Indonesian Journal of Electronics and Instrumentation Systems (IJEIS)
Vol.x, No.x, April xxxx, pp. 1~5
ISSN (print): 2088-3714, ISSN (online): 2460-7681
DOI: 10.22146/ijeis.xxxx

Penelitian Pengaruh MAC Address Collision Pada Perangkat Yang Dapat Merubah MAC Address

Gabriel Possenti Kheisa Drianasta*[1], Bakhtiar Alldino Ardi Sumbodo[2], Jazi Eko Istiyanto[3],
[1]Program Studi Elektronika dan Instrumentasi, DIKE, FMIPA, UGM, Yogyakarta, Indonesia
[1]Departemen Ilmu Komputer dan Elektronika, FMIPA UGM, Yogyakarta, Indonesia
e-mail: *[1]gabrielkheisa@mail.ugm.ac.id, [2]b.alldino.as@ugm.ac.id, [3]jazi@ugm.ac.id,

Abstrak

MAC address collision merupakan fenomena terjadi ketika dua atau lebih perangkat di suatu jaringan menggunakan MAC address yang sama. Beberapa sistem operasi terbaru, seperti Windows 10 dan Android 6.0, mendukung penggantian MAC address secara acak atau MAC address randomization untuk meningkatkan privasi pengguna dengan menyembunyikan MAC address yang sesungguhnya. Jika dua perangkat di suatu jaringan memiliki MAC address yang sama, maka jaringan tidak dapat membedakannya, dan dapat menyebabkan perselisihan dalam transmisi data.

Penelitian pengaruh MAC address collision dilakukan secara langsung pada router fisik dan dilakukan juga pada virtualized environment pada VirtualBox untuk mengetahui dampak apa yang akan terjadi dengan menganalisa variabel – variabel terikat, seperti latency, jumlah hop yang dilalui, dan keberhasilan menerima respon HTTP GET.

Hasil penelitian menunjukkan adanya peningkatan rata - rata latency, dari 0,2 milisekon hingga 1,2 milisekon, baik pada lingkungan virtual, maupun fisik, gagalnya konektivitas pada beberapa skenario percobaan, dan didapat respon HTTP GET yang tidak sesuai sebanyak 100% untuk 10 kali percobaan pada lingkungan fisik jika terjadi MAC Address Collision.

Kata kunci*— MAC address collision, MAC address randomization, duplicate, latency*

Abstract

MAC address collision occurs when two or more devices on a network use the same MAC address. Some of the latest operating systems, such as Windows 10 and Android 6.0, support changing the MAC address randomly with a method MAC address randomization to increase user privacy by hiding the real MAC address. If two devices on a network have the same MAC address, the network cannot tell the difference, and can cause disputes in data transmission.

Research on the influence of MAC address collisions was carried out directly on a physical router and carried out in a virtualized environment on VirtualBox to find out what impact would occur by analyzing dependent variables, such as latency, number of hops traversed, and success in receiving an HTTP GET response.

The results of the study showed an increase in the average latency, from 0.2 milliseconds to 1.2 milliseconds, both in virtual and physical environments. The connectivity also failed in some experimental scenarios, and HTTP GET responses were not received correctly with a failure rate of invalid responses 100% for 10 times of attempts in physical environments if a MAC Address Collision occurred.

Keywords— *MAC address collision, MAC address randomization, duplicate, latency*

Received June 1st,2012; Revised June 25th, 2012; Accepted July 10th, 2012

ceived June 1st,2012; Revised June 25th, 2012; Accepted July 10th, 2012

1. PENDAHULUAN

Dalam suatu jaringan internet, terdapat aturan - aturan atau standar agar suatu perangkat dapat berkomunikasi satu sama lain. Salah satu standar yang paling umum adalah standar IEEE 802 yang umum digunakan dalam jaringan LAN. (IEEE 802, 2004). IEEE 802 meliputi aturan – aturan atau standar untuk Ethernet (IEEE 802.3), Wi-Fi (IEEE 802.11), Bluetooth (IEEE 802.15), dan sebagainya, yang mengatur bagaimana sinyal Wi-Fi dipropagasikan, atau cara mendeteksi dan memperbaiki error dalam mengirimkan data. Di dalam IEEE 802, terdapat dua lapisan, yakni lapisan Data Link Layer yang terdiri dari Logical Link Layer (LLC) dan Medium Access Control (MAC), dan Physical Layer. MAC adalah lapisan yang mengatur pertanggungjawaban perangkat keras dalam komunikasi antar kabel, antar optik, atau nirkabel. MAC address mengatur pengenalan paket internet, pengalamatan tujuan paket, dan kontrol akses dalam medium fisik. Alamat MAC pada setiap perangkat unik, umumnya tidak dapat diubah dan memiliki panjang data 48-bit, dimana 24-bit pertama adalah kode dari perusahaan penerbit perangkat.

Ketika dua atau lebih perangkat di suatu jaringan memiliki MAC address yang sama, dapat membuat perselilishan pada lapisan Open Systems Interconnection (OSI) kedua, atau Data Link Layer. Kejadian ini disebut juga MAC address collision, dan dapat menyebabkan konflik dalam proses transmisi data dengan pengemasan alamat MAC sumber dan alamat MAC tujuan. Sebelum perangkat ingin mengirim paket data ke perangkat lain pada suatu jaringan, perangkat tersebut harus mengetahui dahulu alamat MAC perangkat yang dituju agar paket data dapat dikirim dengan tujuan yang benar. Proses pemetaan ini dilakukan oleh Address Resolution Protocol (ARP), yaitu sebuah protokol yang digunakan untuk menemukan alamat MAC tujuan dengan mencari terlebih dahulu alamat IP atau Internet.

Protocol address dari perangkat tujuan. Ketika ARP menemukan alamat IP perangkat tujuan, maka perangkat pengirim mengirim permintaan ARP ke jaringan tersebut untuk meminta alamat MAC yang sesuai. Hubungan antara alamat MAC dan ARP adalah jika alamat MAC adalah suatu kode untuk mengidentifikasi perangkat pada suatu jaringan, maka ARP merupakan protokol untuk menemukan alamat MAC antara perangkat pengirim dan penerima melalui pencarian alamat IP, dimana alamat IP ini sendiri berjalan pada lapisan yang lebih tinggi pada lapisan OSI ke-3, atau disebut juga Network Layer. Layer ke-3 ini bertanggung jawab dalam pengiriman paket data antar jaringan, dimana setiap perangkat memiliki alamat IP yang unik agar dapat diidentifikasi oleh jaringan, yang memiliki hubungan erat dengan ARP, dimana ARP itu sendiri memetakan alamat IP ke alamat MAC yang berjalan pada lapisan OSI satu tingkat dibawah IP. Network Layer juga menangani proses routing, yaitu proses mengirimkan paket data melalui jalur yang optimal. Network Layer mampu menentukan jalur terbaik untuk mengirimkan data dengan adanya informasi routing yang tersimpan pada routing table. Dalam hal ini, Internet Protocol (IP) juga merupakan elemen yang penting, karena IP, ARP, dan MAC berhubungan erat dan memiliki persamaan untuk mengidentifikasi perangkat pada suatu jaringan.

Meskipun MAC Address umumnya tidak dapat diubah, namun beberapa perangkat bahkan manufaktur boleh merubah alamat yang sudah bersifat hard-coded tersebut. Dengan mengubah MAC Address, identitas perangkat secara fisik juga berubah. (Cardenas, Edgar D., 2013) Dengan mengubah 24-bit pertama, maka perangkat tersebut dapat memanipulasi kode penerbit perangkat yang sebenarnya. Beberapa penerbit perangkat tersebut, khususnya dengan sistem operasi (OS) dan dukungan driver Network Interface Card (NIC) terbaru yang memperbolehkan mengubah MAC address demi tujuan anonimitas penggunanya. Tetapi jika MAC address perangkat tersebut memiliki address yang sama dengan perangkat lain dalam suatu jaringan yang sama, maka secara teoritis akan menimbulkan masalah pada jaringan tersebut.

Sebagai tambahan, metode randomization alamat MAC dapat dilakukan dengan menggunkanan fungsi hash dari alamat MAC asli, sehingga tercipta alamat MAC baru yang berbeda dari MAC address perangkat tersebut. (Jean-François Determe, Sophia Azzagnuni, François Horlin, Philippe De Doncker, 2022). Metode randomization alamat MAC dapat juga dilakukan dengan menambahkan salt, yaitu nilai acak atau random yang ditambahkan pada suatu nilai yang ingin di-hash. (Junade Ali, et al., 2020). Meskipun fungsi salt ini pada praktiknya untuk mengamankan password dari peretas yang mengetahui nilai hash yang umum, salt juga dapat mengurangi kemungkinan collision dengan lebih baik.

Ada pula metode untuk membedakan antara perangkat dengan alamat MAC asli dengan alamat MAC palsu (spoofing), dengan menganalisa variabel – variabel Channel State Information (CSI), yang terdapat pada Wi-Fi, berupa amplitudo dan fasa sinyal, yang memiliki kemungkinan perbedaan antara perangkat asli dan spoofing menggunakan metode Deep Learning (Peng Jiang, et al., 2018). Metode berikut dilakukan di physical layer pada lapisan OSI layer, karena variabel amplitude dan phase dari sinyal Wi-Fi berada pada lapisan berikut. Sayangnya, metode berikut hanya bekerja pada jaringan yang menggunakan Wi-Fi berbasis modulasi OFDM, dan tidak pada jaringan LAN dengan kabel ethernet.

Selain anonimitas dan privasi, pengguna menggunakan fitur tersebut untuk melewati filter MAC, dimana pada beberapa kasus administrator jaringan membatasi akses ke jaringan tersebut baik dengan skema blacklist, dimana perangkat dengan MAC address yang dicatat oleh filter blacklist tidak dapat mengakses jaringan tersebut maupun dengan skema whitelist, dimana hanya perangkat dengan MAC yang dicatat oleh filter whitelist yang diperbolehkan mengakses jaringan tersebut.

2. METODE PENELITIAN

Dalam penelitian ini, dilakukan pengujian pada dua lingkungan, yaitu lingkungan fisik dan lingkungan virtual. Lingkungan fisik terdiri dari perangkat keras mikrokontroler NodeMCU, smartphone Android, dan komputer. Sedangkan lingkungan virtual terdiri dari perangkat keras *virtual machine* (VM) yang memiliki spesifikasi yang sama.

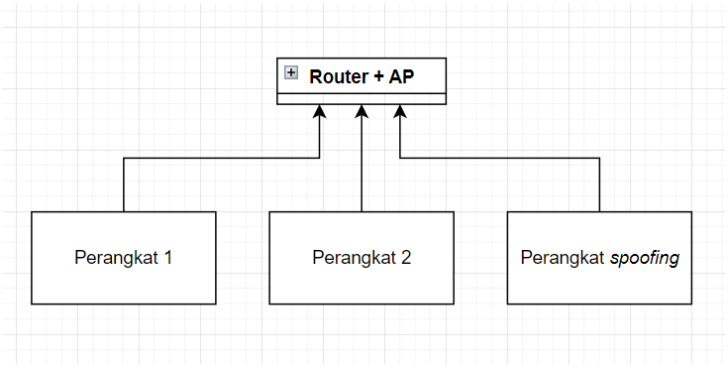
Perangkat yang mendukung penggantian alamat MAC menjadi pertimbangan utama. Pada lingkungan fisik, NodeMCU dengan mikrokontroler ESP8266 dipilih karena mampu mengubah alamat MAC secara langsung. Sedangkan pada lingkungan virtual, VirtualBox digunakan karena memiliki fitur untuk mengubah alamat MAC.

Masing – masing penelitian untuk memperoleh *latency ping*, *latency traceroute*, jumlah *hop*, keberhasilan *ping*, keberhasilan *traceroute* dan keberhasilan HTTP GET dilakukan sebanyak 10 kali

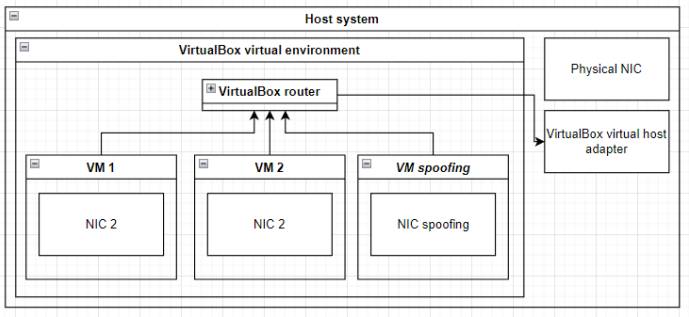
2.1 Rancangan sistem

Infrastruktur jaringan untuk pengujian di lingkungan fisik dengan Perangkat 1, Perangkat 2, dan Perangkat yang melakukan *spoofing* alamat MAC seperti pada Gambar 1. Kemudian untuk lingkugan virtual dengan VM OpenWrt sebagai *router*, VM 1, VM 2, dan VM yang melakukan spoofing seperti pada Gambar 2.

ceived June 1st,2012; Revised June 25th, 2012; Accepted July 10th, 2012



Gambar 1 Topologi pertama (fisik)



Gambar 2 Topologi kedua (virtual environment)

Pada penelitian lingkungan fisik, digunakan library yang digunakan untuk menunjang penelitian pada jaringan fisik. ESP8266WiFi.h digunakan untuk mengakses modul Wi-Fi, serta menjalankan fungsi konektivitas jaringan, mengelola koneksi, mengirim data, dan mendeteksi jaringan WiFi. ESP8266Ping.h digunakan untuk menjalankan fungsi ping, ESP8266HTTPClient.h untuk menjalankan fungsi permintaan HTTP GET sebagai client, WiFiUDP.h digunakan untuk menjalankan fingsi pengiriman atau penerimaan paket UDP, serta ESP8266WebServer digunakan untuk menjalankan fungsi server HTTP.

3. HASIL DAN PEMBAHASAN

3.1 Percobaan pada lingkungan virtual

Pada Percobaan 3.1, dilakukan percobaan pada lingkungan virtual dengan perangkat lunak pendukung VirtualBox.

3.1.1 Keadaan Normal (tidak ada duplikasi alamat MAC)

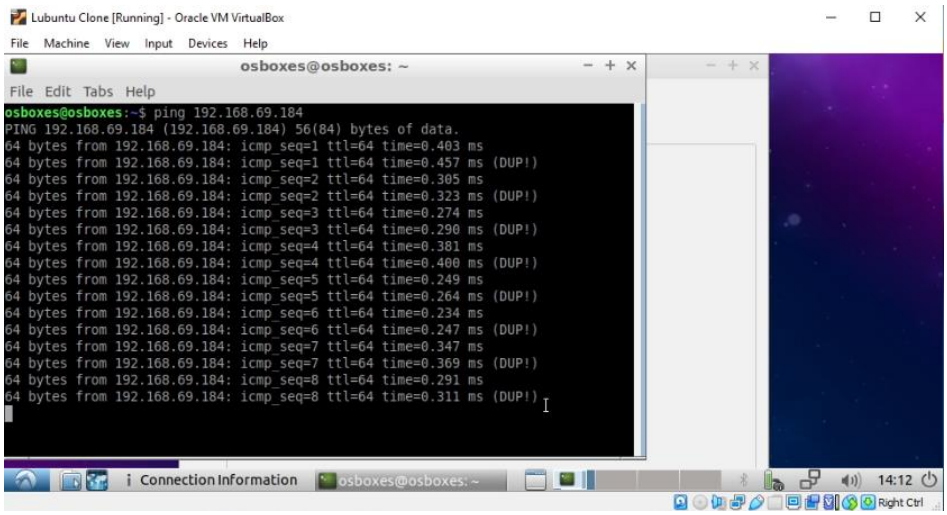
Percobaan jaringan menggunakan VirtualBox dalam keadaan ideal menghasilkan hasil yang baik. Semua VM, termasuk VM OpenWrt, memiliki alamat MAC yang unik. VM OpenWrt yang bertindak sebagai *router* dan *server* DHCP memberikan alamat IP yang unik untuk kedua VM, yaitu 192.168.69.184 dan 192.168.69.137 dengan alamat IP *gateway* 192.168.69.1.

Pengujian *ping* ke VM OpenWrt dari kedua VM membutuhkan waktu kurang dari 1 milisekon. Pengujian ping dari VM pertama ke VM kedua dan sebaliknya juga menghasilkan *latency* di bawah 1 milisekon.

Pengujian HTTP GET ke VM OpenWrt menggunakan browser Mozilla Firefox juga berhasil. *Login page* gateway dapat diperoleh dengan baik.

3.1.2 Keadaan dua VM dengan alamat MAC sama dan satu VM pengamat

Dalam percobaan dengan VirtualBox, dua buah VM dengan alamat MAC yang sama diberikan alamat IP yang sama oleh server DHCP. VM pengamat dapat melakukan *ping* ke kedua VM tersebut, tetapi hasilnya adalah respon ganda dengan pesan "(DUP!)", seperti pada Gambar 3. *Ping* dari kedua VM tersebut ke VM pengamat tampak normal. Percobaan HTTP GET ke *webpage* OpenWrt dari ketiga VM menunjukkan bahwa hanya VM pengamat yang dapat menampilkan webpage tersebut, sedangkan kedua VM dengan alamat MAC yang sama gagal menampilkannya.



Gambar 3 Pesan "DUP!"

3.1.3 Keadaan VM duplicate dengan VM OpenWrt

Pada percobaan collision MAC VM *duplicate* dengan VM OpenWrt, ditemukan bahwa VM dengan alamat MAC yang sama dengan *router* OpenWrt akan mengalami *latency* yang lebih tinggi saat melakukan ping dan traceroute ke VM lain. Selain itu, pada hasil *traceroute* menuju VM OpenWrt, terdapat dua buah *hop* pada salah satu VM unik, yaitu VM OpenWRT pada *hop* pertama, dan VM duplicate pada *hop* kedua.

Analisa melalui program Wireshark menunjukkan bahwa pada saat melakukan ping ke VM OpenWrt dan VM lainnya, terdapat dua kali respon pong untuk setiap kali ping, seperti pada Tabel 1. Hal ini menunjukkan bahwa terjadi duplikasi paket ICMP pada VM duplicate.

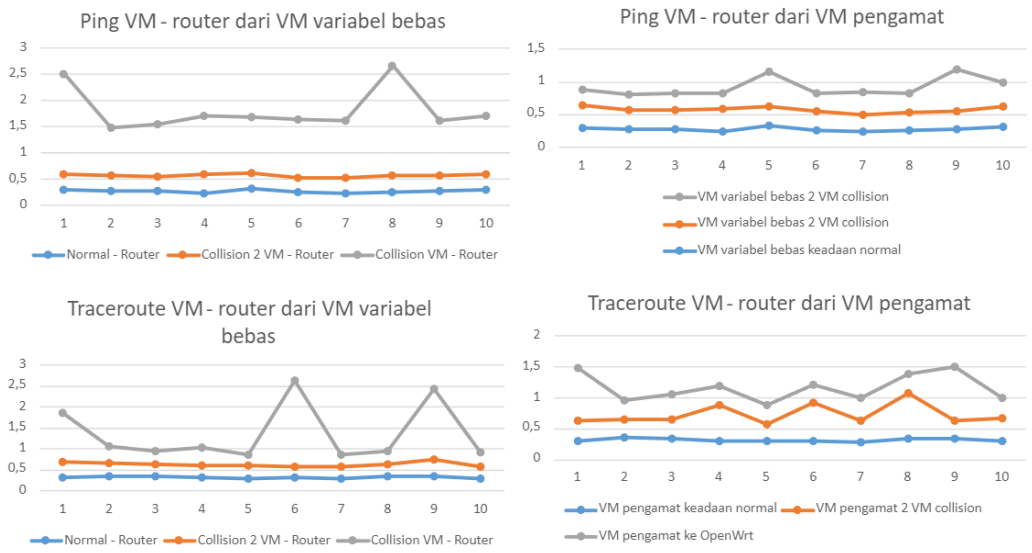
No	Time	Source	Destination	Protocol	Length	Info
1	0	192.168.69.184	192.168.69.181	ICMP	100	Echo (ping) request, seq=1/256, ttl=64
2	0.00016	192.168.69.181	192.168.69.184	ICMP	100	Echo (ping) reply, seq=1/256, ttl=64
3	0.00027	192.168.69.181	192.168.69.184	ICMP	100	Echo (ping) reply, seq=1/256, ttl=64
4	1.01348	192.168.69.184	192.168.69.181	ICMP	100	Echo (ping) request, seq=2/512, ttl=64
5	1.01375	192.168.69.181	192.168.69.181	ICMP	100	Echo (ping) reply, seq=2/512, ttl=64
6	1.01377	192.168.69.1	192.168.69.184	ICMP	128	(Redirect for host)
7	1.01386	192.168.69.181	192.168.69.184	ICMP	100	Echo (ping) reply, seq=2/512, ttl=64
…	…	…	…	…	…	…

Tabel 1 Hasil pengujian Wireshark

Berdasarkan hasil percobaan dan analisa, dapat disimpulkan bahwa collision MAC VM - *Router* dapat menyebabkan *latency* yang lebih tinggi dan duplikasi paket ICMP.

Statistik pada Gambar 4 menujukkan bahwa adanya peningkatan *latency* untuk seluruh skenario penelitian, dimana terjadi duplikasi alamat MAC.

ceived June 1st,2012; Revised June 25th, 2012; Accepted July 10th, 2012



Gambar 4 Statistik *latency ping* dan *traceroute* pada penelitian lingkungan virtual

3.2 Percobaan pada lingkungan fisik

Pada bab ini, dilakukan percobaan pada lingkungan fisik, dengan perangkat keras berupa NodeMCU sebagai perangkat variabel bebas, dimana perangkat ini bertindak sebagai perangkat yang dapat mengubah alamat MAC, serta sebuah komputer dengan NIC Intel AX200 dan Android sebagai *client* dengan *router* Mi Router 4A.

3.2.1 Keadaan Normal (tidak ada duplikasi alamat MAC)

Pada percobaan menggunakan router Mi Router 4A, ping NodeMCU ke router berkisar antara 3ms – 58ms, ping komputer ke router adalah 1ms, serta ping dari perangkat Android ke router adalah 7 – 105ms. Pengujian GET berhasil dilakukan pada server Apache pada komputer dan pada homepage router. *Latency* pada lingkungan nyata lebih bervariatif dan memiliki nilai rata-rata diatas rata-rata dibandingkan pada lingkungan virtual.

3.2.2 Keadaan MAC NodeMCU duplicate dengan router

Pada kondisi MAC NodeMCU duplicate dengan router, NodeMCU tidak dapat melakukan ping dan HTTP GET ke semua perangkat dalam suatu jaringan. Ping dari komputer hanya berhasil menuju router dengan *latency* rata-rata yang lebih besar, sedangkan ping dari Android hanya berhasil menuju router dan komputer. Untuk HTTP GET, terjadi kegagalan untuk kasus HTTP GET dari NodeMCU ke semua perangkat, komputer ke NodeMCU dan Android, dan Android ke NodeMCU.

3.2.3 Keadaan MAC NodeMCU duplicate dengan client (Android)

Pada keadaan MAC NodeMCU sama dengan perangkat Android, *ping* menunjukkan RTO pada *ping* dari NodeMCU ke seluruh perangkat dan ping dari perangkat lain yang relatif lebih besar dari data keadaan normal, seperti pada Gambar 5. Pengujian HTTP GET gagal pada saat

melakukan HTTP GET dari NodeMCU ke semua perangkat, komputer menuju NodeMCU, dan Android menuju NodeMCU dan komputer.

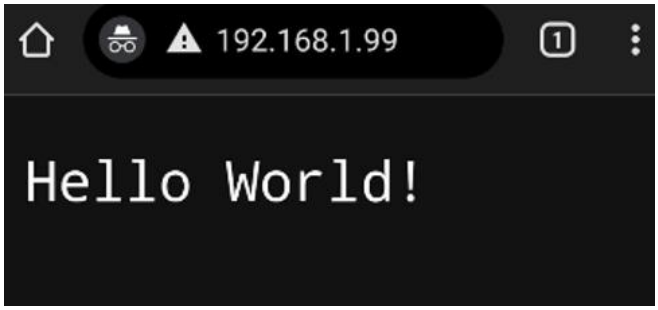
3.2.4 Keadaan MAC NodeMCU duplicate dengan client (Komputer)

Pada keadaan MAC NodeMCU sama dengan perangkat komputer, terjadi *collision* pada jaringan. Hal ini menyebabkan *ping* dari NodeMCU ke semua perangkat, dari komputer menuju Android, dan *ping* perangkat Android menuju router dan komputer relatif lebih besar dibandingkan pada keadaan normal.

Selain itu, HTTP GET dari NodeMCU ke komputer dan Android, komputer ke NodeMCU dan Android, serta Android ke NodeMCU gagal dilakukan, seperti pada Gambar 5. Namun, pada kejadian HTTP GET dari Android ke komputer, respons yang ditampilkan adalah "Hello World!", yang merupakan *server* dari NodeMCU. *Collision* juga menyebabkan rata-rata *latency* pada keadaan *collision* diatas rata-rata keadaan normal.

```
23:56:09.667 -> HTTP server started.
23:56:09.667 -> Pinging 192.168.1.11
23:56:14.672 -> Ping failed!
23:56:14.672 -> Pinging 192.168.1.1
23:56:19.737 -> Ping failed!
23:56:19.737 -> Pinging 192.168.1.74
23:56:24.794 -> Ping failed!
23:56:24.794 -> Pinging 192.168.1.99
23:56:29.897 -> Ping failed!
23:56:29.897 -> HTTP GET request to: http://192.168.1.11
23:56:35.129 -> HTTP GET request failed. Error code: -1
23:56:35.129 -> HTTP GET request to: http://192.168.1.1
23:56:35.177 -> HTTP GET request successful. Response code: 200
23:56:35.318 -> Response (first 10 characters):
23:56:35.318 ->
23:56:35.318 -> HTTP GET request to: http://192.168.1.74:1111
23:56:40.525 -> HTTP GET request failed. Error code: -1
23:56:40.525 -> HTTP GET request to: http://192.168.1.99
23:56:45.671 -> HTTP GET request failed. Error code: -1
```

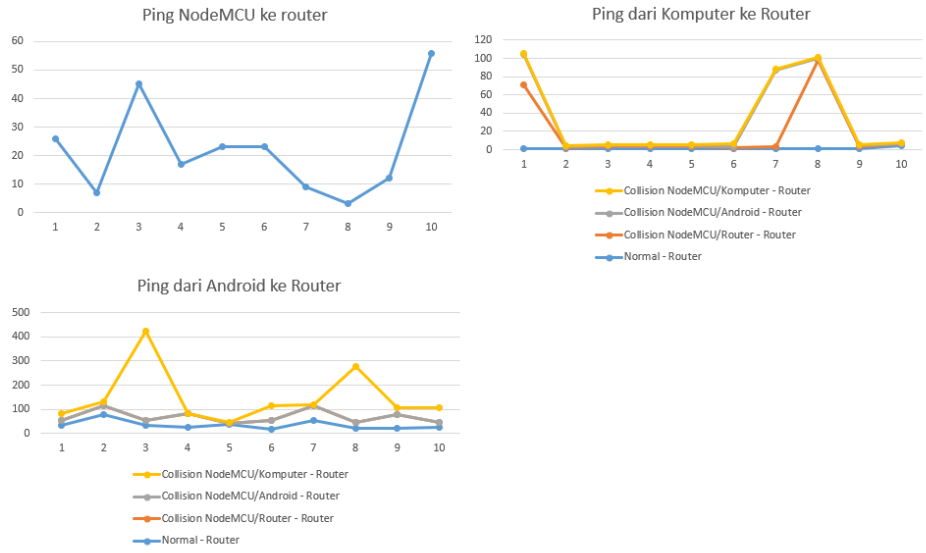
Gambar 5 *Ping* dan HTTP GET dari NodeMCU ke semua perangkat



Gambar 6 "Hello World!" dari NodeMCU

Statistik untuk penelitian pada lingkungan fisik ditampilkan pada Gambar 7, dimana hasilnya juga terdapat peningkatan *latency* pada keadaan *collision* dibandingkan dengan pada keadaan normal.

ceived June 1st,2012; Revised June 25th, 2012; Accepted July 10th, 2012



Gambar 7 Statistik *latency ping* pada penelitian lingkungan fisik

4. KESIMPULAN

Karakteristik anomali yang dihasilkan dari penelitian MAC *address collision* berbeda-beda pada lingkungan virtual VirtualBox dan percobaan pada lingkungan fisik. Pada lingkungan virtual VirtualBox, anomali yang paling terlihat adalah munculnya pesan "DUP!" pada *response* dari *ping*. Pada lingkungan fisik, anomali yang paling terlihat adalah bertambahnya *latency*, bahkan gagalnya *ping*.

Pada pengujian *ping*, fenomena MAC *address collision* dapat ditemukan dengan munculnya pesan "DUP!" pada *response* dari ping. Pada pengujian *traceroute*, terjadi kenaikan nilai *latency*, serta beberapa kali melalui 2 buah *hop*. Pada pengujian HTTP GET, semua perangkat di VirtualBox berhasil menampilkan *webpage*, namun di lingkungan fisik, hanya beberapa perangkat yang berhasil.

Peningkatan latency yang terjadi pada pengujian *ping* dan *traceroute* disebabkan oleh adanya duplikasi paket data yang dikirim oleh perangkat yang mengalami MAC address collision. Pada pengujian HTTP GET, kegagalan konektivitas dan respon HTTP GET yang tidak sesuai disebabkan oleh duplikasi paket data yang dikirim oleh perangkat yang mengalami MAC address collision.

5. SARAN

Berdasarkan hasil penelitian, dapat disimpulkan bahwa MAC *address collision* dapat menyebabkan peningkatan *latency*, kegagalan konektivitas, dan respon HTTP GET yang tidak sesuai. Untuk memperkuat hasil penelitian, perlu dilakukan pengujian ulang pada lingkungan fisik dengan perangkat yang identik. Selain itu, pengujian dapat dilakukan lebih dari 10 kali untuk mendapatkan hasil yang lebih akurat dan dapat digeneralisasi. Untuk mendapatkan hasil yang lebih komprehensif, pengujian pada VirtualBox dapat menggunakan *hub*, bukan *switch*, dimana pengamat Wireshark sebagai pengamat sudut pandang ketiga tidak dapat menganalisa *traffic* yang menggunakan *switch.*

DAFTAR PUSTAKA

[1] Alasdair Allan. 2017. “List of MAC addresses with vendors identities”. GitHub. https://gist.github.com/aallan/b4bb86db86079509e6159810ae9bd3e4.
[2] Android 6.0 changes. 2015. https://developer.android.com/about/versions/marshmallow/android-6.0-changes.html, 2015.
[3] Cardenas, Edgar D. 2013. “MAC Spoofing - An Introduction”. GIAC Security Essentials Certification.
[4] Cisco. 2015. “Configuring Port Security”.
[5] F. Baker. 2015. Baker, F (ed.). “Requirements for IP Version 4 Routers”. p. 52. RFC 1812.
[6] Institute of Electrical and Electronics Engineers. 2004. “Overview and Guide to the IEEE 802 LMSC”. https://grouper.ieee.org/groups/802/802%20overview.pdf.
[7] Jarmo Mölsä. 2006. "Mitigating DoS Attacks against the DNS with Dynamic TTL Values". Networking Laboratory, Helsinki University of Technology. http://lib.tkk.fi/Diss/2006/isbn9512282151/article2.pdf.
[8] Jean-François Determe, Sophia Azzagnuni, François Horlin, Philippe De Doncker. 2020. "MAC Address Anonymization for Crowd Counting". BEAMS-EE, Université Libre de Bruxelles, 1050 Brussels, Belgium. OPERA Wireless Communications Group, Université Libre de Bruxelles, 1050 Brussels, Belgium.
[9] Junade Ali, Vladimir Dyo. 2020. "Practical Hash-Based Anonymity for MAC Addresses". Cloudflare Inc, London, UK. University of Bedfordshire, Luton, UK.
[10] Mathy Vanhoef, Célestin Matte, Mathieu Cunche, Leonardo S. Cardoso, Frank Piessens. 2016. "Why MAC Address Randomization is not Enough: An Analysis of Wi-Fi Network Discovery Mechanisms". iMinds-Distrinet, KU Leuven, Univ Lyon, INSA Lyon, Inria, CITI, France.
[11] Oracle Corporation. 2022. "Changelog for VirtualBox 6.1". https://www.virtualbox.org/wiki/Changelog-6.1.
[12] Pallavi Asrodia, Hemlata Patel. 2012. "Analysis of Various Packet Sniffing Tools for Network Monitoring and Analysis". Department of Computer Science and Engineering, Jawaharlal Institute of Technology, Borawan, Khargone.
[13] Peng Jiang, Hongyi Wu, Cong Wang, Chunsheng Xin. 2018. "Virtual MAC Spoofing Detection through Deep Learning". Department of ECE, Old Dominion University, Norfolk, VA, USA.
[14] S. Pavithirakini, D.D.M.M. Bandara, C.N. Gunawardhana, K.K.S. Perera, B.G.M.M. Abeyrathne, Dhishan Dhammearatchi. 2016. "Improve the Capabilities of Wireshark as a Tool for Intrusion Detection in DoS Attacks". Sri Lanka Institute of Information Technology Computing Pvt. Ltd.
[15] Tang Yong, Liu Xiaoyan. 2007. "System, Node as Well as Method for Detecting MAC Address Collision in Loop Network". Huawei Technologies Co Ltd.
[16] Xerox Corporation. 1984. "Xerox System Integration Standard 098404 - Authentication Protocol".