Perancangan *Bot* Untuk *Monitoring Server* Dari Serangan *Distributed Denial Of Service* Dan Implementasi *JSON Web Token* Pada Sistem Notifikasi Serangan

Arif Maulana Rahman [1], Henki Bayu Seta [2], Ria Astriratma [3]
Program Studi Informatika / Fakultas Ilmu Komputer
Universitas Pembangunan Nasional Veteran Jakarta
Jl. RS. Fatmawati, Pondok Labu, Jakarta Selatan, DKI Jakarta, 12460, Indonesia.
arifmr091@gmail.com[1], henkiseta@upnvj.ac.id[2], astriratma@upnvj.ac.id[3]

Abstrak. Untuk mengamankan sebuah server diperlukan sistem *monitoring* agar meminimalisir resiko jika terjadi percobaan intrusi. Salah satu serangan yang dapat mengancam server adalah serangan *DDOS* (*Distributed Denial of Service*) yang mengakibatkan sistem yang diserang mengalami gangguan berupa error request, halt, kegagalan sistem dan sebagainya. Berdasarkan permasalahan tersebut, diperlukan sebuah *bot* untuk melakukan *monitoring server* dari serangan *DDOS*, dalam penelitian ini difokuskan serangan *DDOS* berjenis *UDP Flooding* dan *SYN Flooding*. Solusi yang diberikan dalam penelitian ini adalah dengan membuat sebuah *bot* untuk melakukan *monitoring* terhadap server dari serangan *DDOS* berjenis *UDP Flooding* dan *SYN Flooding* dengan cara membatasi paket yang dapat diterima oleh *port* yang dibuka oleh server sebanyak 100 paket per detik. Diperoleh waktu rata-rata yang dibutuhkan untuk mendeteksi serangan hingga muncul notifikasi adalah 3.532 detik.

Kata Kunci: Keamanan, Server, *Monitoring*, *Distributed Denial of Service*, *Bot*, *UDP Flooding*, *SYN Flooding*, *JSON Web Token*.

1 Pendahuluan

Di era teknologi informasi seperti sekarang, menjaga keamanan server menjadi hal yang sangat penting agar dapat bersaing. Banyak ancaman yang dapat digunakan untuk mengganggu keamanan server. Pada saat ini serangan luar yang diluncurkan oleh penyerang semakin bervariasi, contohnya adalah serangan *DDOS* (*Distributed Denial of Service*) [1]. Ada beberapa jenis serangan *Distributed Denial of Service* yang sering terjadi seperti *UDP Flooding* dan *SYN Flooding*. Serangan *DDOS* mengakibatkan sistem yang diserang mengalami gangguan berupa *error request*, *halt*, kegagalan sistem dan sebagainya. Oleh karena itu, diperlukan *monitoring* untuk mengamankan jaringan agar dapat mengurangi dampak yang terjadi jika terjadi percobaan intrusi.

Sistem *monitoring* adalah sistem yang dirancang agar dapat memberikan respon saat menjalankan program. Respon yang ditujukan agar dapat memberikan informasi keadaan sistem. Sistem *monitoring* terdiri dari mekanisme serta program dengan menjalankan sistem informasi pada komputer yang dirancang untuk mendata serta mampu untuk mengirimkan data sesuai dengan data yang diperoleh [2]. Dalam merancang sistem *monitoring*, diperlukan sebuah *bot* agar memudahkan admin server dalam melindungi server.

JSON Web Token (*JWT*) merupakan token berupa *string* yang *random* berguna dalam mengoperasikan sistem autentikasi serta untuk mengamankan dalam bertukar informasi [3]. *JSON Web Token* menjaga keamanan data dengan cara melakukan *encode* pada klaim untuk diubah ke *JSON* serta mengubah *JSON Web Signature* ke *payload*. Berdasarkan penjelasan tersebut, dilakukan penelitian untuk mengamankan server dari serangan *DDOS* berjenis *UDP Flooding* dan *SYN Flooding* dengan memanfaatkan bot untuk *monitoring* server menggunakan *JSON Web Token*.

1.1 Rumusan Masalah

1. Bagaimana cara *bot* mendeteksi serangan *DDOS* berjenis *UDP Flooding* dan *SYN Flooding* dalam

monitoring server?
2. Berapa lama waktu yang dibutuhkan *bot* dari mendeteksi serangan hingga admin menerima notifikasi serangan?

2 Persiapan Naskah

2.1 *DDOS (Distributed Denial of Service)*

Merupakan serbuan *DOS* secara bersamaan dengan kuantitas penyerang lebih dari satu serta mengincar satu sasaran [4]. *DDOS* bisa dijalankan melalui dua cara, yang pertama dengan mengoperasikan *bot* yang diatur penyerang agar dapat menjalankan suatu serangan yang terorganisasi yang dijalankan oleh sekelompok orang kepada target. Terdapat beberapa jenis serangan *DDOS*, diantaranya adalah *UDP Flood* dan *SYN Flood*.

2.1.1 *UDP Flooding*

User Datagram Protocol (*UDP) flooding* merupakan sebuah serangan yang membanjiri *port* acak pada target dengan jumlah yang besar menggunakan paket *IP* yang mengandung *UDP*. Serangan ini mengakibatkan server yang diserang mengalami gangguan seperti disebabkan banyaknya paket yang data diterima server [5].

2.1.2 *SYN Flooding*

SYN flooding merupakan serangan yang mengeksploitasi *loophole* ketika koneksi *TCP/IP* terjadi dengan membajiri server target dengan paket *SYN*, ketika server menerima paket *SYN* server akan mengirimkan paket *SYN-ACK* kepada pengirim [6]. Dengan keadaan server yang sedang dibanjiri paket *SYN* akan menyebabkan server tidak dapat diakses.

2.2 *JSON WEB Token*

JSON Web Token merupakan bentuk standar agar dapat menjaga keamanan informasi ke bentuk suatu klaim yang kemudian di *encode* menjadi format *JSON* serta mengubah *JSON Web Signature* menjadi *payload*. Klaim tersebut dilindungi dengan tanda tangan digital yang telah dienkripsi [3].

2.3 Algoritma *HMAC*

Algoritma *Keyed-Hash Message Authentication Code* (*HMAC*) merupakan metode yang digunakan untuk melakukan autentikasi kepada data yang ditransmisikan pada kedua belah pihak dengan menggunakan fungsi *hash* kepada data tersebut dengan sebuah *secret key* yang diketahui oleh kedua belah pihak [7].

$$HMACk(m) = h((K \oplus opad) \square\ h\ ((K \oplus ipad)\ m)) \square$$

Keterangan :
HMACk(m) : nilai *HMAC* pada klaim untuk diautentikasi.
H : fungsi *hash*
K : *secret key*
opad : *Outer pad* (0x5c5c..5c)
ipad : *Inner pad* (0x36363..36)
m : klaim untuk diautentikasi

ISSN : 2655-139X (ONLINE)
ISSN : 0216-4221 (PRINT)

2.4 Algoritma *SHA256*

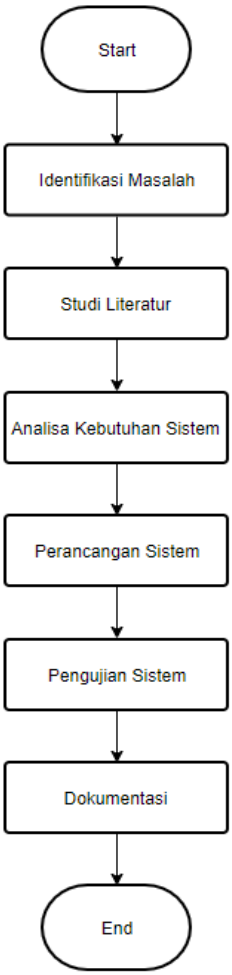
Algoritma *Secure Hash Algorithm 256* (*SHA256*) merupakan sebuah operasi *hash* yang memiliki ukuran 256-bit yang berfungsi untuk mengubah klaim dengan pemetaan *string* dengan panjang acak menjadi teks khusus yang disebut *message digest* [8]. *SHA256* mengimplementasikan 6 *logic function* yang beroperasi pada 32-bit, yaitu:

$$Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z)$$
$$Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$
$$\sum_{0}^{256}(x) = ROTR^{2}(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x)$$
$$\sum_{1}^{256}(x) = ROTR^{6}(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x)$$
$$\sigma_{0}^{256}(x) = ROTR^{7}(x) \oplus ROTR^{18}(x) \oplus SHR^{3}(x)$$
$$\sigma_{1}^{256}(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$$

Pada penelitian ini digunakan perpaduan algoritma *HMAC* dengan algoritma *SHA256* yang disingkat sebagai *HS256*.

3 Metodologi Penelitian

Untuk mencapai tujuan dari kegiatan yang dilakukan, maka disusun beberapa tahapan dari mekanisme kegiatan ini sebagai berikut.



Gambar. 1. Kerangka Pikir.

3.1 Studi Literatur

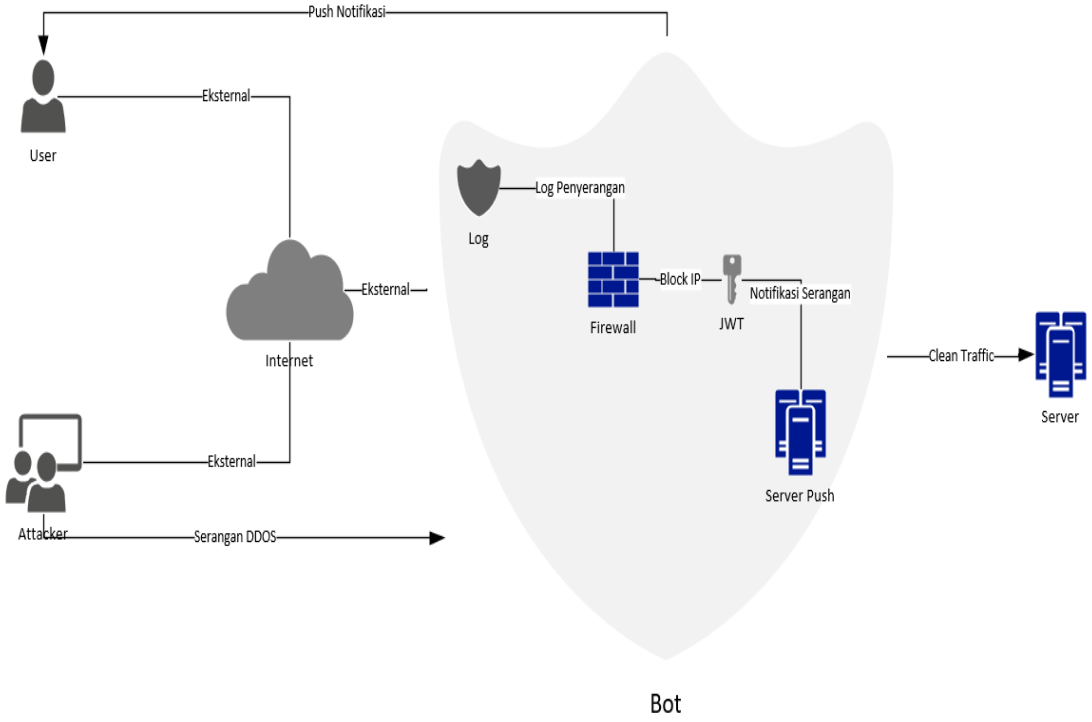
Tahap ini dilakukan untuk mempelajari literatur yang berhubungan dengan permasalahan yang sudah terindentifikasi, diantaranya mengenai serangan *DDOS* berjenis *UDP Flooding* dan *SYN Flooding*, implementasi *JSON Web Token*, implementasi *bot*, dan mengenai hal yang berhubungan dengan identifikasi masalah lainnya. Literatur ini diperoleh dari berbagai sumber, diantaranya buku, *e-book*, jurnal, *website*, dll.

3.2 Perancangan Sistem

Tahap ini berguna untuk memperoleh cara yang efektif dan efisien untuk melakukan implementasi sistem. Tahap perancangan sistem akan menghasilkan kerangka untuk implementasi sistem yang telah dirancang. Berikut ini merupakan tahapan dalam desain yaitu :

3.2.1 *Monitoring* Server

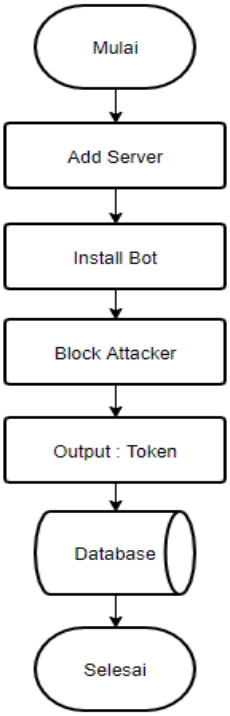
Untuk membuat sebuah sistem *monitoring* server dari serangan *DDOS* berjenis *SYN Flooding* dan *UDP Flooding*, maka akan dilakukan perancangan sistem *monitoring* server. Ditampilkan pada gambar berikut.



Gambar. 2. Rancangan Sistem *Monitoring*.

3.2.2 *Bot Token*

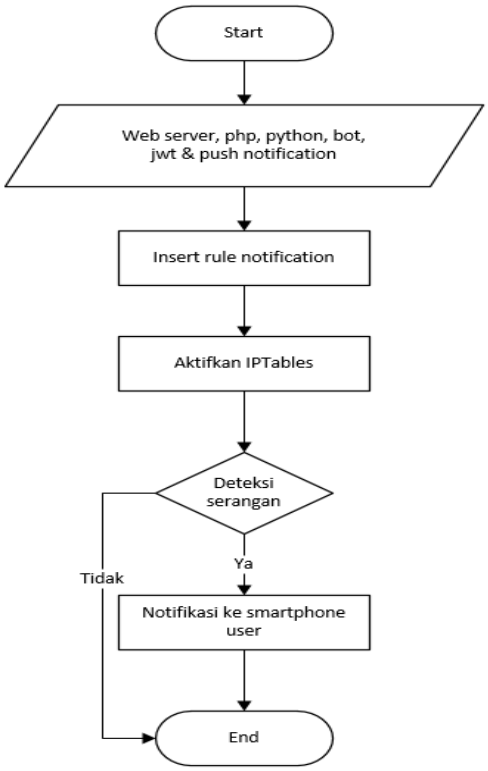
Untuk pemberitahuan peringatan intrusi dilakukan dengan memanfaatkan aplikasi *android* yaitu fitur *push notification bot*. Hal penting dalam menggunakan fitur tersebut adalah sudah memperoleh token *push notification bot*.



Gambar. 3. *Flowchart Set Push Notification Bot Token*

3.2.3 Program *bot*

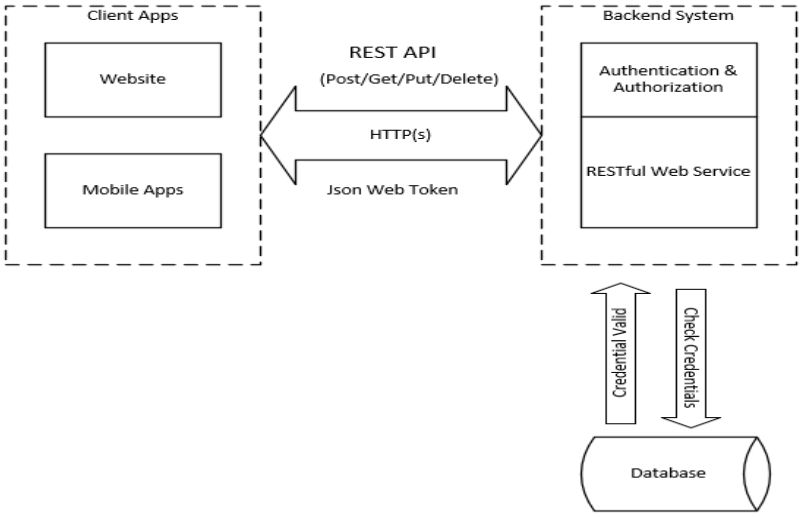
Berikut ini merupakan *flowchart* dari desain program *bot* yang ingin dirancang.



Gambar. 4. *Flowchart* Program *Bot*.

3.2.4 *JSON Web Token*

Berikut ini merupakan model rancangan serta peranan *JSON Web Token* pada sistem yang akan dirancang.

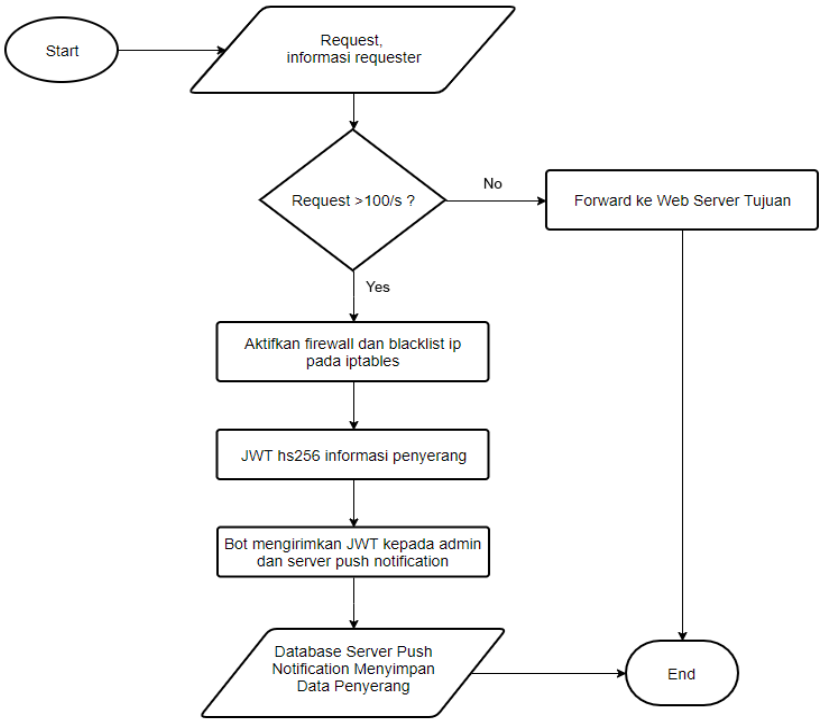


Gambar. 5. Model Penerapan *JWT* Pada Sistem.

4 Hasil dan Pembahasan

4.1 Analisa Sistem Monitoring

Sistem yang akan dirancang merupakan sebuah program bot untuk melakukan monitoring server serta aplikasi web dan aplikasi android. Perancangan ini akan membuat server mudah untuk ditambahkan dari web *interface*, dan *log* notifikasi ke *smartphone* admin server. Dalam gambaran umum sistem, *bot* digambarkan dengan alir posisi dari *bot*, penyerang (*attacker*), dan juga data yang masuk dari internet.
Pada Gambar 7 terlihat ketika penyerang yang ingin mengakses web server terhalang dalam *bot*, kemudian ketika *action* tersebut terjadi maka *bot* akan melakukan *blocking* akses pada *attacker* dan memberikan notifikasi kepada *user* melalui *android*. Berikut ini merupakan *flowchart* untuk menjelaskan cara kerja *bot* yang lebih rinci.



Gambar. 6. *Flowchart* Cara Kerja *Bot*.

Aplikasi *bot* yang telah dirancang menggunakan *python* akan dijalankan secara bersamaan pada satu mesin yang sama dengan web server yang ingin dilindungi oleh *bot*. Ketika terdapat *request* yang ingin menuju kepada web server yang dilindungi oleh *bot* maka *request* tersebut akan ditahan terlebih dahulu oleh *bot* untuk menentukan apakah itu serangan berjenis *UDP Flooding* maupun *SYN Flooding* atau bukan. Cara *bot* untuk menentukan apakah *request* tersebut merupakan serangan berjenis *UDP Flooding* maupun *SYN Flooding* atau bukan adalah dengan cara memberikan 122atasan maksimal pada *request* yaitu 100 paket per detik. Jika *request* paket yang dikirimkan oleh pengirim kurang dari sama dengan 100 *request* per detik maka *bot* akan menyimpulkan bahwa *request* tersebut bukan termasuk serangan dan akan melanjutkan *request* tersebut ke web server yang ditujunya. Sedangkan jika *request* tersebut lebih dari 100 paket per detik maka *bot* akan mengidentifikasi *request* tersebut merupakan serangan *DDOS* berjenis *UDP Flooding* maupun *SYN Flooding*. Setelah mendeteksi adanya upaya serangan *DDOS* berjenis *UDP Flooding* maupun *SYN Flooding* ke web server yang dilindungi oleh *bot*, maka *bot* akan mengaktifkan *firewall* untuk memblokir serangan tersebut dan mengaktifkan *iptables* beserta *rules* di dalam *iptables* untuk memblokir serta *blacklist ip* penyerang. Setelah memblokir *ip* penyerang, *bot* akan mengumpulkan data penyerang seperti *ip* yang tercatat di dalam *iptables*, *port* yang diserang pada saat mengirimkan serangan dan waktu pengiriman serangan tersebut. Kemudian *bot* akan mengaktifkan *JSON Web Token* untuk melakukan *encode* pada data penyerang menggunakan perpaduan algoritma *HMAC* dengan *SHA256* yaitu *HS256*, setelah selesai melakukan *encode bot* akan mengirimkan *JWT* beserta notifikasi serangan melalui *server push notification*, *server push notification* akan menyimpan data penyerang tersebut lalu akan mengirimkan notifikasi serangan beserta *JWT* kepada *smartphone* admin.

4.2 Analisa Penerapan *JSON Web Token* pada *bot*

Pada penelitian ini *JSON Web Token* berfungsi untuk mengamankan pertukaran informasi antara dua belah pihak. Setiap informasi *request* yang dikirimkan oleh *user*, *bot* dan admin kepada web server maupun sebaliknya akan disisipkan ke dalam token oleh *JWT*. Berikut ini merupakan penerapan *JSON Web Token* pada sistem sehingga *bot* mampu mengoperasikan *JSON Web Token*.

```
import os,time,jwt,json,psutil,grequests,requests
```

Gambar. 7. *Package JWT.*

Untuk mengoperasikan *JWT*, langkah awal yang harus dilakukan adalah dengan memasukkan *package JWT* pada *bot*. Kemudian untuk memperkuat keamanan dibutuhkan *secret key* dengan begitu hanya pihak yang memiliki kunci tersebut yang dapat mengakses data.

```
jwt_key = 'PGAnEpRSfFgX9o23JO6iD2r4Jdwi73C6'
url_root="http://newdemo.aplikasiskripsi.com/arif_ddos/index.php/api/helloDecode?token="
url_update="http://newdemo.aplikasiskripsi.com/arif_ddos/index.php/api/updateSync?token="
```

Gambar. 8. *Secret Key* dan *Push Notification* Server.

Setelah *secret key* dideklarasikan, diperlukan juga untuk menentukan tujuan pengiriman *JWT*, pada penelitian ini *JWT* akan dikirimkan kepada *smartphone* admin server melalui *Push Notification Server*. Kemudian agar *bot* dapat mengoperasikan *JWT* diperlukan untuk memanggil fungsi *encode JWT* agar dapat melakukan enkripsi terhadap klaim data yang memiliki format *JSON* yang diperoleh *bot*.

```
mypayload = {"detect_time": detect_time,"port": portIp,"ip": sourceIp,"count_port":jumlahKoneksi,"cpu_percent":cpu_percent,"ram_usage":svmem['percent']}
encoded_jwt = jwt.encode(mypayload, jwt_key, algorithm='HS256')
print("======== JWT =======")
print(encoded_jwt)
url_send_report = url_root+encoded_jwt.decode("utf-8")
request = requests.get(url_send_report)
print(request.content)
```

Gambar. 9. Memanggil fungsi *encode JWT* pada *bot.*

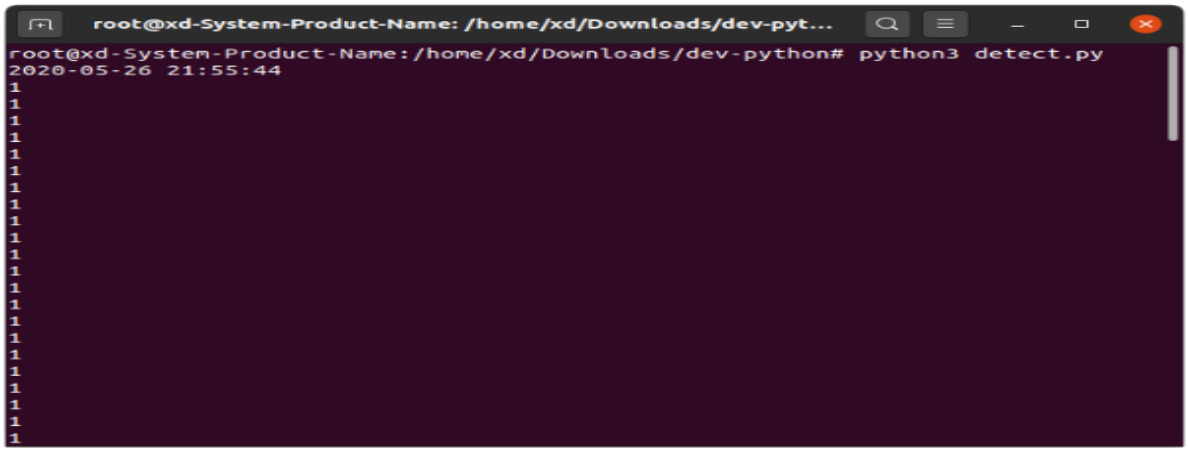
Pada gambar tersebut diketahui bahwa klaim yang diperoleh *bot* berisikan waktu deteksi, *ip* pengirim, dan *port* yang dituju. Kemudian klaim tersebut akan dienkripsi oleh *JWT* dengan memanggil fungsi *encode JWT* sehingga

klaim tersebut diubah menjadi token. Dalam mengubah klaim tersebut menjadi token, *JWT* menggunakan perpaduan algoritma *HMAC* dengan *SHA256* yaitu *HS256*.

4.3 Pengujian Sistem

Pengujian sistem dilakukan agar dapat menguji sistem *monitoring* yang sudah dirancang, serta untuk membuktikan apakah *bot* mampu untuk mendeteksi serangan *DDOS* berjenis *UDP Flooding* dan *SYN Flooding* serta memblokir serangan tersebut beserta dengan ip penyerang, lalu memberikan sebuah notifikasi kepada *smartphone* admin dengan memanfaatkan *set push notification* yang berisikan data serangan berbentuk *JSON* yang telah dilindungi keamanan data tersebut menggunakan *JSON Web Token*.

4.3.1 Pengujian *Bot* Sistem *Monitoring* Sebelum Adanya Serangan

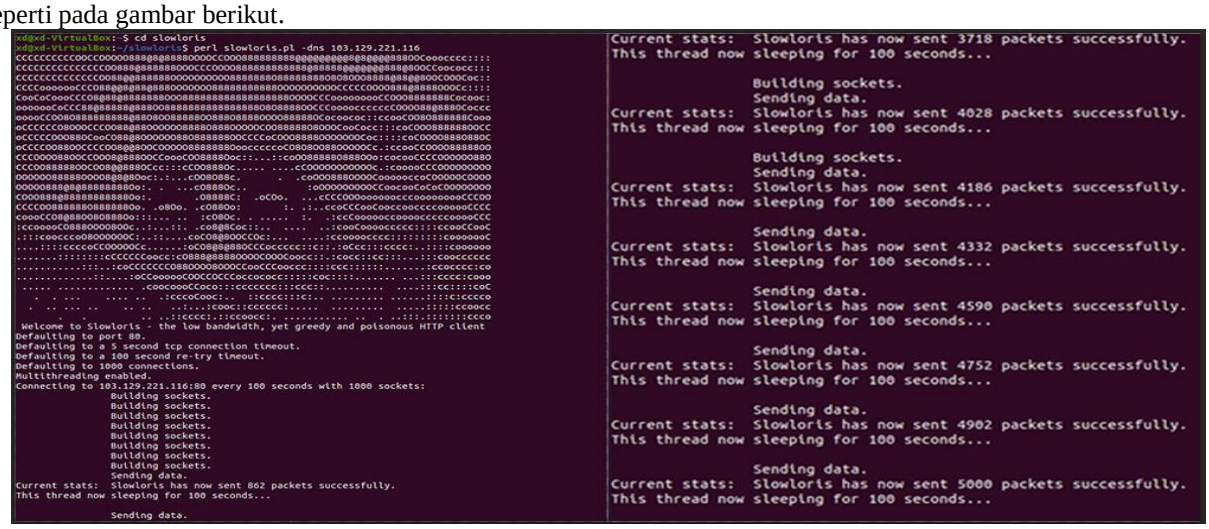


Gambar. 10. *Bot Monitoring* Server.

Pada gambar tersebut dapat dilihat bahwa *bot* sedang melakukan *monitoring* server dengan mengirimkan *report* berupa jumlah *request* yang masuk ke dalam web server yang dilindungi oleh *bot monitoring* server.

4.3.2 Pengujian Serangan *DDOS* Menggunakan *Slowloris*

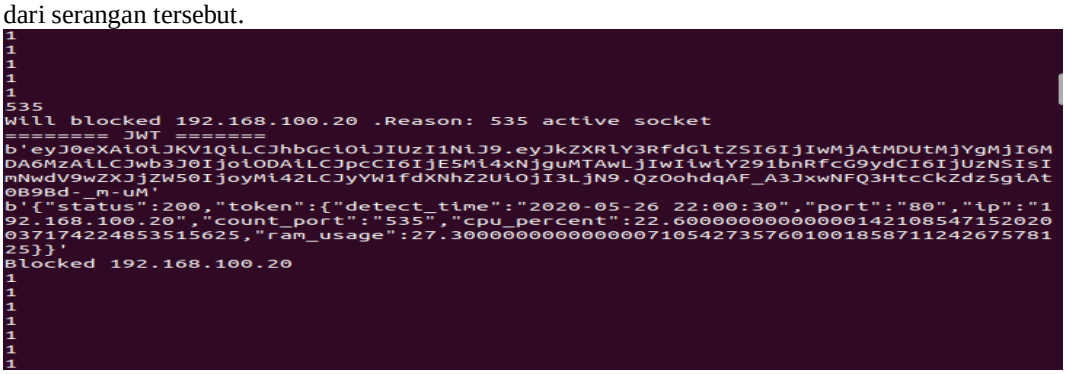
Penelitian ini menggunakan aplikasi *slowloris.pl* untuk mengirimkan serangan *Distributed Denial Of Service* berjenis *UDP Flooding* maupun *SYN Flooding*. Pada percobaan ini web server akan diserang untuk menguji apakah *bot* mampu untuk memblokir serangan *DDOS* berjenis *UDP Flooding* maupun *SYN Flooding* yang ditujukan kepada web server yang dilindungi oleh *bot*. Berikut merupakan serangan menggunakan *slowloris.pl* seperti pada gambar berikut.



Gambar. 11. Serangan *DDOS* Menggunakan *Slowloris*

4.3.3 Pengujian *Bot* Sistem *Monitoring* Saat Adanya Serangan

Pengujian *bot* sistem *monitoring* saat adanya serangan dilakukan untuk memastikan bahwa *bot* mampu untuk mendeteksi serangan *DDOS* berjenis *UDP Flooding* maupun *SYN Flooding* dan mampu untuk melindungi web server dari serangan tersebut.



Gambar. 12. *Bot* Mendeteksi dan Memblokir Serangan.

Gambar. 13. *IP* Penyerang Terdaftar Dalam *Blacklist IPTables Chain Input*.

Gambar. 14. Notifikasi Terdeksinya Serangan.

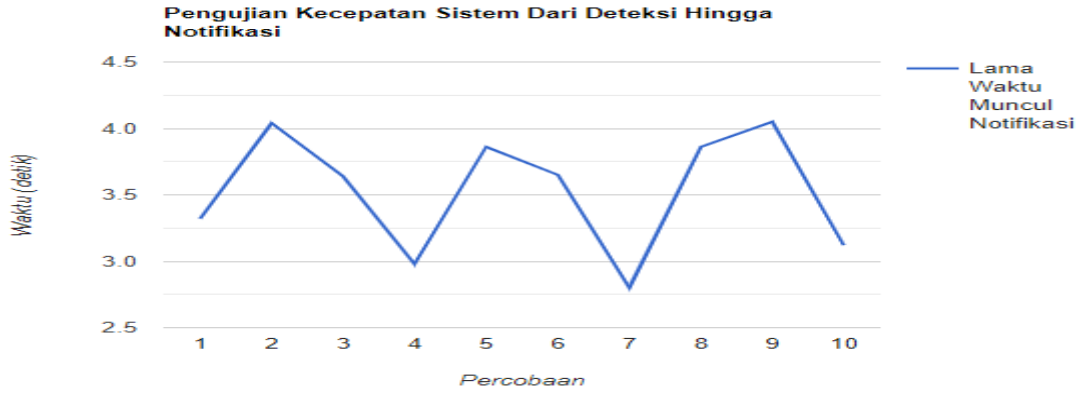
4.4 Pengujian Kecepatan Sistem Dari Deteksi Hingga Notifikasi

Dalam pengujian ini dilakukan sepuluh kali percobaan serangan *DDOS* berjenis *UDP Flooding* dan *SYN Flooding* pada web server yang dilindungi oleh *bot*. Alat yang digunakan untuk mengukur waktu dari terjadinya serangan *DDOS* hingga muncul notifikasi pada *smartphone* admin adalah *stopwatch*. Hasil pengukuran kecepatan sistem dari deteksi hingga muncul notifikasi dari sepuluh kali percobaan adalah sebagai berikut.

Tabel. 1. Tabel hasil pengukuran.

Percobaan	Waktu (detik)
1	3.32
2	4.04
3	3.64
4	2.98
5	3.86
6	3.65
7	2.8
8	3.86
9	4.05
10	3.12

ISSN : 2655-139X (ONLINE)
ISSN : 0216-4221 (PRINT)



Gambar. 15. Hasil Pengukuran Kecepatan Sistem.

Grafik tersebut menampilkan hasil kecepatan dari sepuluh kali percobaan dalam mengukur kecepatan sistem dari deteksi hingga muncul notifikasi ketika web server yang dilindungi oleh sistem diserang. Berdasarkan hasil percobaan tersebut diperoleh rata-rata kecepatan sebesar 3.532 detik.

4.5 Analisa Hasil Pengujian

Berdasarkan dari pengujian-pengujian yang telah dilakukan, dapat diperoleh beberapa hasil pengujian adalah sebagai berikut.
1. *Bot* mampu untuk melakukan *monitoring* server sehingga memudahkan admin server karena tidak harus selalu memantau server.
2. *Bot* berhasil mendeteksi serta memblokir serangan *DDOS* berjenis *UDP Flooding* dan *SYN Flooding*.
3. *Bot* berhasil mengirimkan notifikasi berisikan data penyerangan kepada *smartphone* admin server.
4. Rata-rata kecepatan sistem dari awal deteksi serangan hingga muncul notifikasi pada *smartphone* admin server memakan waktu rata-rata 3.532 detik.

5 Penutup

5.1 Kesimpulan

Adapun kesimpulan yang diambil berdasarkan dari hasil penelitian yang telah dilakukan sebagai berikut :
1. Untuk mendeteksi serangan *DDOS*, jumlah paket yang dapat diterima oleh *port* yang dibuka oleh server dibatasi menjadi 100 paket per detik pada *firewall* yang diatur oleh *bot*.
2. Waktu rata-rata yang dibutuhkan sistem dari memblokir serangan hingga munculnya notifikasi pada *smartphone* admin sebesar 3.532 detik.

5.2 Saran

Adapun saran dari penelitian ini adalah sebagai berikut:
1.Membuat laporan *monitoring* berbentuk *File*.
2.Membuat *bot* bisa bekerja untuk membedakan antara paket serangan *DDOS* atau *packet user*.
3.Melakukan pengembangan untuk jenis-jenis serangan *DDOS*.
4.Melakukan pengembangan untuk melakukan implementasi pada sistem jaringan *workstation* yang lebih besar seperti *smart city*, *smart detector* dan sebagainya.

Referensi

[1] Adrian, R., Nur, I. H. (2016). Analisa Pengaruh Variasi Serangan DDOS Pada Performa Router. Yogyakarta. *Seminar Nasional Teknologi Terapan SV UGM 2016 vol.6*, hh. 1257-1259.

[2] Siswanto A,Faldana, R. (2014). Sistem Monitoring Rumah Berbasis Teknologi Cloud Computing (Tugas Akhir), *Seminar Nasional Sistem Informasi Indonesia*, hh. 276-283.
[3] Bradley, J. (2015). *JSON Web Token (JWT)*, 1– 30.
[4] Dwiyatno, S., Sari, A., Irawan, A., & Safig, S. (2019). PENDETEKSI SERANGAN DDoS (DISTRIBUTED DENIAL OF SERVICE) MENGGUNAKAN HONEYPOT DI PT. TORINI JAYA ABADI. *Jurnal Sistem Informasi Dan Informatika (Simika)*, 2(2), 64-80. *Retrieved from http://ejournal.lppm-unbaja.ac.id/index.php/jsii/article/view/606*
[5] Aprilianto, Doni, Fadila, Triyana, & Muslim, Much Aziz. (2017). Sistem Pencegahan UDP DNS Flood dengan Filter Firewall Pada Router Mikrotik. Techno.COM, 16(2), hh.114-119
[6] Andi, Hasibuan, M. S. (2018). Model Analisis Ancaman SYN Flooding dalam Jaringan. *Jurnal Teknovasi*, 5(1), 66-71. *Retrieved from https://ejurnal.plm.ac.id/index.php/Teknovasi/article/view/227/pdf*
[7] Situmorang, Jhon Daniel. (2013). Implementasi Algoritma Keyed-Hash Message Authentication Code (HMAC) Pada Pesan Teks Berbasis Chatting. *Pelita Informatika Budi Darma*, 3(2), hh. 89-95.
[8] Sulastri, S., Defi, M. P. R. (2018). Studi Perbanding Penggunaan Algoritma Hash SHA 256 dengan Simetrik dan Asimetrik Ciphers dalam Perancangan Secure SWF Rich Internet Application (RIA), *Jurnal Teknik Elektro* 10(2), hh. 70-74

ISSN : 2655-139X (ONLINE)
ISSN : 0216-4221 (PRINT)