Serangan *Universal Forgery* pada Skema Fungsi *Hash* F_t, LightMAC_Plus, dan LightMAC_Plus2 Berbasis Algoritme DLBCA

Muhammad Ridwan[1)], Andriani Adi Lestari[2)]
(1) *Jurusan Kriptografi, Politeknik Siber dan Sandi Negara, muhammad.ridwan@bssn.go.id*
(2) *Politeknik Siber dan Sandi Negara, andriani.adi@poltekssn.ac.id*

Abstrak

Liu dan Liu membuat serangan pemalsuan pesan pada tahun 2017. Serangan pemalsuan pesan yang dibuat oleh Liu dan Liu dikenal sebagai serangan universal forgery yang dapat diterapkan pada beberapa skema fungsi hash dan authenticated encryption. Fungsi hash yang merupakan variasi dari skema LightMAC dan/atau PMAC diklaim tidak aman oleh Liu dan Liu. Skema F_t merupakan variasi dari fungsi hash LightMAC apabila skema F_t menggunakan skema LightMAC sebagai basis fungsi hash-nya. Berbeda dengan skema F_t, `LightMAC_Plus` dan `LightMAC_Plus2` merupakan variasi dari skema LightMAC dan PMAC. Oleh karena itu, pada penelitian ini dilakukan pembuktian dari klaim yang dibuat oleh Liu dan Liu dengan menerapkan serangan universal forgery pada skema fungsi hash F_t, `LightMAC_Plus`, dan `LightMAC_Plus2`. Metode serangan yang dilakukan pada penelitian ini yaitu menerapkan serangan universal forgery dengan menggunakan sepuluh sampel pesan acak yang dikombinasikan dengan lima himpunan kunci. Hasil penerapan serangan universal forgery yaitu tidak ditemukan satu pun pesan palsu dari keseluruhan serangan yang dilakukan. Hal ini diperkuat dengan pembuktian secara matematis bahwa strategi serangan universal forgery tidak dapat diterapkan pada ketiga skema fungsi hash tersebut. Berdasarkan data hasil serangan universal forgery dan pembuktian secara matematis, dapat disimpulkan bahwa skema fungsi hash F_t, `LightMAC_Plus`, dan `LightMAC_Plus2` tahan terhadap serangan universal forgery.

Kata kunci: Fungsi *Hash* F_t, LightMAC_Plus, LightMAC_Plus2, *Universal Forgery*

1 PENDAHULUAN

Fungsi *hash* merupakan suatu fungsi yang memetakan input *string* dengan panjang beragam menjadi *output* dengan panjang tetap [1]. Keutuhan data dan autentikasi pesan merupakan dua hal yang menjadi fokus penggunaan fungsi *hash*. Terdapat dua sifat yang harus dipenuhi oleh fungsi *hash* yaitu kompresi dan mudah dihitung [2]. Fungsi *hash* diklasifikasikan menjadi fungsi *hash* tanpa kunci dan fungsi *hash* dengan kunci. MAC merupakan salah satu bentuk klasifikasi fungsi *hash* dengan kunci yang memiliki sifat tambahan yaitu tahan terhadap komputasi. Sifat tahan terhadap komputasi berarti bahwa apabila diberikan nol atau beberapa pasangan pesan-nilai MAC $(M_i, H_K(M_i))$ maka sulit secara komputasi untuk memperoleh pasangan pesan-nilai MAC $(M,\ H_K(M))$ dengan $M \neq M_i$. Apabila MAC tidak memenuhi sifat tahan terhadap komputasi maka MAC tersebut rentan terhadap pemalsuan pesan [2].

Key recovery resistance dan *unforgeability* merupakan persyaratan keamanan yang harus dipenuhi oleh MAC. *Key recovery resistance* berarti bahwa penyerang tidak dapat memperoleh nilai kunci rahasia yang digunakan dengan komputasi kurang dari 2^k. Sementara itu, *unforgeability* berarti bahwa penyerang tidak dapat membangkitkan pasangan pesan dan nilai *hash* (M, T) dengan komputasi kurang dari 2^n untuk suatu nilai *hash* T yang valid [3].

Pada tahun 2017, Liu dan Liu membuat skema serangan *universal forgery* menggunakan permasalahan hari lahir pada skema MAC berbasis *block cipher*. Liu dan Liu menyatakan bahwa terdapat beberapa skema MAC yang diklaim tidak aman diantaranya CBC-MAC, XCBC, EMAC, OMAC, CMAC, PC-MAC, MT-MAC, PMAC, PMAC dengan *parity*, LightMAC, dan variasi dari skema-skema tersebut [4].

Pada tahun 2016, Iwata dan Minematsu mempublikasikan skema F_t. Skema F_t yang menggunakan LightMAC sebagai basis fungsi *hash*-nya merupakan *Pseudo Random Functio* (PRF) dengan level keamanan $O\left(2^{tn/(t+1)}\right)$. Satu tahun kemudian, Naito mengembangkan skema F_t menjadi skema `LightMAC_Plus` dan `LightMAC_Plus2` yang merupakan kombinasi dari skema LightMAC dan `PMAC_Plus` [5].

Skema F_t, `LightMAC_Plus`, dan `LightMAC_Plus2` merupakan skema-skema baru yang merupakan variasi dari skema LightMAC dan PMAC. Skema-skema tersebut diklaim tidak aman oleh Liu dan Liu. Pada penelitian ini dilakukan serangan *universal forgery* pada skema F_t, `LightMAC_Plus`, dan `LightMAC_Plus2`

berbasis algoritme *lightweight block cipher Desain* 32 bit *Lightweight Block Cipher Algorithm* (DLBCA) untuk membuktikan klaim yang dibuat oleh Liu dan Liu.

Algoritme DLBCA dibuat oleh Salim AlDabbagh pada tahun 2017. DLBCA memerlukan input pesan sepanjang 32 bit dan kunci sepanjang 80 bit [6]. Pemilihan algoritme DLBCA pada penelitian ini didasarkan pada ukuran bit input yang relatif kecil dibandingkan dengan beberapa algoritme *lightweight block cipher* lainnya. Hal tersebut dikarenakan ukuran bit input berpengaruh pada ko**mputasi. DLBCA memiliki panjang output 32 bit sehingga output dari** skema F_t, `LightMAC_Plus`, dan `LightMAC_Plus2` memiliki panjang 32 bit. Salah satu proses serangan *universal forgery* yaitu mencari *collision* dari himpunan G_1 dan G_2 dengan jumlah elemen masing-masing himpunan yaitu $2^{n/2}$. Dengan demikian, penerapan serangan *universal forgery* pada masing-masing skema F_t, `LightMAC_Plus`, dan `LightMAC_Plus2` membutuhkan komputasi sebesar $2 \times 2^{32/2} = 2^{17}$ sehingga masih memungkinkan untuk dilakukan simulasi.

2 LANDASAN TEORI

Bagian ini mencantumkan teori-teori yang terkait dengan penelitian yang dilakukan.

2.1 Algoritme DLBCA

DLBCA merupakan algoritme *lightweight block cipher* yang dibuat oleh Salim AlDabbagh pada tahun 2017 [6]. Algoritme DLBCA memerlukan input *plaintext* sepanjang 32 bit dan kunci *master* sepanjang 80 bit. Kunci *master* dijadikan sebagai input pada proses penjadwalan kunci. Hasil dari penjadwalan kunci yaitu enam belas kunci putaran berukuran 32 bit yang dinotasikan dengan $key_0, key_1, key_2, \dots, key_{15}$. Proses penghitungan *ciphertext* algoritme DLBCA dilakukan sebanyak lima belas putaran dan disetiap putaran terdapat operasi seperti: *substitution box*, permutasi bit, rotasi, dan XOR dengan kunci putaran. Nilai akhir *ciphertext* merupakan hasil XOR antara *output* putaran ke-15 dengan kunci putaran terakhir.

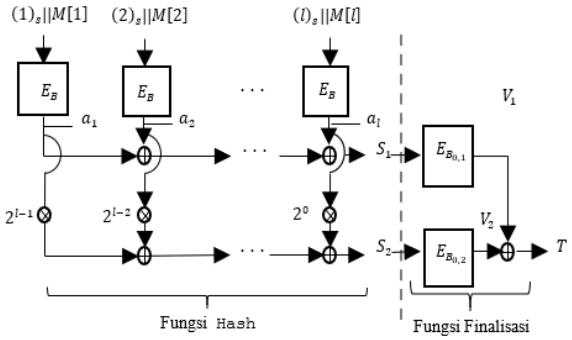
2.2 Skema Fungsi Hash F_t

Pada tahun 2016, Iwata dan Minematsu melakukan modifikasi minor pada skema *Authenticated Encryption* (AE) bernama GCM-SIV menjadi skema GCM-SIV1 [7]. Selanjutnya, Iwata dan Minematsu juga membuat skema GCM-SIV2 yang menjalankan dua fungsi GCM-SIV1 secara paralel dan menggabungkan kedua hasil fungsi tersebut secara sederhana. GCM-SIV2 memiliki level keamanan mencapai $2^{2n/3}$. Kemudian, Iwata dan Minematsu membuat skema umum $\mathrm{GCM-SIV_t}$ yang menjalankan t fungsi GCM-SIV1 secara paralel. $\mathrm{GCM-SIV_t}$ dengan $t \geq 3$ dinyatakan memiliki level keamanan hingga $2^{tn/(t+1)}$ [8].

Skema $\mathrm{GCM-SIV_t}$ menggunakan fungsi pembangkit *tag* F_t yang merupakan MAC/PRF berbasis *block cipher* dengan level keamanan $2^{tn/(t+1)}$ untuk $t \geq 2$. Skema F_t mengoperasikan fungsi *hash* H_{L_i} sebanyak t kali secara paralel dengan L_i adalah kunci rahasia, $1 \leq i \leq t$ sehingga diperoleh t *output* fungsi *hash* H_{L_i}. *Output* fungsi *hash* H_{L_i} dijadikan sebagai input t buah *block cipher E* dengan kunci $A_i, 1 \leq i \leq t$. Nilai MAC dari skema F_t adalah hasil XOR dari t *output block cipher E* [5].

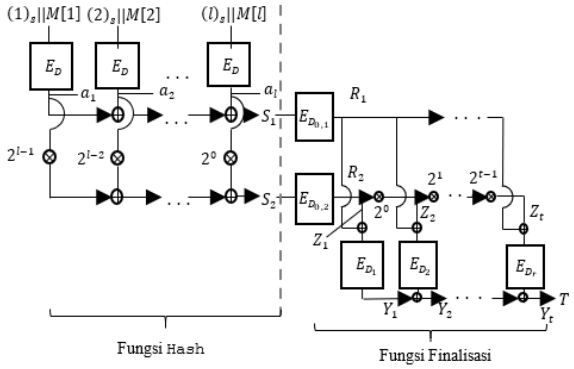
2.3 Skema Fungsi Hash LIGHTMAC_PLUS dan LIGHTMAC_PLUS2

Fokus pertama Naito adalah membuat skema MAC yang lebih efisien dibandingkan dengan F_2 dan memenuhi level keamanan $O(2^{2n/3})$. Naito mengkombinasikan skema LightMAC dan `PMAC_Plus` menjadi skema `LightMAC_Plus`. `LightMAC_Plus` memerlukan satu kali penggunaan fungsi *block cipher* untuk setiap blok pesan sedangkan F_2 memerlukan dua kali penggunaan fungsi *block cipher* untuk setiap blok pesan sehingga skema `LightMAC_Plus` dinyatakan lebih efisien dibandingkan dengan F_2 [5]. Detail skema `LightMAC_Plus` dapat dilihat pada Gambar 1.



Gambar 1. Skema `LightMAC_Plus`

Kemudian, perbaikan efisiensi terhadap skema F_t dengan $t \geq 3$ dan level keamanan $O(2^{tn/(t+1)})$ diwujudkan dalam skema `LightMAC_Plus2`. Skema `LightMAC_Plus2` menggunakan fungsi `Hash` yang sama dengan `LightMAC_Plus`. Oleh karena itu, fungsi `Hash` yang digunakan dinyatakan t kali lebih cepat dari F_t. Fungsi finalisasi pada skema `LightMAC_Plus2` merupakan fungsi XOR dari *output* t *keyed block cipher* dengan tujuan untuk memastikan keacakan dari nilai MAC yang dihasilkan. Input pada fungsi finalisasi merupakan *output* dari fungsi `Hash` dengan panjang $2n$ bit dan menghasilkan *output* sepanjang n bit [5]. Detail skema `LightMAC_Plus2` dapat dilihat pada Gambar 2.



Gambar 2. Skema LightMAC_Plus2

2.4 Serangan Universal Forgery

Serangan *universal forgery* merujuk pada [4]. Liu dan Liu memperkenalkan skema serangan *generic universal forgery* menggunakan serangan hari lahir terhadap beberapa skema MAC berbasis *block cipher*. Skema serangan yang dibuat membutuhkan kompleksitas sebesar $O(2^{n/2})$. Hasil dari serangan tersebut menunjukkan bahwa beberapa skema MAC dinyatakan tidak aman. Beberapa skema MAC yang dinyatakan tidak aman yaitu CBC-MAC, XCBC, EMAC, OMAC, CMAC, PC-MAC, MT-MAC, PMAC, PMAC dengan *parity*, LightMAC, dan variasi dari skema-skema tersebut [4].

Generic universal forgery dilakukan dengan menggunakan *generic birthday attack* pada dua himpunan G_1 dan G_2 untuk suatu pesan $x_1\|x_2\|$ ... $\|x_l$, dengan $l \geq 2$. Pertama, bangkitkan $2^{n/2}$ blok pesan x_2^i pada himpunan G_1, dengan blok pesan x_1 bernilai tetap dan $i \leq 2^{n/2}$. Kemudian, *query* $x_1\|x_2^i$ ke *oracle* MAC sehingga terdapat $2^{n/2}$ elemen nilai MAC yang berkorespondensi dengan blok pesan $x_2^i, T_i = MAC_K(x_1\|x_2^i)$ di himpunan G_1.

Kedua, bangkitkan $2^{n/2}$ blok pesan x_1^j pada himpunan G_2, dengan blok pesan x_2 bernilai tetap dan $j \leq 2^{n/2}$. Selanjutnya, *query* $x_1^j\|x_2$ ke *oracle* MAC sehingga terdapat $2^{n/2}$ elemen nilai MAC yang berkorespondensi dengan blok pesan $x_1^j, T_j = MAC_K(x_1^j\|x_2)$ di himpunan G_2. Bandingkan nilai MAC pada himpunan G_1 dan G_2 sehingga ditemukan nilai $T_i = T_j$ untuk suatu nilai i, j dengan peluang yang besar dengan menggunakan *birthday paradox*.

Ketiga, *query* pesan $x_1^j\|x_2^i\|$ ... $\|x_l$ ke *oracle* MAC sehingga diperoleh nilai MAC yang berkorespondensi, T. Nilai MAC T juga berlaku untuk pesan $x_1\|x_2\|$ ... $\|x_l$ yang belum pernah dihitung oleh penyerang. *Universal forgery* berhasil dilakukan apabila nilai MAC T dari pesan $x_1^j\|x_2^i\|$ ... $\|x_l$ berlaku juga untuk pesan $x_1\|x_2\|$ ... $\|x_l$.

3 METODE PENELITIAN

Metode penelitian yang digunakan dalam penelitian ini adalah metode kepustakaan dan eksperimen. Metode kepustakaan dilakukan dengan mencari, mempelajari, dan memahami teori-teori yang berhubungan dengan penelitian yang dilakukan. Sumber-sumber yang digunakan berupa jurnal, buku, internet, maupun sumber lainnya. Sementara itu, metode eksperimen dilakukan dengan menerapkan serangan *universal forgery* pada skema fungsi *hash* F_t, LightMAC_Plus, dan LightMAC_Plus2 berbasis algoritme DLBCA menggunakan bahasa pemrograman C yang dibahas lebih detail pada subbab 4.1.

3.1 Pembatasan Masalah

Berikut beberapa pembatasan masalah pada penelitian yang dilakukan.

a. Ketiga skema fungsi *hash* F_t, LightMAC_Plus, dan LightMAC_Plus2 menggunakan parameter-parameter yang sama di antaranya sebagai berikut:
   1) *Counter* dengan panjang delapan bit;
   2) Pesan dengan panjang 120 bit;
   3) Implementasi serangan *universal forgery* dilakukan dengan menggunakan sepuluh sampel pesan acak dan lima himpunan kunci; dan
   4) Pembangkitan sampel pesan acak, himpunan kunci, serta modifikasi pesan pada blok pertama dan kedua menggunakan fungsi pembangkit acak *Mersenne Twister* dengan *seed* yang berbeda.
b. Simulasi serangan pada skema F_t dan LightMAC_Plus2 dibatasi dengan $2 \leq t \leq 3$.
c. Skema F_t menggunakan LightMAC sebagai basis fungsi *hash*-nya.

3.2 Tahapan Penelitian

Adapun langkah-langkah yang akan dilakukan dalam penelitian ini adalah sebagai berikut:

a. Telaah kepustakaan dengan cara mengumpulkan, mempelajari, dan memahami teori-teori yang bersumber dari jurnal, buku, internet, maupun sumber lainnya.
b. Membuat *source code* algoritme DLBCA, skema LightMAC, F_t, LightMAC_Plus, dan LightMAC_Plus2 menggunakan bahasa pemrograman C.
c. Membangkitkan sepuluh sampel pesan acak dan lima himpunan kunci serta modifikasi pesan blok pertama dan kedua menggunakan fungsi pembangkit acak *Mersenne Twister* dalam bahasa pemrograman C. Dimisalkan $1 \leq w \leq 5$. Iw serta Jw adalah himpunan kunci pada skema F_t untuk $t = 2$ dan $t = 3$. Himpunan kunci pada skema LightMAC_Plus adalah Kw. Sementara itu, himpunan kunci pada skema LightMAC_Plus2 untuk $t = 2$ dan $t = 3$ adalah Ow dan Pw. Daftar sampel pesan dan himpunan kunci dapat dilihat pada Tabel 1, Tabel 2, Tabel 3, Tabel 4, dan Tabel 5.

Tabel 1 Daftar Sampel Pesan Acak

Mi	$Mi[1]$	$Mi[2]$	$Mi[3]$	$Mi[4]$	$Mi[5]$
1	$0x24e882$	$0x953ff1$	$0xb671e6$	$0xe38ae6$	$0xb9fb02$
2	$0xa04084$	$0x95af1a$	$0x6f2fc1$	$0x9059f1$	$0xd5aa2b$
3	$0x5134c1$	$0x226469$	$0x7d0082$	$0xd4cfa5$	$0x1eee43$
4	$0x0f2bfa$	$0x5dd27c$	$0xde1b9e$	$0x85738a$	$0x0fd5fc$
5	$0xbc47db$	$0xceb17f$	$0xaacc14$	$0x578d99$	$0x5f0155$
6	$0x7535a8$	$0x4bc8aa$	$0x569e41$	$0x14dfd2$	$0xed2fb0$
7	$0x17210d$	$0xa69af1$	$0x52caa0$	$0x287b8d$	$0x42b678$
8	$0x0d6f7d$	$0x924fee$	$0x20bdbd$	$0x695bde$	$0xad16e7$
9	$0xeef8bb$	$0xb42636$	$0xd7f5e3$	$0x3ec56d$	$0xbf5690$
10	$0x75fbc1$	$0x662f0f$	$0x51fc29$	$0xb21a0f$	$0x1720de$

Tabel 2 Kunci Fungsi Hash Skema LightMAC_Plus dan LightMAC_Plus2

No	Kunci B dan D
1	$0x75bdc0ca1b64bf55cb86$
2	$0x464980cc5577715360fc$
3	$0x9e6cc0260b95beb69325$
4	$0xa22258a126aa8276da7c$
5	$0xa4067a8f69c06a165d4a$

Tabel 3 Daftar Kunci Fungsi Finalisasi Skema LightMAC_Plus serta Kunci $D_{0,1}$ dan $D_{0,2}$ Skema LightMAC_Plus2

No	$B_{0,1}/D_{0,1}$	$B_{0,2}/D_{0,2}$
1	$0xc7f59bbae39228dc850f$	$0xc21fb00bdddefc781152$
2	$0xa529124b0fbbc32ec5ef$	$0x8e36adf572a0cf879160$
3	$0x517de6c4d46efb2f4f13$	$0x7e7ece280c9719bf324a$
4	$0x9328d589ce23d374d329$	$0xc9e56503fb1e43858332$
5	$0x512852d18fab3ace5a3e$	$0xeb77db21e73f11e3ea88$

Tabel 4 Daftar Kunci D_1 dan D Skema LightMAC_Plus2 untuk $t = 2$

No	K_1	K_2
1	$0x6deae300e5693dcd09e6$	$0xe77a4f68410e2f480016$
2	$0xc9993ab0bd678c01cf57$	$0x6a77cc32e3c79de43b33$
3	$0xd58287c1171d0f891921$	$0x20a893d3c9216814ca20$
4	$0x20716dcdc87116deea1b$	$0xd714711f0f4644c30e7b$
5	$0xa5413d4670a101bdcfeb$	$0x84780e10100e14b0ada4$

Tabel 5 Daftar Kunci $D_1, \ldots, D_3$ Skema LightMAC_Plus2 untuk $t = 3$

No	K_1	K_2	K_3
1	$0xdd458d46a75a3f75e494$	$0x3c6d14b599956251db6f$	$0x5b25037af2c956c29c73$
2	$0x548f14c921d6629d465c$	$0xd03b73e5046422676354$	$0x580f8bebd6120cb4c57d$
3	$0xd1935b014da5010bee03$	$0xeae3fee2f2b185ea95e9$	$0x707ccb26af77dbbc373c$
4	$0x5f9c590a02b9adf81caf$	$0x1804346b795f25609e01$	$0x647a9cdcaff52dd2bd3e$
5	$0xcd05a26b3393ac6dea1c$	$0xfe3ea93e72651cc3804b$	$0xa7227bec43b76de6c7af$

d. Melakukan simulasi serangan *universal forgery* pada skema F_t, `LightMAC_Plus`, dan `LightMAC_Plus2` berbasis algoritme DLBCA dalam bahasa pemrograman C. Setiap sampel pesan akan dilakukan serangan dengan menggunakan lima himpunan kunci berbeda.
e. Melakukan analisis secara matematis atas hasil simulasi serangan *universal forgery*.
f. Mengambil simpulan mengenai hasil serangan *universal forgery* pada skema F_t, `LightMAC_Plus`, dan `LightMAC_Plus2`.

4 HASIL DAN PEMBAHASAN

Bagian ini membahas mengenai penerapan serangan *universal forgery* dan analisis hasil serangan secara matematis. Pada bagian analisis, diasumsikan telah diperoleh pesan $M' = M[1]'\|M[2]'\|M[3]\|\cdots\|M[l]$ yang direkonstruksi setelah diperoleh *collison* antara nilai T_i dan T_j.

4.1 Penerapan Serangan *Universal Forgery*

Langkah pertama yang dilakukan pada penerapan serangan *universal forgery* adalah menghitung nilai MAC dari sepuluh sampel pesan $M1, M2, \ldots, M10$, menggunakan lima buah himpunan kunci berbeda, $Iw/Jw/Kw/Ow/Pw, 1 \le w \le 5$. Langkah selanjutnya adalah menghitung nilai MAC pada himpunan G_1 yang dinotasikan dengan $T_i = MAC_{Iw/Jw/Kw/Ow/Pw}(Mv[1]\|Mv[2]^i)$ dan himpunan G_2 yang dinotasikan dengan $T_j = MAC_{Iw/Jw/Kw/Ow/Pw}(Mv[1]^j\|Mv[2])$, dengan $1 \le i,j \le 2^{16}$, $1 \le w \le 5$, dan $1 \le v \le 10$. Blok pesan kedua pada proses penghitungan nilai MAC T_i dan blok pesan pertama pada proses penghitungan nilai MAC T_j , $1 \le i,j \le 2^{16}$, merupakan pesan modifikasi.

Setelah memperoleh nilai T_i dan T_j untuk setiap sampel pesan, kemudian dicari *collision* antara nilai T_i dan T_j yaitu nilai yang memenuhi $T_i = T_j$. Langkah berikutnya setelah menemukan *collision* $T_i = T_j$ yaitu menghitung nilai $T' = MAC_{Iw/Jw/Kw/Ow/Pw}\left(Mv[1]^j\middle\|Mv[2]^i\middle\|Mv[3]\|Mv[4]\|Mv[5]\right)$ dengan menggunakan blok pesan modifikasi yang bersesuaian dengan kondisi $T_i = T_j$.

Hasil akhir yang diperoleh yaitu hanya ditemukan *collision* antara T_i dan T_j dan tidak diperoleh satu pun pesan palsu yang berhasil direkonstruksi. Hal ini menunjukkan bahwa semua input blok pesan modifikasi yang dihasilkan dari *collision* pada himpunan G_1 dan G_2 tidak dapat direkonstruksi menjadi pesan palsu karena nilai MAC-nya berbeda dengan nilai MAC pesan asli.

4.2 Analisis Matematis Hasil Serangan pada Skema F_t

Dimisalkan pesan asli adalah M dengan nilai MAC-nya adalah T dan pesan yang akan direkonstruksi sebagai pesan palsu adalah M' dengan nilai MAC-nya adalah T'. Terdapat dua buah kasus perbandingan antara pesan M' dengan pesan M terhadap perolehan pesan palsu yang dijabarkan sebagai berikut.

a. Kasus pertama, nilai T sama dengan nilai T' yang menandakan bahwa M' adalah pesan palsu dari M yang memiliki kesamaan nilai MAC.
b. Kasus kedua, nilai T tidak sama dengan nilai T' yang menandakan bahwa nilai M' tidak dapat direkonstruksi sebagai pesan palsu dari M.

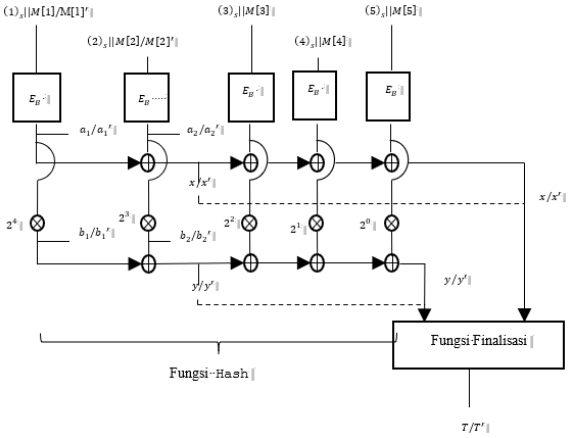
Pesan M' yang dihasilkan pada proses serangan *universal forgery* dapat menjadi pesan palsu M apabila memenuhi kasus pertama. Namun, hasil serangan *universal forgery* menunjukkan bahwa semua nilai M' yang diperoleh pada skema F_t termasuk dalam kasus kedua sehingga tidak ada yang dapat direkonstruksi sebagai pesan palsu M. Tidak diperolehnya nilai M' yang termasuk dalam kasus pertama dikarenakan oleh struktur dari skema F_t. Kesamaan nilai T' dengan T bergantung dari hasil XOR t *output block cipher* E. Total semua kemungkinan nilai T dan T' yaitu sebanyak 2^n. Pada penelitian ini diambil nilai $n = 32$ sehingga peluang nilai T' sama dengan nilai T sangatlah kecil yaitu $\frac{1}{2^n} = \frac{1}{2^{32}}$. Kecilnya peluang untuk mendapatkan pesan M' yang menghasilkan nilai $T' = T$ menyebabkan tidak diperolehnya satupun pesan M' dari hasil eksperimen yang memenuhi kasus pertama, melainkan termasuk dalam kasus kedua.

4.3 Analisis Matematis Hasil Serangan pada Skema LIGHTMAC_PLUS dan LIGHTMAC_PLUS2

Skema `LightMAC_Plus` dan `LightMAC_Plus2` terbagi menjadi dua bagian yaitu bagian fungsi `Hash` dan fungsi finalisasi. Bagian fungsi `Hash` skema `LightMAC_Plus2` sama dengan `LightMAC_Plus` sebagaimana ditunjukkan pada Gambar 3. Berdasarkan Gambar 3 diperoleh data sebagai berikut.

a. Nilai $a_1 = E_D(1_s\|M[1])$ berbeda dengan nilai $a_1' = E_D(1_s\|M[1]')$ karena nilai $[1] \ne M[1]'$.
b. Nilai $a_2 = E_D(2_s\|M[2])$ tidak sama dengan nilai $a_2' = E_D(2_s\|M[2]')$ karena nilai $M[2] \ne M[2]'$.
c. Nilai $b_1 = 2^4 \otimes (a_1)$ berbeda dengan nilai $b_1' = 2^4 \otimes (a_1')$ karena nilai $a_1 \ne a_1'$.
d. Nilai $b_2 = 2^4 \otimes (a_2)$ berbeda dengan nilai $b_2' = 2^4 \otimes (a_2')$ karena nilai $a_2 \ne a_2'$.
e. Nilai $x = a_1 \oplus a_2$ berkemungkinan bernilai sama atau berbeda dengan $x' = a_1' \oplus a_2'$.
f. Nilai $y = b_1 \oplus b_2$ berkemungkinan bernilai sama atau berbeda dengan $y' = b_1' \oplus b_2'$.

Berdasarkan data-data tersebut serta proses serangan *universal forgery* yang merekonstruksi suatu pesan M' yang terdiri dari dua blok pesan pertama yang sama dengan pesan M dan blok lainnya berbeda dari pesan M sebagaimana dijelaskan pada subbab 4.1 dapat dibuat suatu lemma sebagai berikut.



Gambar 3. Detail Fungsi `Hash` Skema `LightMAC_Plus` dan `LightMAC_Plus2`

Lemma 1. Diketahui bahwa bagian fungsi finalisasi skema `LightMAC_Plus` dan `LightMAC_Plus2` dengan $t \in \{2,3\}$ memerlukan dua buah input yang diperoleh dari bagian fungsi `Hash`. Diasumsikan terdapat suatu pesan $M' = M[1]'\|M[2]'\|M[3]\|\cdots\|M[l]$ yang berbeda dengan pesan asli $M = M[1]\|M[2]\|M[3]\|\cdots\|M[l]$. Apabila blok pesan ketiga hingga blok pesan terakhir pada pesan M dan M' diabaikan karena bernilai sama maka akan diperoleh *output* bagian fungsi `Hash` secara berturut-turut yaitu nilai (x, y) dan (x', y'). *Output* tersebut dijadikan input pada fungsi finalisasi. Terdapat empat buah kasus perbandingan nilai (x, y) dan (x', y') yang memberikan tiga kemungkinan hasil yang berbeda, yaitu diperoleh pesan palsu, tidak dapat diperoleh pesan palsu, atau keduanya.

Bukti:

Diketahui pesan $M' = M[1]'\|M[2]'\|M[3]\|\cdots\|M[l]$ dan $M = M[1]\|M[2]\|M[3]\|\cdots\|M[l]$. Apabila blok pesan ketiga hingga blok pesan terakhir pada kedua pesan diabaikan karena bernilai sama maka diperoleh hasil pesan $M' = M[1]'\|M[2]'$ dan $M = M[1]||M[2]$. Hasil dari bagian fungsi `Hash` dengan input pesan M' yaitu x' dan y'. Sementara itu, dengan input pesan M akan diperoleh hasil bagian fungsi `Hash` berupa nilai x dan y. x' dan x adalah hasil XOR yang memiliki 2^n kemungkinan hasil sehingga nilai x' memiliki dua kemungkinan hasil perbandingan dengan x yaitu bernilai sama atau berbeda. Dengan cara yang sama, akan diperoleh dua hasil perbandingan nilai y' dengan nilai y yaitu bernilai sama atau berbeda. Berdasarkan data tersebut, maka terdapat empat buah kasus kombinasi perbandingan nilai x' dengan x dan nilai y' dengan y'. Keempat kasus tersebut diantaranya:

a. Kasus pertama, nilai $x' = x$ dan $y' = y$.
b. Kasus kedua, nilai $x' = x$ dan $y' \neq y$.
c. Kasus ketiga, nilai $x' \neq x$ dan $y' \neq y$.
d. Kasus keempat, nilai $x' \neq x$ dan $y' = y$.

Merujuk pada fungsi finalisasi yang diilustrasikan pada Gambar 1 dan Gambar 2 diperoleh beberapa data mengenai perolehan pesan palsu dengan menggunakan keempat kasus input tersebut. Apabila menggunakan input fungsi finalisasi yang tergolong ke dalam kasus pertama pada skema `LightMAC_Plus` dan `LightMAC_Plus2` maka dapat dipastikan bahwa pesan M' adalah pesan palsu dari M karena menghasilkan nilai MAC yang sama. Sementara itu, input fungsi finalisasi yang tergolong ke dalam kasus kedua dan keempat memberikan hasil yang berbeda pada skema `LightMAC_Plus` dan `LightMAC_Plus2`. Pada skema `LightMAC_Plus`, input yang tergolong ke dalam kasus kedua dan keempat memberikan hasil bahwa pesan M' tidak dapat direkonstruksi menjadi pesan palsu dari M karena memiliki nilai MAC yang berbeda. Sementara itu, pada skema `LightMAC_Plus2` memberikan dua kemungkinan hasil yaitu M' berkemungkinan dapat direkonstruksi sebagai pesan palsu atau bukan sebagai pesan palsu dari M. Kemudian, sama seperti halnya pada input yang tergolong ke dalam kasus pertama yang memberikan kesamaan hasil perolehan pesan palsu pada kedua skema `LightMAC_Plus` dan `LightMAC_Plus2`, kesamaan hasil perolehan pesan palsu juga terjadi pada input yang tergolong ke dalam kasus keempat. Input ini memberikan dua kemungkinan hasil pada kedua skema yaitu M' berkemungkinan dapat direkonstruksi sebagai pesan palsu atau bukan sebagai pesan palsu dari M.

Strategi serangan *universal forgery* adalah mencari pesan modifikasi pada blok pesan pertama dan kedua yang kemudian disusun menjadi pesan M'. Pesan M' terdiri dari lima blok pesan dengan blok pesan pertama dan kedua merupakan pesan modifikasi serta blok pesan ketiga, keempat, dan kelima bernilai sama seperti pada pesan asli M. Liu dan Liu menjelaskan bahwa nilai MAC pesan M' akan bernilai sama dengan nilai MAC pesan asli M dengan peluang 1 yang menunjukkan bahwa M' adalah pesan palsu yang valid dari pesan asli $M[4]$. Berdasarkan strategi tersebut, diharapkan nilai M' yang diperoleh dari proses serangan *universal forgery* pada skema fungsi *hash* `LightMAC_Plus` dan `LightMAC_Plus2` tergolong pada kasus pertama untuk masing-masing skema fungsi *hash* sehingga nilai T' akan sama dengan nilai T dengan peluang 1. Namun, semua nilai M' pada skema `LightMAC_Plus` dan `LightMAC_Plus2` tidak ada yang memenuhi kasus pertama melainkan tergolong pada kasus ketiga. Hal ini dijabarkan lebih detail pada **Proposisi 1**.

Proposisi 1. Misalkan skema `LightMAC_Plus` dan `LightMAC_Plus2` menghasilkan nilai MAC berukuran n bit, $l \geq 2$, dan $E_B: \{0,1\}^n \rightarrow \{0,1\}^n$ merupakan *block cipher* dengan kunci rahasia B. Operasi perkalian pada skema `LightMAC_Plus` dan `LightMAC_Plus2` berjalan pada $GF(2^n)$. Diketahui bahwa input pesan $M' = M[1]'\|M[2]'\|M[3]\|\cdots\|M[l]$ berbeda dengan pesan

asli $M = M[1]\|M[2]\|M[3]\|\cdots\|M[l]$. Apabila ditemukan input pesan M' yang memenuhi $E_B(M[1]') \oplus E_B(M[2]') = E_B(M[1]) \oplus E_B(M[2])$ maka akan diperoleh nilai $[2^{l-1} \otimes E_B(M[1]')] \oplus [2^{l-2} \otimes E_B(M[2]')]$ yang bernilai sama dengan $[2^{l-1} \otimes E_B(M[1])] \oplus [2^{l-2} \otimes E_B(M[2])]$ dengan peluang sebesar $\frac{1}{2^n}$.

Bukti:

Diasumsikan telah diperoleh nilai $M' = M[1]'\|M[2]'\|M[3]\|\cdots\|M[l]$ yang memenuhi nilai $E_B(M[1]') \oplus E_B(M[2]') = E_B(M[1]) \oplus E_B(M[2])$. Berdasarkan algoritme fungsi `Hash` skema `LightMAC_Plus` dan `LightMAC_Plus2` maka hasil enkripsi blok pesan pertama dikali dengan 2^{l-1} dan blok pesan kedua dikali dengan 2^{l-2} sehingga pada pesan M' diperoleh *output* bagian fungsi `Hash` dengan nilai $[2^{l-1} \otimes E_B(M[1]')] \oplus [2^{l-2} \otimes E_B(M[2]')]$. Sementara itu, pada pesan asli M diperoleh *output* bagian fungsi `Hash` dengan nilai $[2^{l-1} \otimes E_B(M[1])] \oplus [2^{l-2} \otimes E_B(M[2])]$. Dari hasil operasi perkalian tersebut diperoleh data berupa $2^{l-1} \otimes E_B(M[1]') \neq 2^{l-1} \otimes E_B(M[1])$ dan $2^{l-2} \otimes E_B(M[2]') \neq 2^{l-2} \otimes E_B(M[2])$. Nilai-nilai tersebut berbeda dikarenakan $E_B(M[1]) \neq E_B(M[1]')$ dan $E_B(M[2]) \neq E_B(M[2]')$ sehingga hasil perkaliannya pun berbeda. Berdasarkan data tersebut, perbandingan nilai $[2^{l-1} \otimes E_B(M[1]')] \oplus [2^{l-2} \otimes E_B(M[2]')]$ dengan $[2^{l-1} \otimes E_B(M[1])] \oplus [2^{l-2} \otimes E_B(M[2])]$ memiliki dua kemungkinan yaitu bernilai sama atau berbeda.

Terdapat 2^n kemungkinan hasil XOR $[2^{l-1} \otimes E_B(M[1]')] \oplus [2^{l-2} \otimes E_B(M[2]')]$. Oleh karena itu, peluang nilai $[2^{l-1} \otimes E_B(M[1]')] \oplus [2^{l-2} \otimes E_B(M[2]')]$ sama dengan $[2^{l-1} \otimes E_B(M[1])] \oplus [2^{l-2} \otimes E_B(M[2])]$ sangat kecil yaitu $\frac{1}{2^n}$. Sebaliknya, peluang nilai $[2^{l-1} \otimes E_B(M[1]')] \oplus [2^{l-2} \otimes E_B(M[2]')]$ berbeda dengan $[2^{l-1} \otimes E_B(M[1])] \oplus [2^{l-2} \otimes E_B(M[2])]$ sangat besar yaitu $1 - \frac{1}{2^n}$. Adanya proses perkalian pada skema `LightMAC_Plus` dan `LightMAC_Plus2` menyebabkan sulitnya memperoleh pesan M' yang memenuhi kasus pertama sebagaimana dijabarkan pada **Lemma 1**. Hal ini dikarenakan peluang diperolehnya nilai $[2^{l-1} \otimes E_B(M[1]')] \oplus [2^{l-2} \otimes E_B(M[2]')]$ sama dengan $[2^{l-1} \otimes E_B(M[1])] \oplus [2^{l-2} \otimes E_B(M[2])]$ sangat kecil sehingga pesan M' yang diperoleh akan tergolong pada kasus kedua, ketiga, atau keempat.

5 KESIMPULAN

Serangan *universal forgery* pada skema F_t, `LightMAC_Plus`, dan `LightMAC_Plus2` dilakukan sebanyak 50 kali dengan menggunakan sepuluh sampel pesan acak dan lima himpunan kunci; dan parameter pesan berukuran 120 bit, panjang *counter* 8 bit, dan nilai $t = \{2,3\}$. Hasil simulasi menunjukkan bahwa hanya ditemukan *collision* tetapi tidak ditemukan pesan palsu untuk setiap serangan yang dilakukan. Hal ini diperkuat dengan pembuktian matematis yang menunjukkan bahwa peluang nilai MAC pesan palsu bernilai sama dengan nilai MAC pesan asli sangat kecil yaitu $\frac{1}{2^{32}}$. Berdasarkan data-data tersebut, dapat disimpulkan bahwa skema fungsi *hash* F_t, `LightMAC_Plus`, dan `LightMAC_Plus2` tahan terhadap serangan *universal forgery*.

REFERENSI

[1] W. Stallings and M. J. Horton, Cryptography and Network Security : Principles and Practice, 7th ed., England: Pearson Education Limited, 2017.

[2] A. J. Menezes, P. C. V. Oorschot and S. A. Vanstone, Handbook of Applied Cryptography, USA: CRC Press, 1996.

[3] T. Peyrin and L. Wang, "Generic Universal Forgery Attack on Iterative Hash-Based MACs," *Advances in Cryptology – EUROCRYPT 2014*, vol. 8441, pp. 147-164, 2014.

[4] F. Liu and F. Liu, "Universal Forgery with Birthday Paradox: Application to Blockcipher-based Message Authentication Codes and Authenticated Encryptions," *IACR Cryptology ePrint Archive*, pp. 1-24, 2017.

[5] Y. Naito, "Blockcipher-Based MACs: Beyond the Birthday Bound Without Message Length," *Advances in Cryptology – ASIACRYPT 2017*, Vols. Lecture Notes in Computer Science, vol 10626, pp. 446-470, 2017.

[6] S. M. AlDabbagh, "Design 32 bit Lightweight Block Cipher Algorithm (DLBCA)," *International Journal of Computer Applications*, vol. 166, no. 8, pp. 17-20, 2017.

[7] S. Gueron and Y. Lindell, "GCM-SIV: Full Nonce Misuse-Resistant Authenticated Encryption at Under One Cycle per Byte," *ACM CCS*, pp. 109-119, 2015.

[8] T. Iwata and K. Minematsu, "Stronger Security Variants of GCM-SIV," *IACR Transactions on Symmetric Cryptology*, vol. 1, pp. 134-157, 2016.