DONE - Docker Orchestrator for Network Emulation

Studio di fattibilità

CdL in Sicurezza dei Sistemi e delle Reti Informatiche

Samuele Manclossi, Melissa Moioli, Tiziano Radicchi
09882A, 09831A, 12172A

26 marzo 2024

"Non quia difficilia sunt non audemos, sed quia non audemos difficilia sunt"

— Seneca

Abstract

Si propone lo studio di fattibilità per la proposta di progetto "DONE": la creazione uno strumento, ispirato a IMUNES, per l'emulazione di reti mediante l'utilizzo automatizzato di Docker, interfaccia grafica e terminale.

Esso si basa sugli stessi principi di virtualizzazione alla base di altri software di emulazione di reti. L'obiettivo è quello di fornire un ambiente di sviluppo e test per reti complesse, in modo da poter testare nuove configurazioni di rete e nuovi servizi senza dover ricorrere a costosi e complessi apparati fisici, garantendo allo stesso momento dipendenze da pochi strumenti quali Docker, OpenVSwitch e la segregazione dei namespaces offerta dai sistemi operativi Linux.

Il nostro studio si concentrerà su tre macroargomenti: la virtualizzazione delle topologie di rete, la logica di emulazione, la logica di interazione, che permetterà all'utente di interagire con il programma sia tramite interfaccia grafica che da terminale.

Contents

1 Introduzione

Alla luce delle funzionalità che gli emulatori di rete oggi esistenti offrono, desideriamo, con l'obiettivo di approfondire le nostre conoscenze in ambito di reti e di sviluppo software, proporre una soluzione alternativa che offra feature extra ed un miglioramento generale della qualità di utilizzo.

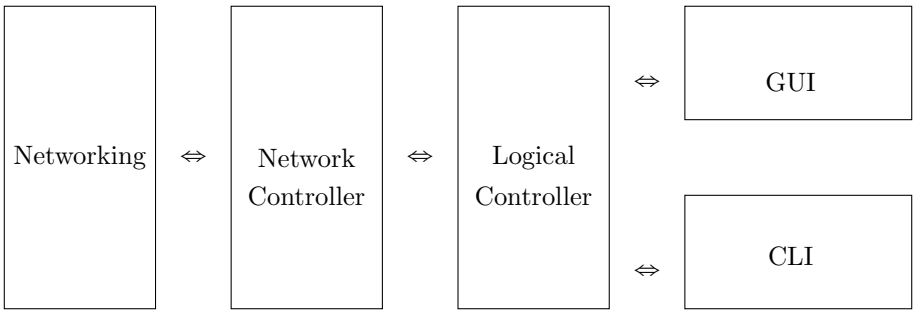
1.1 Funzionalità

Le funzionalità che desideriamo offrire attraverso l'utilizzo di `DONE` sono le seguenti:

- L'utente può creare, aprire, modificare progetti di topologie di rete, aggiungendo apparati, definendo collegamenti tra questi, nonché fare uso di apparati specifici volti al collegamento della topologia definita a reti esterne alla simulazione.
- Una volta realizzata la topologia desiderata, l'utente può salvare su file il progetto, con lo scopo di poter riprendere il lavoro in un secondo momento
- L'utente può assegnare per ogni singolo device delle configurazioni, che verranno applicate all'avvio della simulazione
- L'utente può avviare la simulazione, con la possibilità in seguito di accedere tramite terminale ai singoli nodi per aggiungere ulteriori configurazioni, per testare la connettività e/o effettuare troubleshooting. Il tutto, esclusivamente tramite linea di comando

1.2 Architettura

Alla luce delle funzionalità desiderate, l'architettura che abbiamo ideato per raggiungere gli obiettivi preposti è così organizzata:



Per tanto, le componenti da realizzare sono:

- **Networking**: Utilizzo di container e apparati virtualizzati. Si tratta del modo in cui vengono realizzati, come in IMUNES, i componenti veri e propri. Essi sono poi connessi tra di loro mediante gli appositi comandi e possono simulare una rete. Sarà gestito dal relativo controller attraverso utilizzo di librerie apposite sviluppate ad hoc.
- **Network Controller**: Automatizzazione, a fronte di una topologia creata, della creazione della configurazione di rete
- **Logical Controller**: Logica di controllo, a più alto livello, che possa gestire:
  - Strutture per gestire la topologia creata
  - Salvataggio della topologia
  - Gestione e salvataggio delle configurazioni
  - Invio delle informazioni necessarie al network controller per realizzare la topologia creata
- **GUI**: si tratta dell'interfaccia su cui l'utente può disegnare la topologia logica della rete, posizionando quindi nodi e link tra nodi, trascinandoli, modificandoli e interagendoci in genere
- **CLI**: da qui si possono lanciare i comandi di configurazione dei vari componenti. Essa aprirà un editor di testo sui file di configurazione, permettendo quindi di modificarli, salvarli e caricare le modifiche anche nella visualizzazione GUI. Permetterà di accedere a CLI dedicate ai singoli nodi per inserire configurazioni

Gli obiettivi che desideriamo perseguire attraverso questo progetto sono:

- **Ridurre le dipendenze** al minimo necessario, in modo da rendere il programma il più portabile possibile
- Fornire la **capacità di salvare** non solo la topologia fisica ma anche tutte le configurazioni, senza bisogno di script ulteriori
- **Maggiore pulizia dell'ambiente**, evitando che Docker rimanga in uno stato intermedio (ossia con i vecchi container ancora presenti) in caso di chiusura della applicazione senza arresto della simulazione, nonché pulizia dalle altre interfacce e apparati virtuali creati
- Offrire una soluzione open-source che possa essere **estesa** e **modificata**

2 Analisi

Di seguito si analizzeranno in maniera più dettagliata le singole componenti, descrivendo come si intende realizzarle.

Tecnologie utilizzate Complessivamente, il progetto sarà realizzato utilizzando le seguenti tecnologie: linguaggio **C**, con l'uso della libreria `raylib` per la realizzazione dell'interfaccia grafica; **Docker**, per la virtualizzazione dei nodi e dei router; **OpenVSwitch** per la virtualizzazione di switch. L'assenza di ulteriori dipendenze assicura un'elevata portabilità dell'applicazione, basando tutte le configurazioni di rete necessarie su strumenti già presenti in un ambiente Linux, come ad esempio i namespaces e la segregazione dello stack di rete tramite questi.

Come introdotto nella sezione precedente, il progetto si compone di cinque componenti principali:

Networking La componente di networking è interamente gestita attraverso l'uso di `openvswitch-switch`, docker e l'interazione con i namespace dei container creati. La topologia viene realizzata concretamente creando i nodi richiesti, gli switch necessari e realizzando i collegamenti desiderati attraverso la creazione di interfacce virtuali collegate tra di loro mediante cavi virtuali. Alla creazione di un container, per creare dei collegamenti con altri apparati si creano delle interfacce virtuali lavorando direttamente sul namespace del container. Il livello di networking viene gestito direttamente da un network controller, attraverso l'uso di una libreria appositamente realizzata.

Network controller Si occupa di inviare i comandi necessari per la creazione della rete, in base alla topologia precedentemente realizzata dall'utente. Le informazioni relative ad apparati, relativo tipo e collegamenti sono comunicati al network controller dalla logica soprastante.

Logical controller Rappresenta il fulcro della logica di controllo. Riceve informazioni sulla topologia creata dal livello di GUI, le invia al controller di rete all'avvio della simulazione, gestisce il salvataggio dei progetti (topologia ed eventuali configurazioni dei nodi) e l'apertura di progetti salvati in precedenza, comunicando alla GUI quali nodi devono essere visualizzati.

GUI Realizza la componente grafica con cui l'utente può interagire con l'applicazione

CLI Realizza la componente di interfaccia a riga di comando, con cui l'utente può interagire con i singoli nodi della rete, inserendo configurazioni e comandi. Questi comandi si distinguono da quelli che l'utente può specificare per i singoli apparati antecedentemente all'avvio della simulazione.

3 Proof of Concept