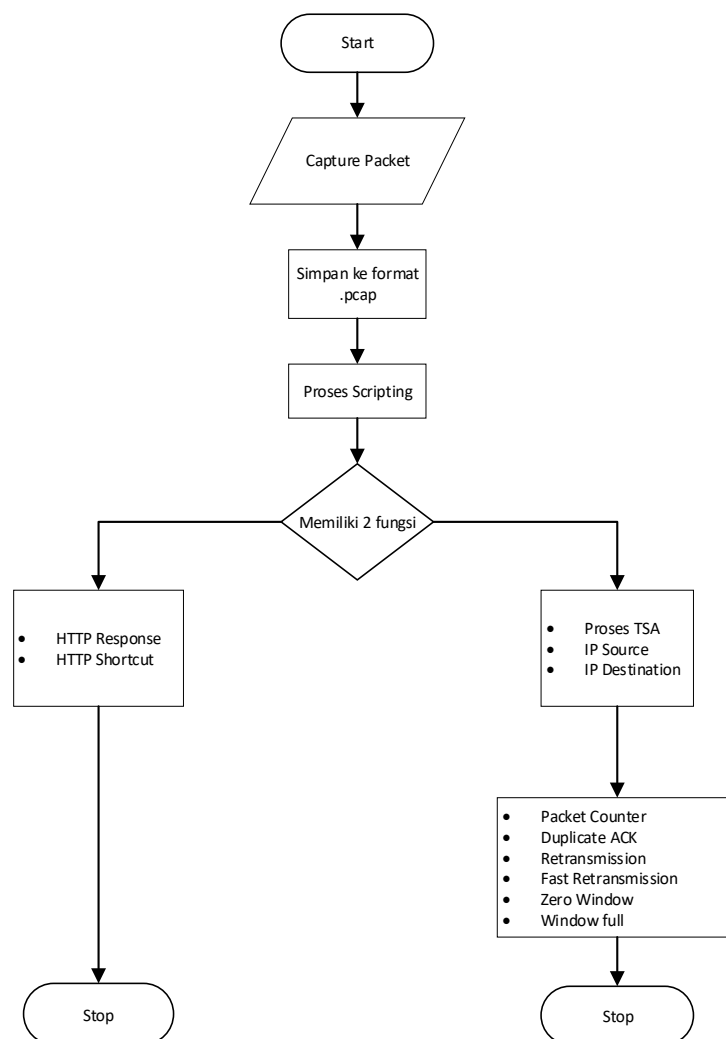


## BAB III

### METODE PENELITIAN

#### 3.1 Metode Analisis

Berdasarkan pengumpulan data dari studi literatur yang dilakukan terkait dengan *scripting* Lua pada aplikasi Wireshark, maka penulis mulai mencoba aplikasi tersebut di komputer dan mempersiapkan file hasil *capture* seperti format file pcap dan pcapng. Kegiatan tersebut memiliki tujuan untuk menyaring, dan mengetahui bagaimana cara kerja menggunakan *scripting* lua dan tanpa menggunakan *scripting* lua pada aplikasi Wireshark. Setelah mengetahui cara kerja dan karakteristik *scripting* Lua pada aplikasi tersebut dapat dilakukan pengajuan pembuatan kostumisasi untuk dikembangkan. Berikut adalah tahap alur pelaksanaan penelitian ditunjukkan pada Gambar 3.1.



Gambar 3.1 Alur penelitian.

### 3.2 Perangkat Yang Digunakan

Sebelum melakukan uji coba, penulis akan menyajikan pengantar secara singkat untuk menganalisis paket. Kemudian menjelaskan perangkat keras, perangkat lunak serta bagaimana analisis paket digunakan dalam suatu jaringan.

c. Perangkat Keras terdiri dari:

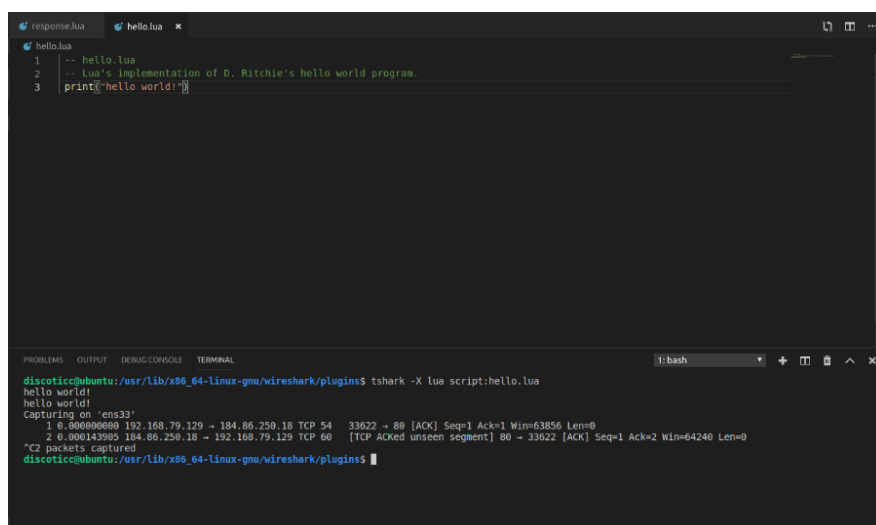
- 1) Laptop berbasis sistem operasi Windows 10 dengan Intel i5 Quad Core, RAM 8GB dan internal penyimpanan 750GB tipe HDD ukuran 2.5 inci, digunakan untuk melakukan penarikan data dan analisis secara mobilitas.
- 2) *Capture* paket menggunakan USB TP-Link TL-WN722N High-Gain.
- 3) Router dengan berbagai macam merek, sebagai tujuan untuk penarikan data.

d. Perangkat lunak terdiri dari:

- 1) VMware Workstation Pro dengan Sistem Operasi Ubuntu 19.04 (Disco Dingo) digunakan untuk melakukan Task di dalam mesin virtual, keuntungan aplikasi ini dapat dipindahkan di berbagai macam perangkat.
- 2) Wireshark dengan plug-in Lua.
- 3) Visual Studio Code sebagai tempat untuk edit *sourcecode*.

### 3.3 Scripting

Tahapan pengambilan data di atas bertujuan untuk menguji apakah aplikasi Wireshark dengan *scripting* Lua mampu mengambil dan menampilkan data dan aktivitas dari perangkat tujuan atau tidak. Berikut contoh cara kerja *scripting* Lua.



```

response.lua  hello.lua
hello.lua
1  -- hello.lua
2  -- Lua's implementation of D. Ritchie's hello world program.
3  print("hello world!")

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
discotic@ubuntu:/usr/lib/x86_64-linux-gnu/wireshark/plugins$ tshark -X lua script:hello.lua
hello world!
hello world!
Capturing on 'ens33'
1 0.000000000 192.168.79.129 -> 184.86.250.18 TCP 54 33622 -> 80 [ACK] Seq=1 Ack=1 Win=63856 Len=0
2 0.000143905 184.86.250.18 -> 192.168.79.129 TCP 66 [TCP ACKed unseen segment] 80 -> 33622 [ACK] Seq=1 Ack=2 Win=64240 Len=0
^C 0 packets captured
discotic@ubuntu:/usr/lib/x86_64-linux-gnu/wireshark/plugins$

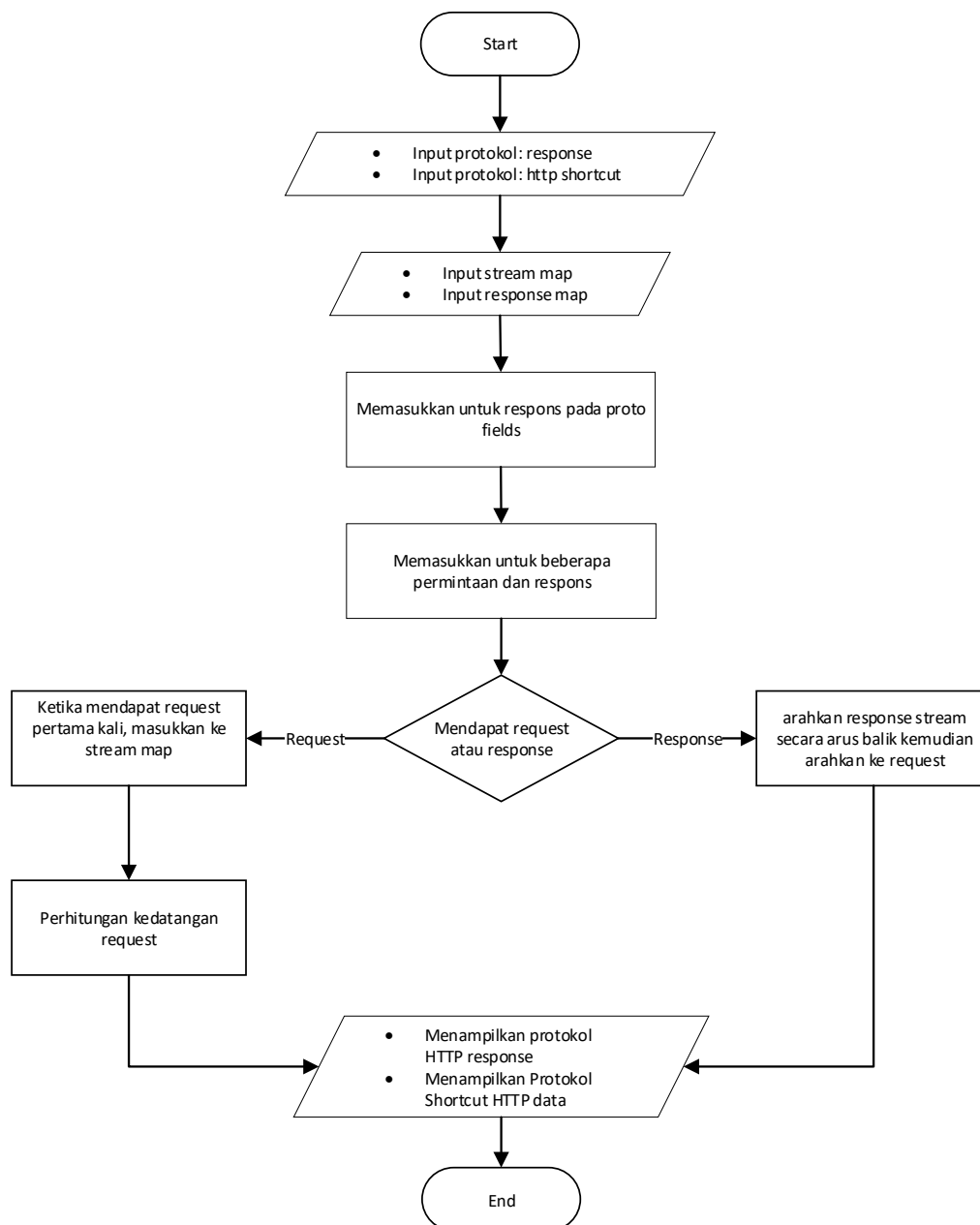
```

Gambar 3.2 *Scripting* sederhana pada Lua menggunakan Visual Studio Code.

Di aplikasi Visual Studio Code memiliki fitur *terminal*, digunakan untuk memudahkan eksekusi jika *scripting* tersebut mendukung *terminal* tshark.

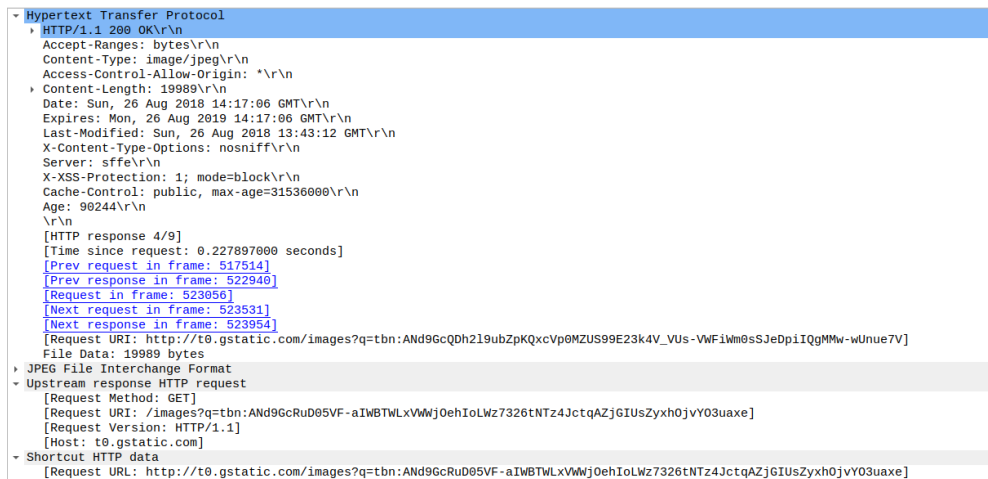
### 3.4 Request HTTP Data

Data permintaan pada protokol HTTP ditugaskan untuk mengambil paket data dalam bentuk halaman situs web, serta versi HTTP. Adapun algoritma untuk mengetahui cara kerja sebuah HTTP Request/Response pada Gambar 3.3.



Gambar 3.3 Flowchart Request HTTP Data.

Untuk eksekusi *request* data pada paket menggunakan aplikasi Wireshark terlebih dahulu. Sesuai gambar 3.3, eksekusi hasil paket akan muncul di panel detail paket. Panel Upstream response HTTP request dan Shortcut HTTP data berfungsi untuk mengambil detail penting dari Hypertext Transfer Protocol agar mudah diketahui pengguna secara langsung.



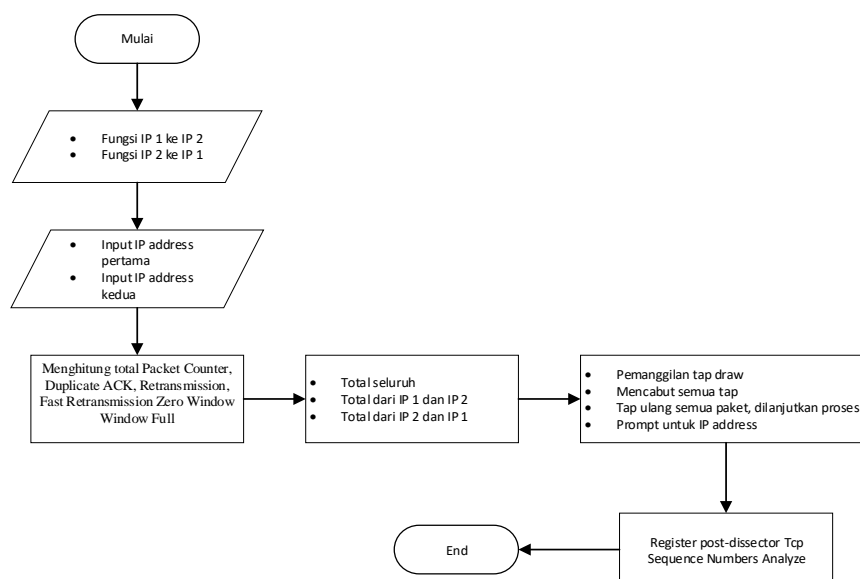
```

Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Accept-Ranges: bytes\r\n
  Content-Type: image/jpeg\r\n
  Access-Control-Allow-Origin: *\r\n
  Content-Length: 19989\r\n
  Date: Sun, 26 Aug 2018 14:17:06 GMT\r\n
  Expires: Mon, 26 Aug 2019 14:17:06 GMT\r\n
  Last-Modified: Sun, 26 Aug 2018 13:43:12 GMT\r\n
  X-Content-Type-Options: nosniff\r\n
  Server: sffe\r\n
  X-XSS-Protection: 1; mode=block\r\n
  Cache-Control: public, max-age=31536000\r\n
  Age: 90244\r\n
  \r\n
  [HTTP response 4/9]
  [Time since request: 0.227897000 seconds]
  [Prev request in frame: 517514]
  [Prev response in frame: 522940]
  [Request in frame: 523050]
  [Next request in frame: 523531]
  [Next response in frame: 523954]
  [Request URI: http://t0.gstatic.com/images?q=tbn:AND9GcQDh2l9ubZpKQxcVp0MZUS99E23k4V_VUS-VWf1Wm0sSJedpiIQgMMw-wUnue7V]
  File Data: 19989 bytes
  JPEG File Interchange Format
  Upstream response HTTP request
    [Request Method: GET]
    [Request URI: /images?q=tbn:AND9GcRu005VF-aIwBTWLxVWwj0ehIoLWz7326tNTz4JctqAZjGIUsZyxh0jvY03uaxe]
    [Request Version: HTTP/1.1]
    [Host: t0.gstatic.com]
  Shortcut HTTP data
    [Request URL: http://t0.gstatic.com/images?q=tbn:AND9GcRu005VF-aIwBTWLxVWwj0ehIoLWz7326tNTz4JctqAZjGIUsZyxh0jvY03uaxe]
  
```

Gambar 3.4 Hasil panel detail paket menggunakan *scripting* Lua pada Wireshark.

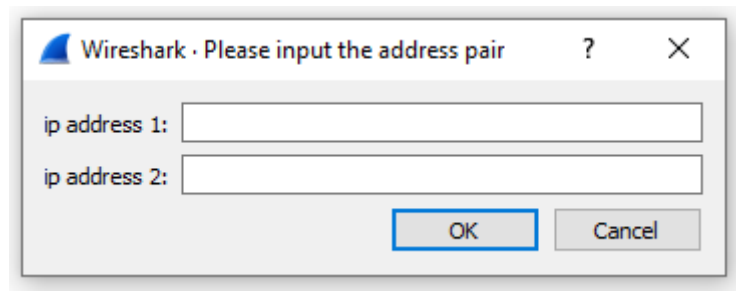
### 3.5 TCP Sequence Analyzer (TSA)

TSA berfungsi untuk menghitung jumlah paket dalam IP sumber ke IP destinasi, *add on* tersebut akan memproses dengan cara menghitung dan menampilkan angka yang sesuai dengan algoritma berikut ini:



Gambar 3.5 Flowchart TCP Sequence Analyzer (TSA).

Secara *default*, Wireshark dan Tshark akan melacak semua sesi TCP dan mengimplementasikan ke versi kasarnya. Hal ini membutuhkan beberapa informasi dan memori yang membutuhkan ekstra untuk disimpan oleh *dissector*. Tetapi memungkinkan deteksi yang lebih baik dari *event* TCP seperti transmisi ulang. Memungkinkan mengukur paket loss dan transmisi ulang yang jauh lebih baik dan lebih akurat daripada analisis protokol lainnya, tetapi itu masih belum sempurna.



Gambar 3.6 User Interface pada TCP Sequence Analyzer

Sesuai Gambar 3.6, fitur ini berfungsi menghitung jumlah paket yang akan dieksekusi, pada dasarnya tidak berdampak terlalu besar pada memori run-time dari Wireshark, serta dapat diperlukan jika melihat hasil paket TCP. Ketika fitur ini dijalankan berdasarkan input IP sumber ke IP destinasi, jendela TCP Sequence Analyzers akan berjalan menghitung jumlah dari seluruh paket yang telah direkam.