BAB IV
HASIL DAN PEMBAHASAN

Pada bagian ini menjelaskan hasil yang didapatkan selama penelitian yang telah dilakukan berdasarkan perumusan & tujuan penelitian, yaitu: 1) penerapan algoritma *density k-means* dalam mengelompokkan basisdata *log*; dan 2) analisis kerapatan (densitas) data pada studi kasus serangan *DoS.*

4.1. Analisis *Log* Menggunakan *Density K-Means*

a. *Log*

Pada bagian ini dijelaskan cara untuk mendapatkan *log* data lalu lintas jaringan dengan alat bantu tcpdump dengan parameter dijelaskan sebagai berikut:

`-i`: berfungsi untuk menangkap paket dari *interface ethernet* tertentu. (*ethernet*).

`-tttt`: berfiungsi untuk menangkap paket dan menampilkan *default* waktu (*timestamp*).

`-n`: berfungsi untuk menangkap paket dan menampilkan dalam format `IP` *address*.

`-q`: berfungsi untuk menangkap dan menampilkan informasi protokol.

`-e`: berfungsi untuk menangkap paket dan menampilkan *header* paket.

Hasil dari eksekusi perintah tersebut dihasilkan berkas *log* yang disimpan dalam bentuk teks. Alasan dipergunakan *tcpdump* sebagai alat untuk membuat *log* adalah, karena merupakan alat yang paling umum digunakan dalam *troubleshooting* jaringan. Untuk membuka berkas *log* yang masih berbentuk teks dengan ukuran besar memerlukan penanganan tersendiri, hal ini disebabkan karena teks editor yang ada tidak dapat memproses berkas *log* tersebut. Untuk menangani hal tersebut, skrip *regex* dimanfaatkan agar dapat mencari & mencocokkan pola yang dibutuhkan

dan kemudian disimpan dalam basisdata *log*. Skrip untuk melakukan ekstraksi data dari berkas *log* ditunjukkan pada **gambar 4.1.**

```
#!/bin/bash
host=localhost
user=root
pass=qwerty
db=mikro
#ifconfig eth0 up
cat 250815.log | grep IPv4 | awk -F "[:,> ]"
{'print $1 " "$2":"$3":"$4" "$5":"$6":"$7":"$8":"$9":"$10"
"$13":"$14":"$15":"$16":"$17":"$18" "$25" "$28" "$30" "$23" "$31'}
| awk -F "[. ]" '{ if ($25=="")
print "INSERT IGNORE INTO paket_mikro
(`timestamps`, `mac_add_sbr`, `mac_add_tuj`, `ip_port_add_sbr`, `ip_port_add_tuj`,
protokol`, `length`, `tcplength`)
values
(\""$1" "$2"\", \""$4"\", \""$5"\", \""$6" "$7" "$8" "$9" "$10"\",
\""$11" "12" "$13" "$14" "$15"\", \""$16"\", \""$17"\", \""$18"\");"}'
| mysql -u$user -p$pass $db
```

Gambar 4.1. Skrip Regex

Penjelasan dari masing-masing blok perintah adalah sebagai berikut:

- Baris 1, menjelaskan jenis *shell* yang digunakan dalam melakukan adalah *shell bash*.
- Baris 2 - Baris 5, menjelaskan variabel untuk melakukan hubungan dengan basisdata, basisdata yang digunakan pada penelitian ini menggunakan *Database Management System* (DBMS) `MySQL`.
- Baris 7 - Baris 8, menjelaskan proses untuk menampilkan data yang dicari untuk kemudian diekstrak, *output* dari baris tersebut menampilkan informasi *timestamps*, alamat *mac*, alamat *ip* dan *port*, protokol, ukuran paket.
- Baris 10 - Baris 11, merupakan lanjutan proses yang diawal menampilkan kriteria informasi yang diinginkan untuk selanjutnya disimpan dalam basisdata dengan menggunakan 1 (satu) tabel dengan *field* yang telah dibuat sebelumnya.

Tujuan dari penggunaan skrip regex tersebut adalah agar data yang tersimpan dalam berkas *log* dapat dimasukan pada basisdata secara otomatis. Hasil dari eksekusi skrip regex tersebut adalah sebuah basisdata *log*. Basisdata *log* tersebut memiliki 9 (sembilan) atribut dengan banyak

data sebanyak 11.358.001 *record*, ke-sembilan atribut tersebut yaitu: *id*, *timestamps*, *mac_add_sbr*, *mac_add_tuj*, *ip_port_add_sbr*, *ip_port_add_tuj*, *protokol*, *length* & *tcplength*. **gambar 4.2.** menunjukan potongan basisdata *log*.

id	timestamps	mac_add_sbr	mac_add_tuj	ip_port_add_sbr	ip_port_add_tuj	protokol	length	tcplength
1	2015-08-25 22:30:51	18:03:73:8b:8e:e2	01:00:5e:00:00:fb	192 168 0 248 5353	224 12 0 251 5353	UDP	87	0
2	2015-08-25 22:31:35	d4:ca:6d:5a:d8:39	ff:ff:ff:ff:ff:ff	192 168 0 1 5678	255 12 255 255 5678	UDP	134	0
3	2015-08-25 22:31:39	9c:b7:0d:a8:66:06	01:00:5e:00:00:16	192 168 0 246 224	0 12 22 igmp 60		0	0
4	2015-08-25 22:32:08	d0:df:9a:19:b1:e0	ff:ff:ff:ff:ff:ff	0 0 0 0 68	255 12 255 255 67	UDP	354	0
5	2015-08-25 22:32:08	d0:df:9a:19:b1:e0	01:00:5e:00:00:16	192 168 0 254 224	0 12 22 igmp 60		0	0

id	timestamps	mac_add_sbr	mac_add_tuj	ip_port_add_sbr	ip_port_add_tuj	protokol	length	tcplength
5000000	2015-09-11 23:01:03	d0:df:9a:19:b1:e0	18:03:73:8b:8e:e2	192 168 0 254 49194	192 12 0 248 42811	tcp	60	0
5000001	2015-09-11 23:01:03	18:03:73:8b:8e:e2	d0:df:9a:19:b1:e0	192 168 0 248 42811	192 12 0 254 49194	tcp	1514	1460
5000002	2015-09-11 23:01:03	18:03:73:8b:8e:e2	d0:df:9a:19:b1:e0	192 168 0 248 42811	192 12 0 254 49194	tcp	1514	1460
5000003	2015-09-11 23:01:03	d0:df:9a:19:b1:e0	18:03:73:8b:8e:e2	192 168 0 254 49194	192 12 0 248 42811	tcp	60	0
5000004	2015-09-11 23:01:03	18:03:73:8b:8e:e2	d0:df:9a:19:b1:e0	192 168 0 248 42811	192 12 0 254 49194	tcp	1514	1460

id	timestamps	mac_add_sbr	mac_add_tuj	ip_port_add_sbr	ip_port_add_tuj	protokol	length	tcplength
358001	2015-09-11 23:21:58	d4:ca:6d:5a:d8:39	ff:ff:ff:ff:ff:ff	192 168 0 1 5678	255 12 255 255 5678	UDP	134	0
358000	2015-09-11 23:20:58	d4:ca:6d:5a:d8:39	ff:ff:ff:ff:ff:ff	192 168 0 1 5678	255 12 255 255 5678	UDP	134	0
357999	2015-09-11 23:20:44	b8:76:3f:a5:3c:bb	18:03:73:8b:8e:e2	192 168 0 253 44789	192 12 0 248 21	tcp	60	0
357998	2015-09-11 23:20:44	18:03:73:8b:8e:e2	b8:76:3f:a5:3c:bb	192 168 0 248 21	192 12 0 253 44789	tcp	68	14
357997	2015-09-11 23:20:44	b8:76:3f:a5:3c:bb	18:03:73:8b:8e:e2	192 168 0 253 44794	192 12 0 248 21	tcp	60	0

Gambar 4.2. Potongan Basisdata log

b. Klasifikasi Tingkat Bahaya

Pada bagian ini menjelaskan klasifikasi tingkat bahaya serangan *denial of service* berdasarkan kedua atribut *length* & *tcplength* yang dikumpulkan berdasarkan jam tercatatnya paket data lalu lintas jaringan.

Untuk memudahkan dalam mengenali tingkat bahaya diperlukan parameter yang digunakan sebagai pembanding atas data pada proses *clustering*. Dalam penelitian ini ditentukan bahwa tingkat bahaya dapat dievaluasi berdasarkan nilai terkecil, nilai tengah dan nilai terbesar dari total kedua atribut. Klasifikasi tingkat bahaya ditunjukan pada **tabel 4.1.**

Tabel 4.1. Klasifikasi Tingkat Bahaya

	Tingkat Bahaya	***Totlength***	***Tottcplength***
Min	Rendah	2619	0
Median	Sedang	3685.5	2
Max	Tinggi	10104895	4971517

Tujuan penggunaan nilai terkecil, nilai tengah dan nilai terbesar adalah sebagai acuan untuk membandingkan hasil yang didapatkan pada proses

clustering, nilai tersebut digunakan sebagai inisiasi awal dalam menentukan *centroid* diawal proses *clustering* sebelum dilakukan perhitungan jarak untuk semua data.

c. *Density K-Means*

Pada bagian ini akan menjelaskan langkah-langkah *clustering* menggunakan algoritma *density k-means*, algoritma ini pada dasarnya adalah algoritma *K-means* dengan penambahan fungsi untuk melakukan pengujian densitas data dengan hasil berupa nilai index. algoritma ini membutuhkan 2 (dua) nilai parameter diawal proses. Pertama menentukan jumlah *cluster* yang dibutuhkan. Tujuan dalam penelitian ini adalah mengelompokan tingkat bahaya yang dibagi menjadi 3 tingkat yaitu rendah, sedang & tinggi, sehingga untuk menghasilkan *cluster* sebanyak tiga maka ditentukan jumlah $k = 3$. Kedua adalah inisiasi *centroid* diawal, tujuan dilakukan inisiasi *centroid* adalah sebagai acuan dalam melakukan perhitungan jarak antar data. Inisiasi *centroid* diawal ditetapkan menggunakan nilai terkecil, nilai tengah dan nilai terbesar, lihat **tabel 4.1 Klasifikasi tingkat bahaya**. Setelah kedua nilai parameter ditetapkan. Langkah selanjutnya dari algoritma *density k-means* adalah melakukan perhitungan jarak untuk semua data berdasarkan *centroid* yang telah ditetapkan sebelumnya. Hasilnya adalah data yang memiliki jarak minimum dengan *centroid*-nya akan dimasukan pada *cluster*.

Langkah berikutnya adalah melakukan perhitungan *centroid* baru berdasarkan jarak minimum dari proses perhitungan jarak pada iterasi sebelumnya (iterasi 1), bila ditemukan kondisi dimana nilai *centroid* baru sama dengan nilai *centroid* lama, maka proses *clustering* dengan menggunakan algoritma *density k-means* telah selesai, namun bila ditemukan kondisi dimana nilai *centroid* baru tidak sama dengan nilai *centroid* lama, maka dilakukan proses perhitungan jarak hingga mendapatkan kondisi nilai *centroid* baru sama dengan *centroid* lama.

Proses perhitungan jarak menggunakan persamaan jarak *euclidean*, tujuan dari penggunaan persamaan jarak tersebut adalah karena persamaan ini paling banyak digunakan pada proses *clustering*. Persamaan jarak *euclidean* ditunjukan pada **persamaan 4.1.**

$$D(\chi,y) = \|\chi - y\|_2 = \sqrt{\sum_{j=1}^{N} |x - y|^2} \quad \text{.....} \quad (4.1)$$

perhitungan jarak berdasarkan persamaan 4.1. untuk tiap iterasi disertakan pada lampiran.

d. *Davies-Bouldin Index*

Nilai *Davies-bouldin index* berada pada interval (0, 1), dimana 0 menunjukan jarak minimum (*densitas*) antar data pada *cluster* & 1 menunjukan jarak maksimal (*separate*) antar *cluster*. nilai *Davies-Bouldin Index* yang didapatkan pada penelitian ini adalah 0.082, nilai tersebut menunjukan *cluster* optimal.

4.2. Hasil *Clustering Density K-Means*

Pada bagian ini menjelaskan hasil klasifikasi setelah dilakukannya *clustering* dengan menggunakan algoritma *density k-means*. Hasilnya ditunjukan pada **tabel 4.2**. Terdapat 2 (dua) jenis tingkat bahaya yaitu sedang dan tinggi. Dimana pada jam ke 0 hingga 22 digolongkan dalam *cluster* tingkat bahaya sedang, Pada kolom *tottcplength* terdapat nilai 0 yang dikategorikan sebagai *cluster* dengan tingkat bahaya sedang, hal tersebut disebabkan karena nilai 0 tidak berisi data apapun yang ditemukan pada protokol TCP berdasarkan kriteria < 10, < 100, < 1000 dan > 1000. Sehingga proses klasifikasi menggunakan kolom totlength. sedangkan pada jam ke 23 digolongkan dalam *cluster* dengan tingkat bahaya tinggi disebabkan karena ditemukannya data yang melalui protokol TCP berdasarkan kriteria diatas.

Tabel 4.2. Klasifikasi Tingkat Bahaya

Jam ke-*i*	***totlength***	***tottcplength***	**Klasifikasi**
0	3404	2	SEDANG
1	2619	1	SEDANG
2	3862	1	SEDANG
3	2948	0	SEDANG
4	3871	1	SEDANG
5	4622	0	SEDANG
6	3163	0	SEDANG
7	3253	2	SEDANG
8	2911	1	SEDANG
9	2648	0	SEDANG
10	3376	12	SEDANG
11	2692	0	SEDANG
12	3020	0	SEDANG
13	3658	0	SEDANG
14	9809	2688	SEDANG
15	26871	6905	SEDANG
16	983613	355522	SEDANG
17	3973	100	SEDANG
18	3713	84	SEDANG
19	4114	90	SEDANG
20	4029	69	SEDANG
21	3416	84	SEDANG
22	160322	85971	SEDANG
23	10104895	4971517	TINGGI

Density K-means menunjukan tingkat kerapatan data dengan mengevaluasi nilai tiap *SSW*, dengan hasil menunjukan bahwa ada serangan dalam layanan, hal ini ditunjukan dengan tingkat densitas data dengan *centroid*-nya dengan jarak bernilai 0.

4.3. Skenario Pengujian Serangan *Denial of Service*

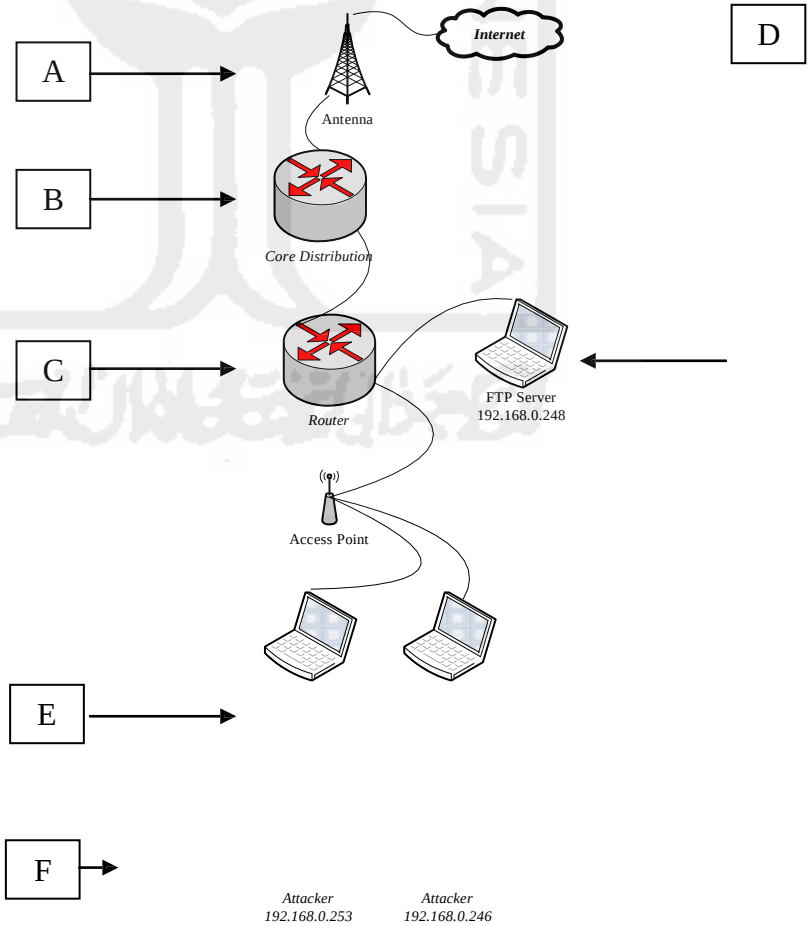
Pengujian dilakukan dari jaringan lokal UKM Mandala Citra Media di Surakarata, dimana penyerang menggunakan alat bantu LOIC (*Low Orbit Ion Canon*). Skenario pengujian ditunjukan pada **gambar 4.3.** pada gambar fungsi masing-masing dijelaskan sebagai berikut:

a. *Antenna*

Berfungsi sebagai pemancar bagi pengguna-pengguna Layanan Internet SOHO UKM Mandala Citra Media Surakarta.

b. *Core Distribution*

Berfungsi sebagai *gateway* utama yang berinteraksi langsung dengan jaringan internet dan memberikan akses layanan internet bagi perangkat lain dalam Virtual LAN.



Gambar 4.3. Topologi Pengujian

c. *Router*

Berfungsi sebagai *Gateway*, *router* ini tidak secara langsung berinteraksi dengan internet melainkan melalui *Core Distribution* untuk dapat mengakses internet.

d. *Access Point*

Berfungsi dalam menyediakan layanan internet menggunakan gelombang radio bagi perangkat pengguna.

e. *Victim*

Berfungsi sebagai penyedia layanan transfer berkas (`FTP Server`) yang menggunakan *port* 21 sebagai saluran dalam melakukan transaksi.

f. *Attacker*

Berperan sebagai penyerang yang melakukan serangan *denial of service* dengan jenis serangan *flooding* dan *ip spoffing* pada layanan transfer berkas.

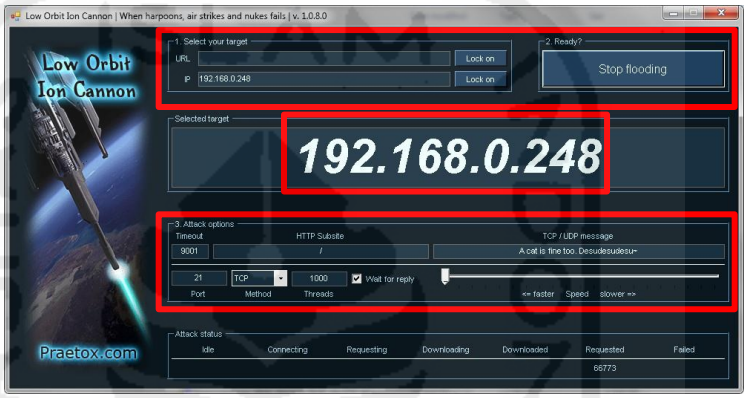
Korban dengan *ip address* 192.168.0.248 yang terhubung dengan jaringan melalui sebuah *router*, merupakan sebuah FTP server yang disiapkan dengan kondisi *default*. Kemudian ada beberapa *attacker* yang melakukan serangan melalui jaringan lokal. Pengujian serangan ini menggunakan alat, yaitu:

a. LOIC (*Low Orbit Ion Canon*) (Technologies, 2014)

Alat alternatif yang digunakan untuk menguji serangan pada *victim*. Alat ini memiliki kelebihan dapat melakukan pengiriman paket *request* berdasarkan protokol *tcp* maupun *udp*. Selain itu target *port* yang akan dikirim dapat ditentukan oleh penyerang. Dalam pengujian ini, LOIC digunakan untuk melakukan serangan ke *port* 21.

Alasan penggunaan *Port* tersebut sebagai target adalah karena port tersebut merupakan *port* yang sering digunakan oleh pengguna untuk mengakses informasi menggunakan jaringan internet maupun lokal. Proses pengujian serangan dilakukan dengan memasukan alamat *ip victim* pada aplikasi LOIC dari mesin *attacker* pada menu 1 (satu) atau *Select*

your target, kemudian tetapkan alamat yang akan diserang dengan menggunakan tombol *lock on* yang berada pada menu 1 (satu), selanjutnya menentukan target *port* adalah 21, target protokol adalah TCP, jumlah *threads* yang akan dikirimkan sebanyak 1000, dan kecepatan pengiriman paket pada tingkat *faster* pada menu 3 (tiga). Aplikasi LOIC ditunjukan gambar 4.4.



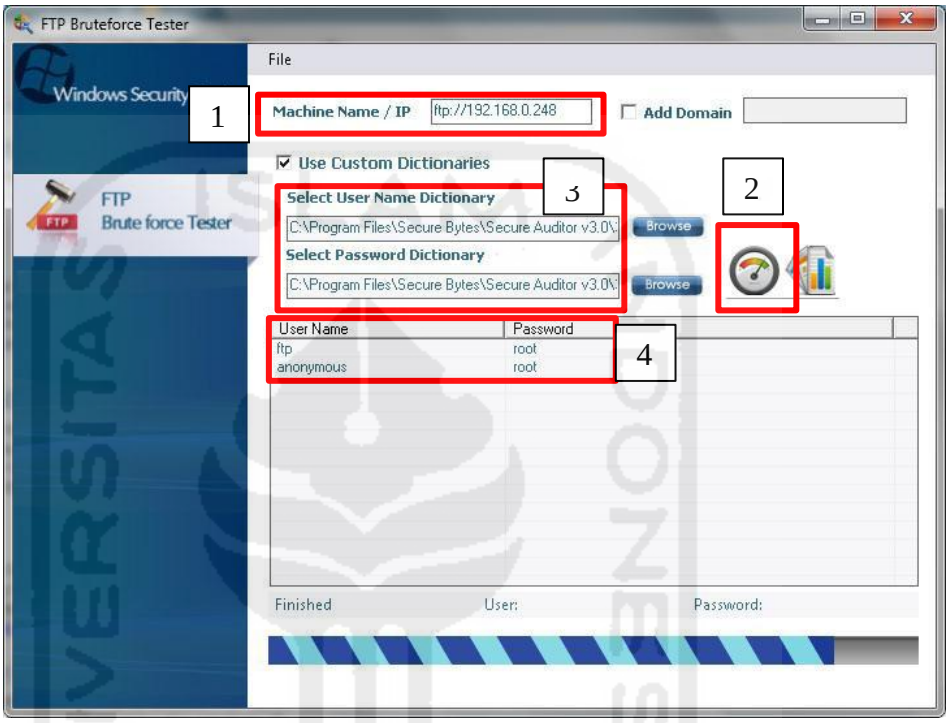
Gambar 4.4. LOIC

Setelah semua konfigurasi dimasukan pada aplikasi LOIC kemudian melakukan serangan dengan menekan tombol *start flooding* untuk memulai atau *stop flooding* untuk menghentikan serangan pada menu 2 (dua).

b. FTP BruteForce (Bytes, 2015)

Alat yang menggunakan metode *bruteforce* dalam mendapatkan informasi pengguna seperti *username* & *password* dalam mengakses ftp server, kelebihan alat ini karena kemampuannya dalam mengenali kelemahan yang dimiliki oleh ftp server, alat ini bekerja dengan menguji semua kombinasi *username* & *password* yang umum digunakan, dalam pengujian ini FTP *BruteForce* menggunakan *port* 443. *Port* tersebut merupakan *port* yang sering digunakan oleh pengguna untuk mengakses informasi menggunakan jaringan internet maupun lokal secara aman. Proses pengujian serangan dilakukan dengan memasukan alamat *ip victim* pada aplikasi FTP *Bruteforce* dari mesin *attacker*, yang kemudian untuk

melancarkan serangan *ip spoofed* dilakukan dengan menekan tombol yang ditunjukan oleh label 2 (dua). Aplikasi FTP *Bruteforce* ditunjukan gambar 4.5



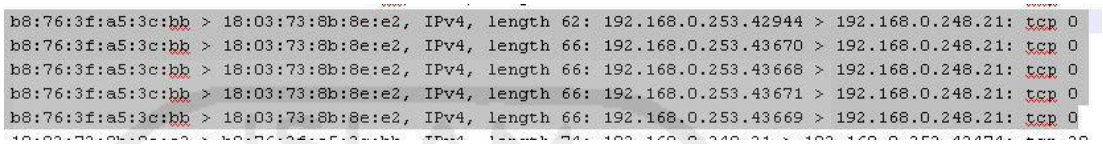
Gambar 4.5. FTP BruteForce

Aplikasi *FTPBruteforce* akan melakukan pemeriksaan segala kombinasi username dan password yang digunakan dalam mengakses FTP server dengan menggunakan basisdata bawaan aplikasi *FTPBruteforce*, ditunjukan pada label 3 (tiga) adalah alamat *dictionary* bawaan aplikasi tersebut, hasil dari semua pengujian yang dilakukan oleh aplikasi ini ditunjukan pada label 4 (empat), dimana *username* & *password* yang digunakan dan dapat dikenali oleh aplikasi ini adalah *username ftp* menggunakan *password root* serta *username anonymous* menggunakan *password* yang sama yaitu *root.*

4.4. Verifikasi *log* untuk *DoS Port* 21 (FTP) dengan LOIC

Berdasarkan hasil yang sudah didapatkan dari target serangan, maka langkah selanjutnya adalah melakukan verifikasi informasi dari proses analisis

basisdata *log* dengan *log* asli yang sudah tersimpan dalam berkas teks. Untuk serangan *DoS* pada *port* 21 dengan LOIC dapat diketahui hasilnya bahwa telah terjadi serangan pada sistem melalui *port* 21 dengan panjang *header* 66 seperti terlihat pada **gambar 4.6**.

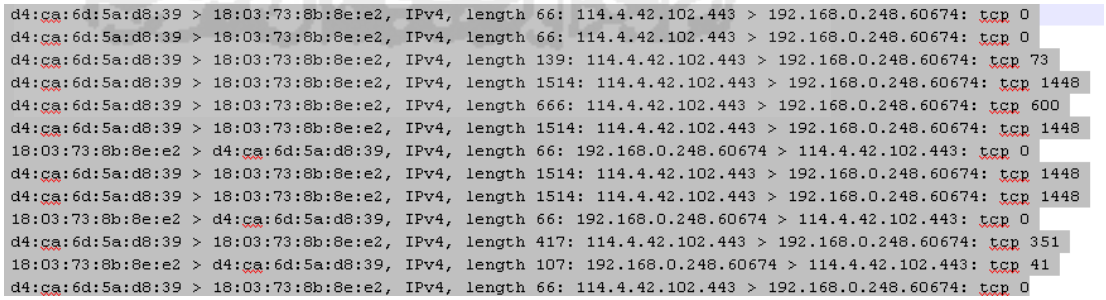


Gambar 4.6. Verifikasi log asli

Penyerang dengan IP 192.168.0.253 mengirimkan data dengan *length* 66 *bytes* ke target dengan IP 192.168.0.253:21 (FTP server), pengiriman data tersebut dilakukan tanpa henti untuk membebani beban target maupun jaringan yang menyebabkan sumberdaya yang dimiliki dialokasikan untuk melayani permintaan dari penyerang.

4.5. Verifikasi *log* untuk *DoS* dengan FTP BruteForce

Berdasarkan hasil yang sudah didapatkan dari mesin korban, maka langkah selanjutnya adalah melakukan verifikasi informasi dari proses analisis basisdata *log* dengan *log* asli yang sudah tersimpan dalam berkas teks. Untuk serangan *DoS* dengan FTP *BruteForce* dapat diketahui hasilnya bahwa pada telah terjadi serangan pada sistem melalui *port* 443 dengan panjang *header* 66 dan tcplength 0 namun asal alamat IP yang didapatkan tidak sama dengan saat serangan menggunakan LOIC seperti terlihat pada **gambar 4.7**.



Gambar 4.7. Verifikasi log asli

Berdasarkan hasil verifikasi yang dilakukan dengan cara melakukan verifikasi informasi hasil analisis dengan berkas *log* asli dalam bentuk berkas teks, dapat diketahui dan dipastikan bahwa telah terjadi serangan *DoS*.