
- être seul maître à bord dans la mise en œuvre complète d'un réseau (définition, câblage, configuration des appareils et des services)
- travailler les différentes couches de l'architecture TCP/IP (en IPv4 ou IPv6)

Pour l'enseignant :

- faciliter la conception, la correction et le partage d'exercices

Pour une structure universitaire (gestion des salles de TP) :

- répondre aux problèmes d'équipement réseau
- répondre aux problèmes de disponibilité des salles

Pour le chercheur en informatique :

- expérimenter la programmation multi-paradigmes
- preuves de concepts (test de nouveaux principes de programmation)

Composants virtuels

Machine (emulation)

- User Mode Linux (UML patché) + Copy on Write (COW)

Concentrateur (Hub)/commutateur (switch) (simulation)

- Virtual Distributed Ethernet (VDE)

Routeur (simulation/emulation)

- Quagga (fork de Zebra)

Câble Ethernet droit et croisé (simulation)

- Virtual Distributed Ethernet (VDE)

Passerelles virtuel ↔ réel (simulation)

- Gateway (niveau 3, slirpvde de VDE)
- Bridge (niveau 2, noyau linux)

1. Couche physique

Travailler au niveau de la couche physique signifie définir l'ensemble des appareils utiles, les ports physiques nécessaires pour chaque appareil, et l'ensemble des connexions entre ces ports. Des dysfonctionnements associés aux ports d'entrée-sortie (voir Figure 1.d) peuvent être éventuellement simulés, si l'utilisateur le souhaite.

(a) Composants physiques — *(d) Simulation de dysfonctionnements*

(b) Cartes Ethernet — *(c) Espionner une connexion* — *(e) Espionner un réseau* — *(g) Connexions entre hubs* — *(f) Ports physiques* — *(h) Connexions entre machines*

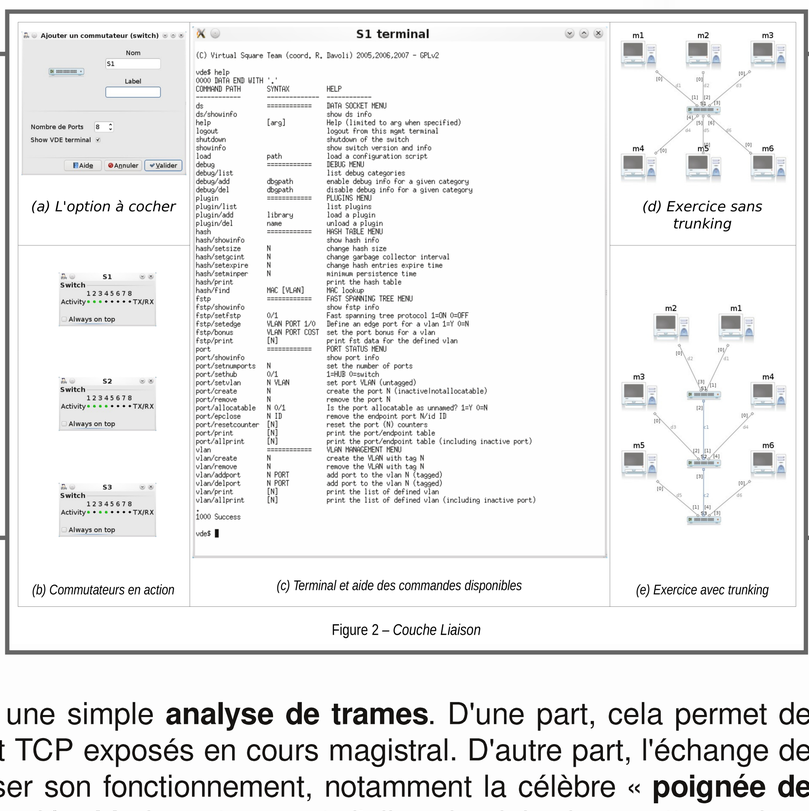
Figure 1 – *Couche Physique*

Dessin automatique du graphe

L'utilisateur ne dessine pas le réseau directement. En ajoutant les composants dans son projet, c'est-à-dire dans sa salle de TP virtuelle, il définit l'ensemble des appareils ou nœuds (tout ce qui n'est pas de type câble) et des connexions ou arêtes (câbles) qui constituent son réseau. Le graphe du réseau, qui est dessiné dans la partie centrale de l'interface, est alors mis à jour automatiquement en tenant compte des ajouts et retraits de composants et de l'état courant de chaque appareil (éteint, en marche ou en pause) et de chaque câble (branché, débranché).

2. Couche Liaison

À ce niveau, **trois types d'exercices** peuvent être abordés avec Marionnet : étudier la structure des trames Ethernet 802.3, la mise en œuvre de VLAN de type 1 (basées sur une partition des ports physiques) et l'étude, et en partie la configuration, du Spanning Tree Protocol (STP) entre commutateurs reliés en boucle. Des exercices complémentaires très formateurs comme la répartition de charge (par exemple par l'Ethernet Bonding) et les pare-feux de niveau 2 (wire firewall) sont possibles en exploitant les fonctionnalités réseaux avancées du noyau Linux.



(a) L'option à cocher — *(c) Terminal et aide des commandes disponibles* — *(d) Exercice sans trunking* — *(b) Commutateurs en action* — *(e) Exercice avec trunking*

Figure 2 – *Couche Liaison*

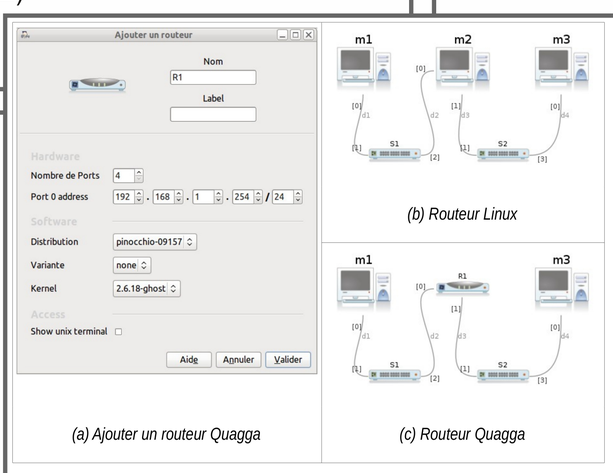
3. Couche Internet

Routeur Linux (statique)

Pour découvrir le routage internet, l'utilisateur peut construire un réseau de trois machines (voir Figure 3.b), dont une (m2), avec ses deux cartes réseaux, fera office de « routeur du pauvre ». Une fois la configuration des adresses réalisée, il faudra considérer les tables de routage. Mais avant cela, on peut se préoccuper du fonctionnement du protocole ARP, partie intégrante de IP, nécessaire à la résolution adresse IPv4 → adresse MAC. Le cache ARP des machines Linux est manipulable par la commande arp, qui permet d'ajouter des entrées statiques (permanentes) ou encore de supprimer des entrées. Un outil tel que arpsk, par exemple, peut être utilisé pour introduire la notion de ARP-spoofing lorsque l'on s'inquiète de la sécurité.

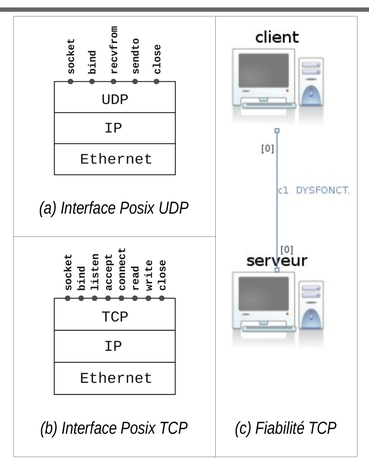
Routeur Quagga (dynamique)

Marionnet rend disponible un composant « boîtier routeur » spécialisé dans le routage, aussi bien statique que dynamique. Lorsqu'un tel routeur est introduit dans le réseau, l'utilisateur choisit une adresse IP pré-déterminée qui permettra l'accès en telnet à l'équipement (voir Figure 3.a, champ « Port 0 address »). Pour réaliser ce composant, Marionnet utilise le logiciel Quagga embarqué dans une machine virtuelle. L'icône représentant ce composant est un boîtier similaire à ceux des hubs et switchs (voir Figure 3.c, où le routeur est R1). L'utilisateur se connecte par le port 0 avec la commande telnet sous GNU/Linux (depuis la machine m1) et peut ensuite configurer le routeur avec des commandes d'inspiration IOS CISCO (enable, configure terminal, interface, ip address...). Quagga peut émuler un routeur BGP, OSPF, ISIS, RIP et RIPng toujours avec des commandes d'inspiration IOS CISCO.



(a) Ajouter un routeur Quagga — *(b) Routeur Linux* — *(c) Routeur Quagga*

Figure 3 – *Couche Internet (IP)*



(a) Interface Posix UDP — *(b) Interface Posix TCP* — *(c) Fiabilité TCP*

Figure 4 – *Couche Transport*

4. Couche Transport (OSI 4)

Le niveau Transport peut être abordé par une simple **analyse de trames**. D'une part, cela permet de réviser le format des datagrammes UDP et TCP exposés en cours magistral. D'autre part, l'échange de trames générées par TCP permet de réviser son fonctionnement, notamment la célèbre « **poignée de main** ». Grâce aux dysfonctionnements simulés, Marionnet permet d'aller plus loin dans cette analyse du comportement : on peut **« stresser » TCP** pour voir jusqu'où il résistera aux pannes matérielles. Enfin, un troisième type d'exercice qui semble fondamental, pour un traitement exhaustif de la couche Transport, consiste à prendre connaissance des services offerts par UDP et TCP (voir Figure 4).

5. Couche Application (OSI 5 à 7)

Marionnet peut servir de plate-forme d'entraînement et de **test de services réseaux** divers et variés. Les administrateurs débutants peuvent tester divers serveurs avec divers clients avant une éventuelle mise en production réelle, comme par exemple DNS et HTTP.

Développement

Licence Libre (GNU GPL)
40k lignes de code OCaml
Patch du noyau Linux (C)
Internationalisé et complètement **traduit en 9 langues par des bénévoles** :

- French, English, German, Greek, Italian, Russian, Slovak, Spanish, Turkish

Hébergement du code sur launchpad (`https://launchpad.net/marionnet`)
Autres contributeurs au LIPN : Franck Butelle (packaging)

People @LIPN

Jean-V. Loddo
LCR

Luca Saiu
LCR

Franck Butelle
AOC