Aplikasi untuk *Monitoring* Jaringan IOT Menggunakan Algoritma *Address Shuffling* dengan HMAC

Elvan Alandi [1], Agustinus Noertjahyana [2], Justinus Andjarwirawan [3]
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121-131. Surabaya 60236
Telp (031) – 2983455, Fax. (031) 8417658
E-mail: elvanalandi@gmail.com [1], agust@petra.ac.id [2], justin@petra.ac.id [3]

ABSTRAK

Dengan meningkatnya pasar *Internet of Things* dan *Wireless Sensor Network* (WSN). Peningkatan keamanan jaringan pada *Internet of Things* menjadi hal yang penting. *Internet of Things* dipakai mulai dari *smart home, smart campuses, smart city,* dan lain-lain sehingga mengamankan jaringan *Internet of Things* lebih penting daripada mengamankan jaringan lokal biasa. Selain itu, perangkat IoT memiliki kelemahan dalam kapabilitas komputasi dan penyimpanan data sensitif. Terdapat empat macam kelemahan utama pada jaringan yaitu *address spoofing attack, false address conflict, address exhaustion attack dan negative reply attack*.

Metode yang digunakan adalah *Address shuffling* dengan HMAC. Metode ini dijalankan oleh DHCPv6 *server* (*Dynamic Host Configuration Protocol*) sebagai *network coordinator* dan di-*monitoring* oleh aplikasi Android untuk mengamankan jaringan *Internet of Things*. DHCPv6 *server* digunakan pada jaringan *Internet of Things* ini karena lebih cepat dan mudah untuk dikontrol melalui *server,* dapat digunakan pada jaringan yang besar dan perintah *stop* dan *restart* dapat dilakukan pada *server*. Selain itu, ketika DHCPv6 *server* bermasalah, hal ini tidak akan menginterupsi perangkat lainnya dalam satu jaringan.

Hasil dari program ini yaitu aplikasi *monitoring* yang mampu memberitahukan kepada pengguna adanya serangan dan jaringan mampu meminimalisir empat kelemahan pada jaringan. Sehingga nantinya aplikasi ini dapat membantu pengguna untuk melakukan tindakan preventif terhadap serangan pada jaringan.

Kata Kunci: *Internet Of Things, Wireless sensor network, Monitoring,* Keamanan

ABSTRACT

With the increasing market of the Internet of Things and Wireless Sensor Network (WSN). Increasing network security on the Internet of Things is important. Internet of Things is used starting from smart homes, smart campuses, smart cities, etc. so securing the Internet of Things network is more important than securing ordinary local networks. In addition, IoT devices have weaknesses in computing capabilities and storing sensitive data. There are four main types of weaknesses in the network, namely address spoofing attacks, false address conflicts, address exhaustion attacks and negative reply attacks.

The method used is Address shuffling with HMAC. This method is run by the DHCPv6 server (Dynamic Host Configuration Protocol) as a network coordinator and is monitored by the Android application to secure the Internet of Things network. DHCPv6 server is used on the Internet of Things network because it is faster and easier to control through the server, can be used on large networks and stop and restart commands can be done on the server. Additionally, when a DHCPv6 server has a problem, this will not interrupt other devices on one network.

The result of this program is a monitoring application that is able to notify users of attacks and the network is able to minimize four weaknesses in the network. So that later this application can help users to take preventive actions against attacks on the network.

Keywords: *Internet Of Things, Wireless sensor network, Monitoring, Security*

1. PENDAHULUAN

Dengan meningkatnya pasar *Internet of Things* dan *Wireless Sensor Network* (WSN). Peningkatan keamanan jaringan pada *Internet of Things* menjadi hal yang penting. *Internet of Things* dipakai mulai dari *smart home, smart campuses, smart city,* dan lain-lain sehingga mengamankan jaringan *Internet of Things* lebih penting daripada mengamankan jaringan lokal biasa. Pada jaringan *ad hoc* atau jaringan yang biasa dipakai untuk suatu jaringan terdapat berbagai macam kelemahan dalam hal keamanan. Mulai dari *address spoofing attack, false address conflict, address exhaustion attack dan negative reply attack* [7]. *Address spoofing attack* merupakan serangan yang dilakukan melalui penyadapan IP *address* dan menginjeksi *packet*. *False address conflict* merupakan serangan yang dilakukan untuk mempengaruhi *address collision prevention mechanism* yang berakibat pada *Denial of Service* (DoS). *Address exhaustion attack* merupakan serangan yang bertujuan menguras kuota untuk IP *address* dengan cara mendeklarasikan IP *address* dengan jumlah besar yang membuat perangkat baru tidak dapat bergabung dengan jaringan. *Negative reply attack* merupakan serangan yang dilakukan dengan berpura-pura menjadi koordinator jaringan dan memberi *deny message* kepada perangkat yang ingin mendapatkan IP *address* baru.

Pada jaringan digunakan algoritma *address shuffling* dengan HMAC (AShA), algoritma ini dijalankan oleh DHCPv6 *server* (*Dynamic Host Configuration Protocol*) sebagai *network coordinator* dan di-*monitoring* oleh aplikasi Android untuk mengamankan jaringan *Internet of Things*. DHCPv6 *server* digunakan pada jaringan *Internet of Things* ini karena lebih cepat dan mudah untuk dikontrol melalui *server,* dapat digunakan pada jaringan yang besar dan perintah *stop* dan *restart* dapat dilakukan pada *server*. Selain itu, ketika DHCPv6 *server* bermasalah, hal ini tidak akan menginterupsi perangkat lainnya dalam satu jaringan. Keunggulan-keunggulan dari DHCPv6 server tersebut tidak dapat ditemukan pada router. Dalam algoritma ini terdapat *hashing* yaitu *Keyed-hash Message Authentication Code* (HMAC) yang digunakan untuk mengamankan IP *address* yang dikirim. *Network coordinator* dapat mengirimkan *address change multicast message* kemudian semua perangkat yang terkoneksi akan mengevaluasi IPv6 dan MAC yang baru. Dengan algoritma ini, empat kemungkinan serangan di atas dapat terminimalisir.

Aplikasi yang diusulkan pada skripsi ini menggunakan sistem operasi Android. Market share Android pada April 2019 mencapai angka 74.85 % melampaui sistem operasi iOS [10]. Angka tersebut membuktikan bahwa sistem operasi Android banyak digunakan.

2. TINJAUAN STUDI

2.1 IoT

IoT atau *Internet of Things* merupakan perangkat di seluruh dunia yang saling terkoneksi pada internet, mengkoleksi dan berbagi data. IoT membuat dunia menjadi lebih cerdas dan responsif, menggabungkan dunia digital dan fisik. Contoh perangkat IoT adalah lampu dapat dinyalakan dengan aplikasi *smartphone* atau mobil yang dapat mengemudi sendiri. Sensor dari IoT mengambil berbagai macam data yang sensitif tetapi kemanan dari IoT masih sangatlah rendah [9].

2.2 Android

Android adalah sistem operasi yang berbasis linux. Android diciptakan oleh Google yang digunakan untuk sistem operasi pada smartphone. Android memiliki berbagai macam versi seperti Marshmallow, Nougat, Pie atau Android 10. Perangkat android dibuat oleh berbagai macam perusahaan seperti Samsung, HTC, Motorola, Sony, OnePlus dan sebagainya [2].

2.3 IPv6

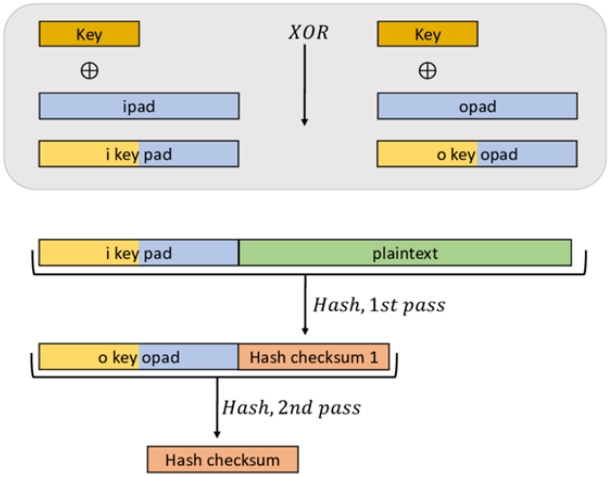
IPv6 merupakan standard protokol internet untuk menghindari terjadinya *address exhaustion*. IPv6 memiliki 128 bit *size* dan menggunakan hexadecimal sebagai IP *address* [4].

2.4 DHCPv6

Secara umum DHCPv6 adalah protokol yang memberikan DHCPv6 *server* fleksibilitas untuk mengalokasikan IPv6 *address* dan mengatur informasi *address* untuk *client*. Proses dimulai dari *client* memberikan RS *(Router Solicitation*) *message* kepada *router* dengan protokol ICMPv6 untuk mencari letak dari *router*, *router* memberikan tanggapan dengan memberikan RA (*Router Advertisement*) *message* kepada *client* yang berisi *prefix* yang akan digunakan oleh *client*, *client* memberikan *solicit message* secara *multicast* kepada *server*, *server* memberikan tanggapan dengan informasi konfigurasi kepada *client* melalui *advertise message*, *client* memberikan konfirmasi dengan mengirimkan *request message* kepada *server*, selanjutnya *server* akan mengkonfirmasi konfigurasi dari *client* dnegan memberikan *reply message* [6].

2.5 HMAC-SHA256

HMAC-SHA256 merupakan metode autentikasi dengan *hashing* [3]. Data dan pesan akan melalui algoritma yang disebut sebagai *cryptographic hash function* atau *one way hash function* yaitu SHA256. Proses dari HMAC ditunjukan oleh gambar 1. Ipad memiliki value 0x5c, opad memiliki value 0x36, i key pad merupakan xor dari ipad dan *key*, o *key* pad merupakan xor dari opad dan *key*.



Gambar 1. Proses HMAC [8]

2.6 Algoritma *Address Shuffling* dengan HMAC

Algoritma AShA dijalankan oleh *network coordinator*. Algoritma ini memiliki variabel METAindexmax dan METAindexcur yang diinisialisasi oleh *network coordinator* dimana METAindexmax > METAindexcur. Variabel r merupakan variabel baru yang digunakan sebagai *counter* dan perhitungan HMAC. Saat nilai r ≤ METAindexmax, maka setiap *node* yang terhubung dengan *network coordinator* akan menetapkan IPv6 dan MAC *address*-nya berdasarkan perhitungan HMAC dimana *key* dan ID setiap *node* telah diketahui oleh *network coordinator*. Bila terdapat address yang sama maka *collision* terdeteksi sehingga mengubah variabel *collision* menjadi *True* dan menambah satu *counter* variabel r. Bila *collision* tetap *False* maka METAindexcur = r dan *network coordinator* akan mengirimkan nilai r kepada setiap node. Bila r = METAindexmax maka *counter* r menjadi 0 dan network coordinator akan men*generate* dan mendistribusikan *key* yang baru. *Hashing* yang dipakai adalah SHA-256 [7].

2.7 DHCPKit

DHCPKit merupakan aplikasi *open source* DHCPv6 *server* yang ditulis dengan bahasa pemrograman Python. Aplikasi ini mampu memberikan layanan DHCPv6 *server* yang dapat dimodifikasi sesuai keinginan atau dapat dikatakan sebagai DHCPv6 *server* yang fleksibel [5].

2.8 Penelitian Sebelumnya

Penelitian yang berjudul "*IoT Security via Address Shuffling: the Easy Way*" [7]. Dalam penelitian ini mereka hanya melakukan simulasi dengan metode *address shuffling* pada LR-WPAN tetapi tidak melakukan simulasi pada jaringan WLAN.

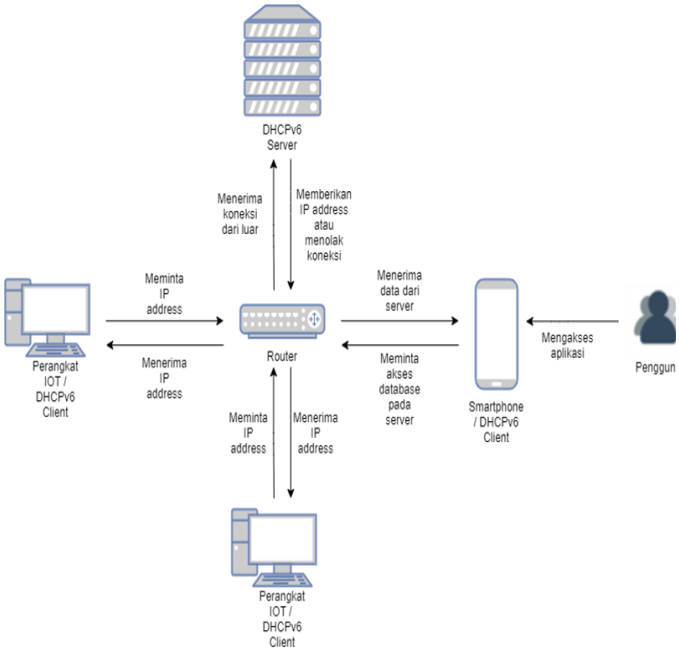
Selain itu, ada penelitian yang berjudul "*Authentication and Privacy Approach for DHCPv6*" [1]. Dalam penelitian ini mereka menggunakan enkripsi AES-GCM pada DHCPv6 tetapi tidak menggunakan HMAC dengan hashing SHA-256.

3. DESAIN SISTEM

3.1 Arsitektur Sistem Keseluruhan

Untuk mengetahui bagaimana sistem bekerja, dibutuhkan sebuah desain jaringan dari sistem. Hal ini bertujuan agar pengguna bisa

memahami proses yang terjadi pada *monitoring* jaringan IoT. Gambar 2 merupakan gambar desain sistem secara keseluruhan dari aplikasi *monitoring* jaringan IoT.

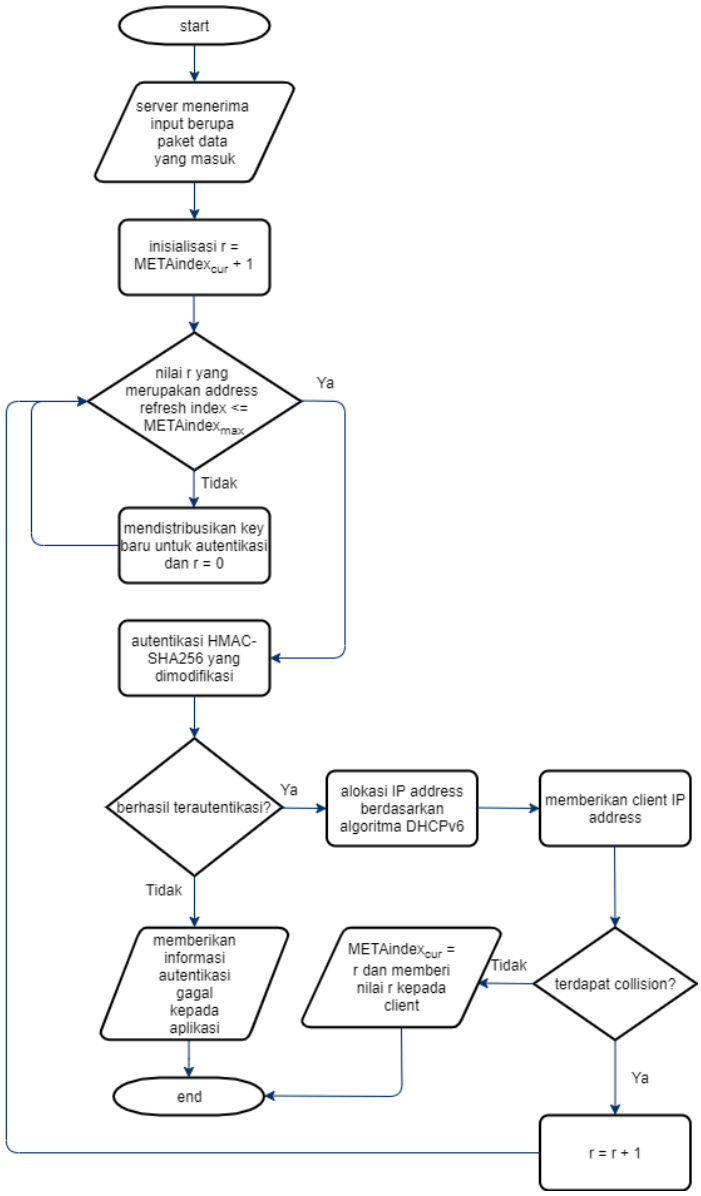


Gambar 2. Arsitektur Sistem Keseluruhan

Pengguna harus menggunakan *smartphone* berbasis sistem operasi Android untuk bisa mengakses aplikasi *monitoring* jaringan IoT. Hal ini karena aplikasi dibuat menggunakan Android Studio, yang mana, hanya bisa digunakan pada *smartphone* berbasis sistem operasi Android. Pertama-tama, perangkat IoT terhubung dengan *server* dan meminta IP *address*. Selanjutnya *server* melakukan autentikasi dengan perangkat IoT. Ketika berhasil melakukan autentikasi, perangkat IoT akan diberikan IP *address*. Pengguna dapat melakukan *monitoring* melalui pengaksesan aplikasi. Aplikasi akan meminta akses menuju *database* dari *server*. Jika *server* memberikan akses, aplikasi akan menerima data dari *server*. Data-data dari *server* berupa list IP *address* yang akan ditampilkan pada aplikasi dan informasi saat ada *collision* atau autentikasi gagal.

3.2 Arsitektur Sistem pada *Server*

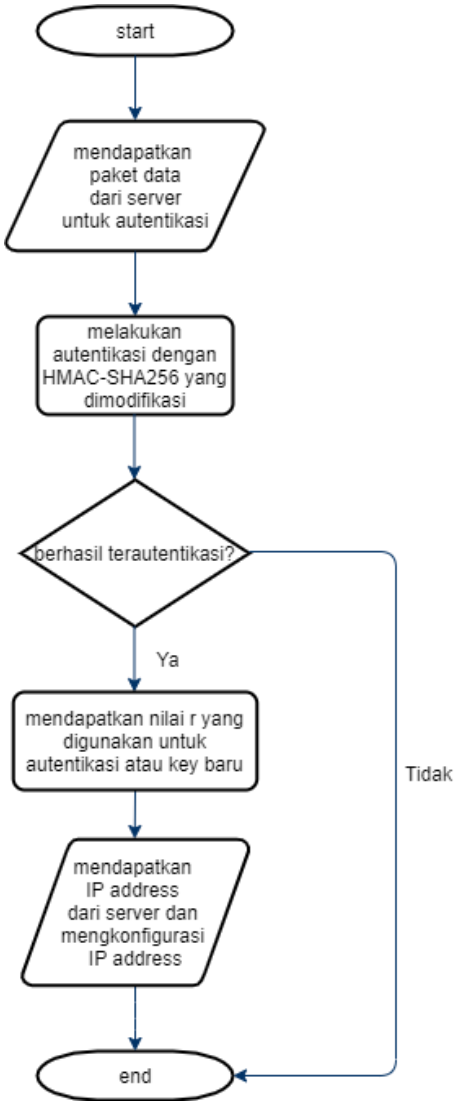
Untuk membuat sistem pada *server* yang sesuai dengan algoritma *address shuffling* dengan HMAC, maka dilakukan perancangan sistem dengan menggunakan *flowchart*. Gambar 3 merupakan *flowchart* yang berisikan sistem pada *server*. Alur *flowchart* ini merupakan alur berdasarkan dengan algoritma *address shuffling* dengan HMAC seperti yang ditunjukan bab sebelumnya, namun terdapat modifikasi seperti pemberian informasi *collision* atau autentikasi gagal pada aplikasi.



Gambar 3. *Flowchart* Sistem pada *Server*

3.3 Arsitektur Sistem pada *Client*

Untuk membuat sistem pada *client* yang dapat berkomunikasi dengan *server* dengan algoritma *address shuffling* dengan HMAC, maka dilakukan perancangan sistem dengan menggunakan *flowchart*.

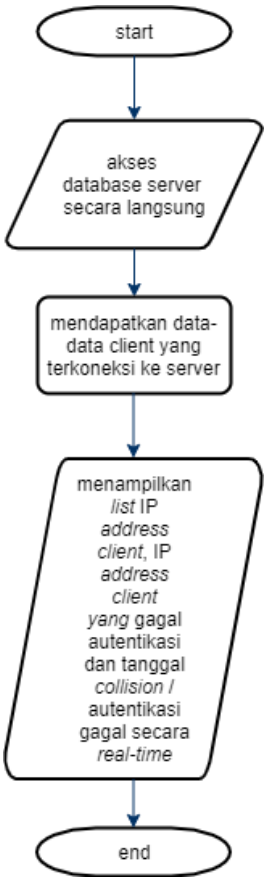


Gambar 4. *Flowchart* Sistem pada *Client*

Gambar 4 merupakan *flowchart* yang berisikan sistem pada *client*. Jika autentikasi pada *client* gagal, client tidak mendapatkan IP *address*. Jika berhasil, *client* mendapatkan nilai r yang disimpan untuk digunakan pada pesan autentikasi atau *key* baru dan mendapatkan IP *address* dari *server* untuk dikonfigurasi.

3.4 Arsitektur Sistem pada Android

Untuk membuat sistem pada Android yang dapat memberikan aktivitas *monitoring* pada setiap *client*, maka dilakukan perancangan sistem dengan menggunakan *flowchart*. Gambar 5 merupakan *flowchart* yang berisikan sistem pada Android. Secara sederhana sistem dari Android memiliki dua fungsi yaitu memberitahukan *user* mengenai *collision* atau autentikasi gagal dan menampilkan data IP *address* dari *client*. *Collision* atau autentikasi gagal diinformasikan dengan cara menampilkan data waktu *collision* atau autentikasi gagal. Untuk autentikasi gagal ditambahkan data IP *address* untuk memberitahu pengguna, *client* dengan IP *address* tersebut gagal melakukan autentikasi.



Gambar 5. *Flowchart* Sistem pada Android

4. PENGUJIAN SISTEM

4.1 Pengujian HMAC-SHA256

Pengujian bertujuan untuk mengetahui apakah HMAC berjalan dengan baik. Pada pengujian digunakan *log* yang membuktikan hasil *hashing* dari HMAC pada *server* dan *client*. Hasil pengujian dapat dilihat pada gambar 6.

Gambar 6. *Log* HMAC-SHA256

Pada gambar 6 terlihat hasil *hashing* antara *client* dan *server*. Hasil *hashing* yang sama membuktikan autentikasi sukses. Pada baris sebelum hasil *hashing client* dan *server* merupakan IP *address* dari *client* yang terkoneksi. IP *client* yang terkoneksi merupakan IP *address* sebelum pergantian IP *address*.

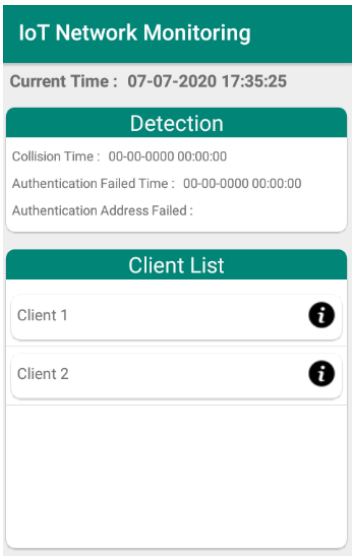
4.2 Pengujian Aplikasi Android

Pengujian aplikasi difokuskan untuk mengetahui tampilan pada *smartphone* dengan spesifikasi yang berbeda-beda. Ada dua *smartphone* yang digunakan untuk pengujian aplikasi. Spesifikasi masing-masing *smartphone* yang digunakan untuk pengujian dapat dilihat pada Tabel 1.

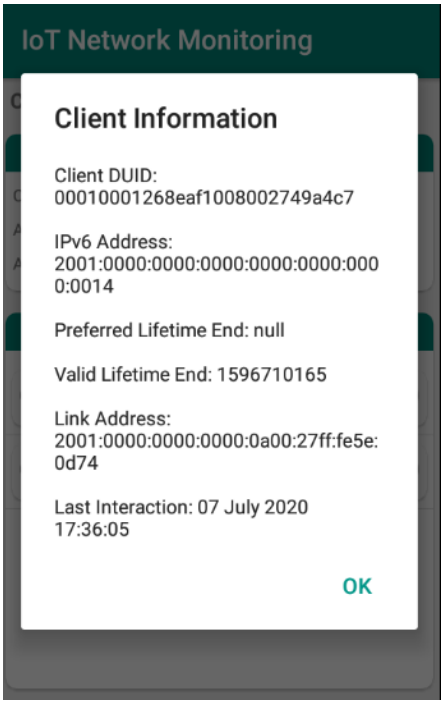
Tabel 1. Tabel Spesifikasi *Smarthphone* untuk Pengujian Aplikasi

Smartphone	*Operating System*	*Display Size*	CPU	*Memory*
Samsung Galaxy S8 (SM-G950FD)	Android 8.0 (Oreo)	1440 x 2960 pixels, 18.5:9 ratio (~570 ppi density)	Octa-core (4x2.3 GHz Mongoose M2 & 4x1.7 GHz Cortex-A53)	64GB, 4GB RAM
Samsung Galaxy S7 (SM-G930F)	Android 7.1.2 (Nougat)	1440 x 2560pixels, (577 ppi density)	Octa-core (4x2.3 GHz Mongoose & 4x1.6 GHz Cortex-A53)	32 GB, 4 GB RAM

Pengujian tampilan awal aplikasi *monitoring* ditunjukkan oleh gambar 7 dan gambar 8. Pada gambar 7 aplikasi berjalan dengan baik pada kedua *smartphone* yang menjadi perangkat untuk menguji aplikasi. *Current time* pada aplikasi merupakan waktu yang berjalan secara *real time. Collision time* dan *authentication failed time* berisikan tanggal *collision* atau autentikasi gagal sedangkan *authentication address failed* merupakan IP *address* dari *client* yang gagal melakukan autentikasi. Bagian bawah merupakan *list client* yang terkoneksi pada *server*. Saat *client* ditekan, akan muncul *detail client* yang berisi informasi data *client* seperti di gambar 8.

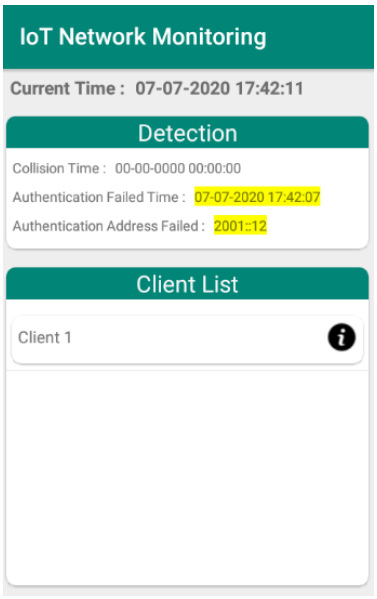


Gambar 7. Tampilan Awal Aplikasi

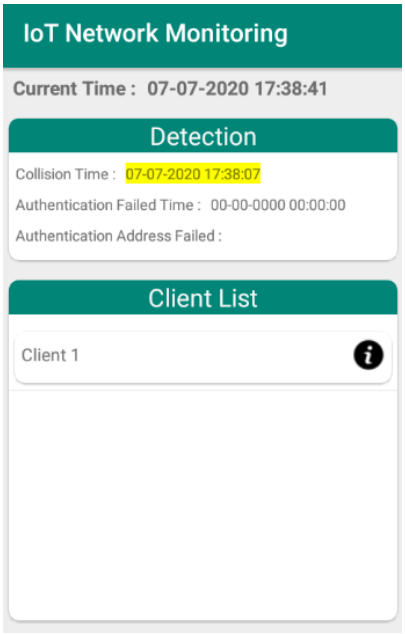


Gambar 8. Tampilan *Detail* pada *Client*

Terdapat fitur pendeteksi autentikasi gagal pada aplikasi. Pendeteksian melalui tanggal dan IP *address client* yang muncul pada aplikasi. Pada aplikasi tidak digunakan dialog karena kemungkinan pengguna tidak menyadari adanya dialog sedangkan dengan tanggal yang muncul dan IP *address client* pengguna aplikasi dapat mengetahuinya selama aplikasi masih menyala. Pada gambar 9 terlihat tanggal autentikasi gagal dan IP *address* dari *client* yang gagal terisi. Terlihat bahwa untuk melakukan koneksi autentikasi digunakan IP *address* lokal. Untuk koneksi autentikasi dapat menggunakan IP *address unicast* yang sebelumnya diberikan oleh *server*.



Gambar 9. Tampilan Autentikasi Gagal pada *Client*



Gambar 10. Tampilan *Collision* pada Jaringan

Terdapat fitur pendeteksi *collision* pada aplikasi. Pengguna dapat mengetahui adanya *collision* melalui tanggal seperti fitur pendeteksi autentikasi gagal. Hasil pendeteksian *collision* jaringan pada aplikasi *monitoring* dapat dilihat pada gambar 10.

4.3 Pengujian Fungsionalitas IP *Address*

Pada pengujian ini, digunakan ICMP untuk mengetahui koneksi jaringan berjalan dengan baik. Untuk ICMP versi 6 digunakan *command* "ping6 <*ip address* tujuan>". *Client* yang digunakan sebagai objek untuk diuji ditunjukan oleh gambar 11. Hasil dari pengujian fungsionalitas IP *address* ditunjukan oleh gambar 12.

```
user@client:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:36:fd:62
          inet addr:192.168.1.60  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: 2001::22/128 Scope:Global
          inet6 addr: fe80::a00:27ff:fe36:fd62/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:16054 errors:0 dropped:0 overruns:0 frame:0
          TX packets:15732 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1165275 (1.1 MB)  TX bytes:1259316 (1.2 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:176 errors:0 dropped:0 overruns:0 frame:0
          TX packets:176 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:13296 (13.2 KB)  TX bytes:13296 (13.2 KB)
```

Gambar 11. IP *Address Client* yang Diuji

```
sqlmoby@moby:~$ ping6 2001::22
PING 2001::22(2001::22) 56 data bytes
64 bytes from 2001::22: icmp_seq=1 ttl=255 time=0.426 ms
64 bytes from 2001::22: icmp_seq=4 ttl=255 time=0.626 ms
64 bytes from 2001::22: icmp_seq=5 ttl=255 time=0.702 ms
64 bytes from 2001::22: icmp_seq=6 ttl=255 time=0.659 ms
64 bytes from 2001::22: icmp_seq=7 ttl=255 time=0.457 ms
```

Gambar 12. Hasil Pengujian Fungsionalitas IP *Address*

Dari hasil pengujian fungsionalitas IP *address client*, IP *address* yang diberikan oleh DHCPv6 *server* dapat melakukan koneksi antar *client* dengan baik.

4.4 Pengujian Waktu Mendapatkan IP *Address*

Pada pengujian ini, digunakan *command time* untuk mengetahui waktu yang digunakan *client* untuk mendapatkan IP *address* dan dilakukan 3 kali percobaan. Hasil pengujian dapat dilihat pada tabel 2.

Tabel 2. Tabel Hasil Pengujian Waktu Mendapatkan IP *Address*

Percobaan	Waktu
1	56,212 detik
2	57,677 detik
3	59,029 detik

Dari hasil pengujian didapatkan bahwa untuk pembaruan IP diperlukan waktu yang sangat singkat.

4.5 Pengujian Rekomendasi Waktu *Shuffle* IP *Address*

Untuk mendapatkan rekomendasi dari pengujian sampel, dilakukan perhitungan rata-rata waktu yang dibutuhkan untuk pengiriman data dan rata-rata kecepatan pengiriman data. Hasil perhitungan dapat dilihat pada tabel 3.

Tabel 3. Tabel Perhitungan Rekomendasi Waktu *Shuffle* IP *Address*

Percobaan Sampel	Ukuran Sampel (MB)	Kecepatan (MB/s)	Waktu (detik)
1	70	3.9	17
2	75	4.2	18
3	80	4.0	20
Rata-rata	75	4.03	18.3

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil perancangan sistem dan penelitian ini, dapat diambil beberapa kesimpulan antara lain:

- Berdasarkan hasil pengujian, kesimpulan yang diperoleh adalah autentikasi HMAC-SHA256 sukses ditunjukkan oleh *hashing* pada *client* dan *server* yang sama pada pengujian HMAC-SHA256 pada algoritma AShA dengan HMAC.
- Aplikasi mendapatkan data *list client* beserta *detail client* melalui *server* dibuktikan oleh pengujian tampilan awal dan *detail client*.
- Aplikasi dapat mendeteksi *collision* atau autentikasi gagal dengan baik dibuktikan oleh pengujian pendeteksian *collision* dan autentikasi gagal.
- Jaringan IoT yang didasarkan pada algoritma *address shuffling* (AShA) dengan HMAC dapat berjalan baik pada

WLAN yang dibuktikan melalui pengujian fungsionalitas IP menggunakan ICMPv6.

- Pengujian waktu mendapatkan IP *address* sesuai dengan iterasi algoritma *address shuffling* (AShA) yang telah disebutkan pada implementasi *background process*.
- Waktu *shuffle* IP *address* dapat diatur menggunakan rumus pada pengujian rekomendasi waktu, diasumsikan kecepatan yang dikirimkan sesuai dengan rata-rata kecepatan pada pengujian rekomendasi waktu dan data yang dikirimkan statis.
- Dari pembuatan sistem dengan algoritma *address shuffling* dengan HMAC, algoritma *address shuffling* dapat membantu mengatasi kelemahan yang ada pada jaringan IoT. *Address spoofing* dapat dihindari dengan adanya *shuffle* IP *address*, *false address conflict* dapat diminimalisir dengan adanya pengecekan dan deteksi *collision* dengan algoritma AShA dan dapat di-*monitor* melalui aplikasi, *address exhaustion attack* dapat dimitigasi dengan menggunakan IPv6 dan autentikasi HMAC-SHA256, dan *negative reply attack* dapat dimitigasi dengan menggunakan autentikasi HMAC-SHA256 dan dapat di-*monitor* melalui aplikasi.
- Dari pembuatan aplikasi Android untuk *monitoring* jaringan IoT, pengguna dapat mengontrol jaringan IoT. Dengan adanya fitur deteksi *collision* dan autentikasi gagal, pengguna dapat mendeteksi 2 serangan pada jaringan IoT yaitu *false address conflict* dan *negative reply attack*. Informasi *client* dapat diketahui oleh pengguna aplikasi melalui adanya *list client* yang terhubung pada *server*.

5.2 Saran

Saran yang diberikan untuk penyempurnaan dan pengembangan lebih lanjut untuk penelitian ini adalah sebagai berikut:

- Implementasi program *address shuffling* dapat diberikan waktu sesuai dengan yang dimasukkan pada aplikasi.
- Sistem dengan algoritma *address shuffling* dengan HMAC dan aplikasi *monitoring* dapat dicoba pada jaringan yang besar atau dengan *client* kurang lebih 10 *client*.
- Waktu pembaruan IP *address* dapat disesuaikan dengan *size* data yang dikirimkan dan dikalkulasi dengan rumus rekomendasi waktu *shuffle* IP *address*.
- Aplikasi dapat dijalankan di *background* dan diberikan notifikasi saat terjadi *collision* atau autentikasi gagal.
- Aplikasi diberikan fitur lainnya, seperti *alert*, data grafik, dan gambar *chart.*

6. DAFTAR PUSTAKA

[1] Al-Ani, A., Anbar, M., Hasbullah, I. H., Abdullah, R., & Al-Ani, A. K. 2019. *Authentication and Privacy Approach for DHCPv6.* URI = https://www.researchgate.net/publication/333505317.

[2] Android Authority. 2020. *What is Android? Here's everything you need to know*. URI = https://www.androidauthority.com/what-is-android-328076/.

[3] Azeez, N. A., & Chinazo, O. J. 2018. *Achieving Data Authentication with HMAC-SHA256 Algorithm*. URI = https://www.researchgate.net/publication/332182220.

[4] Bajrami, Valentin. 2019. *What you need to know about IPv6*. URI = https://www.redhat.com/sysadmin/what-you-need-know-about-ipv6.

[5] DHCPKit. 2019. *DHCPKit – A Flexible DHCP Server for IPv6 Networks.* URI = https://www.dhcpkit.org/.

[6] Li, L., Ren, G., Liu, Y., & Wu, J. 2018. *Secure DHCPv6 Mechanism for DHCPv6 Security and Privacy.* URI = https://ieeexplore.ieee.org/abstract/document/8293069.

[7] Nizzi, F., Pecorella, T., Esposito, F., Pierucci, L., & Fantacci, R. 2019. *IoT Security via Address Shuffling: the Easy Way*. URI = https://ieeexplore.ieee.org/document/8606197.

[8] Ozkan, A. 2014. *Implementation of a Lightweight Trusted Platform Module*. URI = https://www.researchgate.net/publication/305956412.

[9] Ranger, S. 2020. *Everything you need to know about the Internet of Things right now.* URI = https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iot-right-now/.

[10] Statcounter. 2019. *Mobile Operating System Market Share Worldwide - April 2019.* URI = http://gs.statcounter.com/os-market-share/mobile/worldwide/.