UNIVERSITÀ DI PISA

DIPARTIMENTO DI INFORMATICA

Progetto Gestione di Reti

RTT Geo-Location Anomaly Detector

Nome: email:

Nicolò Fontanarosa **n.fontanarosa@studenti.unipi.it**

ANNO ACCADEMICO 2024/2025

Contents

Chapter 1

Introduzione

1.1 Traccia del Progetto

Wireshark Plugin Analisi dell'RTT per la Verifica della Localizzazione IP

Sviluppare un plugin per Wireshark in `Lua` che analizzi il Round-Trip Time (RTT) dei pacchetti di handshake (es. *TCP SYN-ACK*, *TLS ClientHello*) e lo confronti con un valore medio atteso per il paese associato all'IP tramite un database di geolocalizzazione (es. *MaxMind*). L'obiettivo è verificare se l'host che risponde si trova effettivamente nella regione in cui il suo indirizzo IP è stato registrato

1.2 Descrizione

L'architettura del progetto prevede due componenti principali:

1. Un file `generator.py` scritto in Python, che effettua ping verso vari server NTP appartenenti all'*NTP Pool Project* per ottenere RTT medi dal paese da cui viene eseguito lo script verso vari paesi dei server NTP.

2. Uno script `rtt_check.lua` per Wireshark che utilizza i valori calcolati per identificare eventuali anomalie nell' RTT misurato rispetto a quanto atteso dalla localizzazione dell'indirizzo IP fatto con Maxmind DB.

1.3 Background Teorico

1.3.1 Dissector, Post-Dissecotr e Tap

In Wireshark, utilizzando il linguaggio Lua, esistono tre meccanismi principali per ispezionare ed elaborare i pacchetti:

- **Dissector**: utilizzato quando si vuole modificare il modo in cui Wireshark mostra il pacchetto, analizzando il *payload* grezzo di un protocollo.

  - Viene chiamato durante la fase di dissezione, ovvero appena Wireshark identifica il protocollo.

- **Post-dissector**: usato quando si vogliono analizzare i pacchetti dopo che tutti i dissector standard hanno terminato la loro analisi, per aggiungere informazioni extra o confrontare dati tra più protocolli.

  - Non analizza direttamente il *payload*, ma legge i campi già estratti da altri dissector.

- **Tap**: utile per raccogliere statistiche, dati temporali o esportare flussi di pacchetti.

1.3.2 RTT (Round-Trip Time)

L'RTT (Round-Trip Time) è il tempo che un pacchetto impiega per viaggiare dalla sorgente alla destinazione e poi ritornare alla sorgente. È una misura comune utilizzata per valutare la latenza di una rete e può essere influenzato da vari fattori, come la distanza fisica tra i nodi, la congestione della rete, e la qualità della connessione.

1.3.3 Outlier

Un *outlier* è un dato che si discosta significativamente dal resto dei dati in un insieme. In un contesto di RTT, un outlier rappresenta un valore di latenza che è notevolmente più alto o più basso rispetto alla maggior parte degli altri valori raccolti. Gli outlier possono essere causati da vari fattori, come errori di rete, congestione o malfunzionamenti di qualche nodo della rete. È importante identificare e trattare questi valori, in quanto possono influire negativamente sulle analisi statistiche, portando a conclusioni errate.

1.3.4 Media

La *media* è una misura di tendenza centrale che rappresenta il valore medio di un insieme di dati. Nel caso dell'RTT, la media rappresenta il tempo medio che un pacchetto impiega per viaggiare dalla sorgente alla destinazione e ritorno.

1.3.5 Deviazione Standard

La *deviazione standard* è una misura che indica quanto i valori di un insieme di dati si discostano dalla media. Una bassa deviazione standard indica che i valori sono vicini alla media, mentre una deviazione standard elevata implica che i valori sono sparsi su un ampio intervallo. Nel caso dell'RTT, una deviazione standard bassa indica che la latenza è consistente e stabile, mentre una deviazione standard alta potrebbe suggerire variabilità nella connessione o la presenza di problemi di rete.

Chapter 2

Funzionamento

2.1 Files

2.1.1 rtt_check.lua

Lo script `rtt_check.lua` analizza il Round-Trip Time (RTT) dei pacchetti **TCP e ICMP**. Questi pacchetti vengono esaminati estraendone l'RTT e confrontandolo con i valori di RTT attesi in base alla localizzazione geografica dell'indirizzo IP associato al pacchetto tramite database.

Lo script `rtt_check.lua` è stato implementato come un **Post Dissector**

Lo script `rtt_check.lua` aggiunge una voce personalizzata a ciascun pacchetto contenente:

- Il paese reale dell'IP secondo MaxMind: `Detected country`
- Il tempo di risposta misurato (RTT reale): `Mesured RTT`
- Il paese stimato in base all'RTT: `Estimated country`
- L'RTT medio stimato per il paese rilevato: `Expected RTT`

Un valore di RTT osservato (`RTT_value`) viene considerato **anomalo** se non rientra nell'intervallo definito dalla media e dalla deviazione standard dei tempi di risposta noti per un determinato paese.

Formalmente, un `RTT_value` è considerato **valido** se:

$$\texttt{RTT\_MEAN} - k \cdot \texttt{DEVIAZIONE\_STANDARD} \leq \texttt{RTT\_value} \leq \texttt{RTT\_MEAN} + k \cdot \texttt{DEVIAZIONE\_STANDARD} \tag{2.1}$$

Nel nostro sistema, il fattore k è impostato a 2 (valore comunemente usato per

identificare valori anomali in una distribuzione normale). Pertanto, un RTT è accettabile se si trova **entro due deviazioni standard dalla media**.
Se il valore `RTT_value` non rientra in questo intervallo, viene etichettato come **potenziale anomalia** e segnalato come errore di protocollo in rosso su Wireshark. Questa segnalazione avviene anche in caso di incongruenza tra il **il paese sorgente misurato e il paese sorgente stimato**.

2.1.2 Gestione Paesi Sconosciuti e Calcolo del Continente

All'interno dello script `rtt_check.lua`, viene effettuato un confronto tra il paese rilevato per un determinato pacchetto e i dati salvati in memoria nel file `ntp_rtt_stats.txt`, generato dallo script Python.

Tuttavia, non tutti i paesi definiti nel database MaxMind sono presenti nel file di riferimento RTT, poiché il file è generato in modo selettivo. Quando ciò accade, lo script opera come segue:

1. Estrae le coordinate geografiche (latitudine e longitudine) del paese non presente in memoria.
2. Utilizza queste coordinate per determinare il continente di appartenenza.
3. Confronta l'RTT del pacchetto osservato con il valore medio stimato per quel continente.

I continenti sono definiti tramite intervalli di latitudine e longitudine minime e massime. Questi intervalli sono stati verificati attraverso gps-coordinates.net e uno script Python di testing, che analizza un set di coordinate geografiche note, rappresentandone i confini:

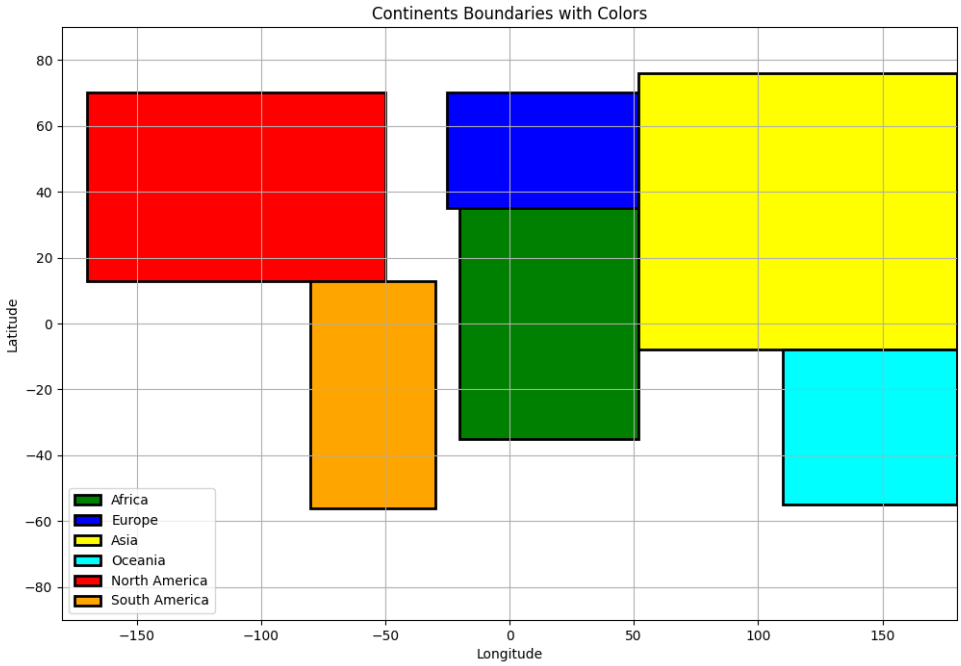

Figure 2.1: Risultato del test Python per la verifica delle coordinate dei continenti

Se si esegue generetor.py dall'italia, il risultato dovrebbe assomigliare ad una cosa di questo tipo:

Within Europe (EU & Nearby Countries)

- **Nearby EU countries (Germany, France, Spain, Netherlands, etc.):**
  - **Average RTT:** 20–50 ms
  - **Best-case (optimized routes):** < 20 ms (e.g., Milan to Frankfurt)
  - **Worst-case (congested paths):** 60–80 ms
- **Eastern Europe (Poland, Czech Republic, Hungary, etc.):**
  - **Average RTT:** 40–70 ms
- **UK & Ireland:**
  - **Average RTT:** 40–80 ms (depends on subsea cable routing)
- **Nordic Countries (Sweden, Norway, Finland):**
  - **Average RTT:** 50–90 ms

Outside Europe

- **North America (USA, Canada):**
  - **East Coast (New York, Washington):** 90–120 ms
  - **West Coast (Los Angeles, San Francisco):** 140–180 ms
- **South America (Brazil, Argentina):**
  - **Average RTT:** 180–250 ms
- **Middle East (UAE, Saudi Arabia, Israel):**
  - **Average RTT:** 80–120 ms
- **Asia:**
  - **India (Mumbai, Delhi):** 120–180 ms
  - **Singapore, Japan, South Korea:** 180–250 ms
  - **China (Beijing, Shanghai):** 200–300 ms (varies due to routing restrictions)
- **Africa:**
  - **North Africa (Egypt, Morocco):** 60–100 ms
  - **South Africa (Johannesburg, Cape Town):** 180–250 ms
- **Australia & New Zealand:**
  - **Average RTT:** 280–350 ms

Figure 2.2: RTT medio tra i paesi europei e l'estero

2.1.3 country.json

Il sistema sfrutta il file `country.json`, contenente le associazioni tra codici paese (es. `"US"`) e i relativi domini dei server NTP (es. `0.us.pool.ntp.org`). Questi domini vengono pingati per stimare il tempo medio di risposta.

2.1.4 parameters.json

Il file `parameters.json` contiene i parametri per eseguire i ping:

- `PING COUNT`: definisce quanti ping verranno eseguiti per ciascun host.
- `PING TIMEOUT`: è il tempo massimo di attesa per ogni ping prima di considerarlo fallito.
- `PING INTERVAL`: definisce quanto velocemente i ping vengono inviati dopo aver ottenuto un risultato.
- `OUTLIER FACTOR`: è usato per escludere i valori RTT che sono troppo lontani dalla media, regolando la tolleranza per i valori anomali.

2.1.5 ntp_rtt_stats.txt

Il file di output `ntp_rtt_stats.txt`, generato dal `generator.py`, contiene le medie degli RTT per ogni paese misurato. Questo viene poi letto dallo script Lua durante l'analisi dei pacchetti.

2.2 Installazione

Per iniziare con RTT GAD, segui questi passaggi:

1 **Installa il database GeoIP2 Country di MaxMind**
Scarica il database GeoLite2 Country (gratuito con registrazione) o un altro database GeoIP2 Country dal sito ufficiale di MaxMind.

Puoi anche seguire questo utile tutorial video di Chris Greer per maggiori informazioni:
YouTube – Come scaricare GeoLite2

2 **Installa il plugin Lua**
Sposta lo script `rtt_check.lua` nella directory dei plugin di Wireshark. Wireshark carica automaticamente i plugin Lua posizionati in queste directory a seconda del tuo sistema operativo:

- **Windows:** C:\Program Files\Wireshark\plugins\<version>\
- **macOS:** /Applications/Wireshark.app/Contents/PlugIns/wireshark/<version>/
- **Linux:** /usr/lib/wireshark/plugins/<version>/

Copia lo script Lua (`rtt_check.lua`) nella directory dei plugin di Wireshark. Se stai usando la directory dei plugin personali, assicurati che esista e crea la cartella plugins se necessario.

Metodo GUI (consigliato):
Apri Wireshark
Clicca su Aiuto → Info su Wireshark
Vai alla scheda Cartelle
Clicca sul link accanto a Plugin Personali
Sposta lo script `rtt_check.lua` in questa cartella

3 **Aggiungi il file di statistiche RTT**
Sposta il file `ntp_rtt_stats.txt` nella directory principale di Wireshark (lo stesso percorso in cui è installato l'eseguibile principale di Wireshark o dove si trova la cartella dei plugin).

4 **Riavvia Wireshark per caricare il plugin**

5 **Ora sei pronto per iniziare ad analizzare le anomalie RTT e rilevare incongruenze nella geolocalizzazione!**

2.3 Requirements

Per eseguire lo script `generator.py`, è necessario avere Python installato sul proprio sistema (**Testato su Python versione >= 3.12**). Lo script utilizza le seguenti librerie standard, che sono incluse nella libreria standard di Python.

1 Non sono richieste dipendenze esterne.

2 Questo plugin richiede Wireshark versione 3.6 o successiva.

Chapter 3

Testing

3.1 Tabella RTT utilizzati

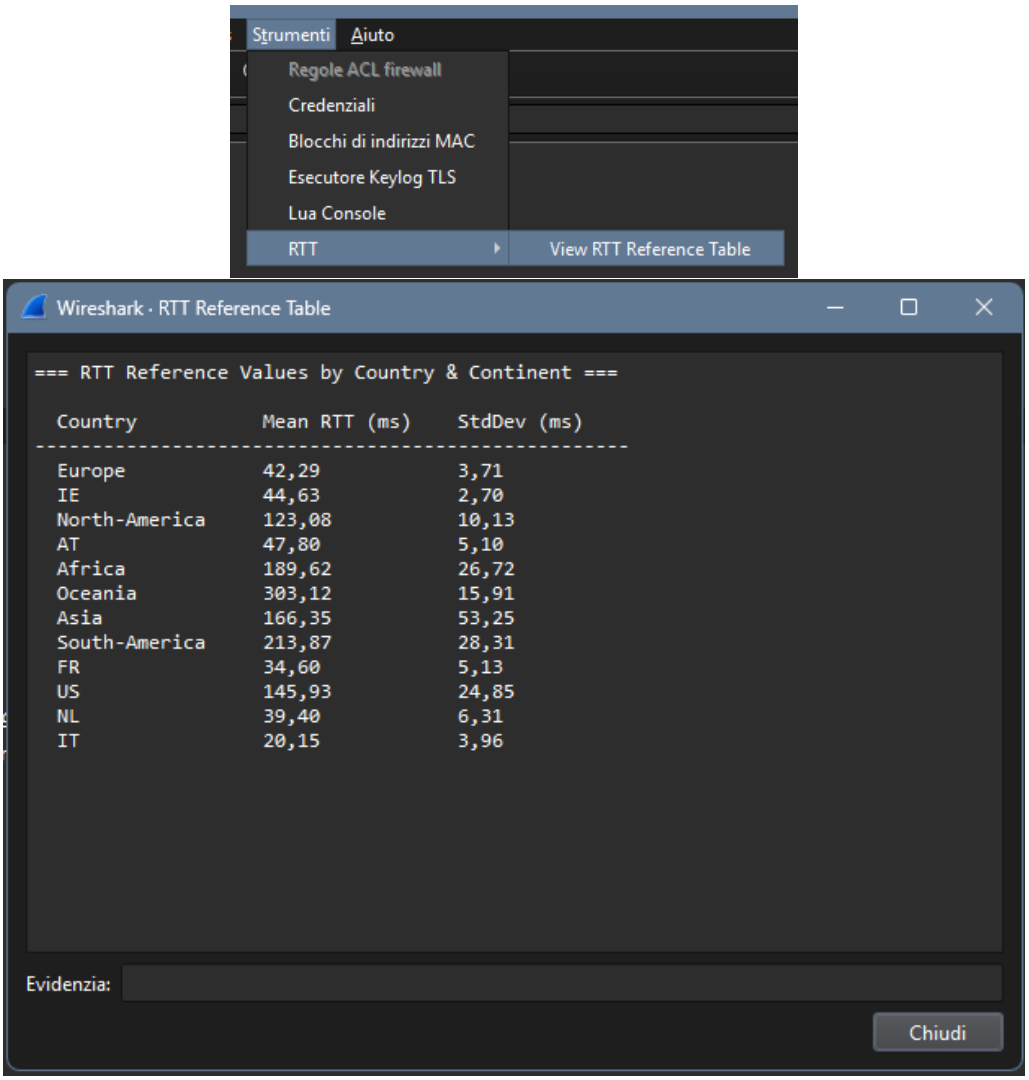

Figure 3.1: Tabella RTT dei vari paesi

3.2 Esempi pacchetti anomali

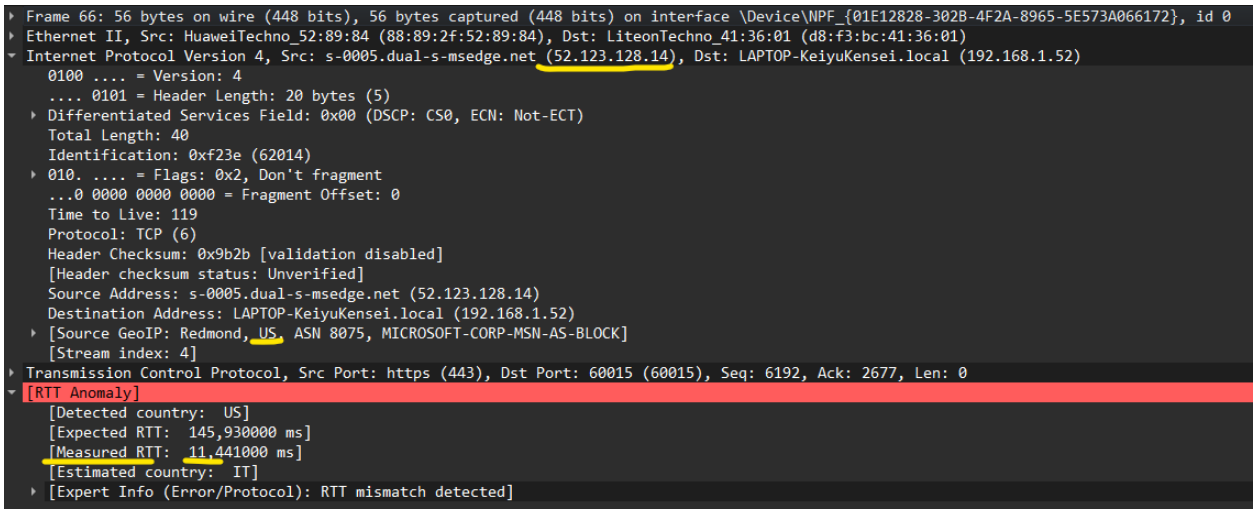

Figure 3.2: Esempio pacchetto TCP anomalo

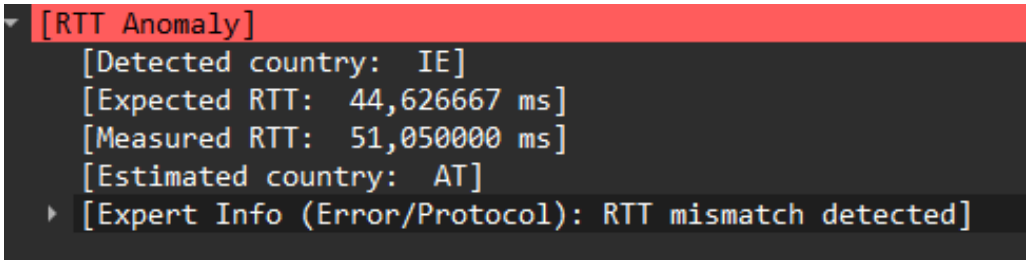

Figure 3.3: Probabile falso positivo 1

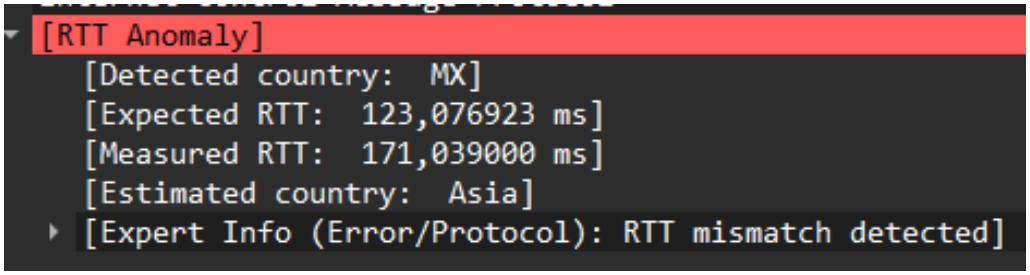

Figure 3.4: Probabile falso positivo 2