KRIPTOGRAFI RINGAN DI INTERNET OF THINGS: TINJAUAN LITERATUR SISTEMATIS

Rudhi Wahyudi Febrianto[1], Arief Zulianto[2]
Prodi Magister Teknik Informatika, Pascasarjana, Universitas Langlangbuana [1,2]
rudhiwahyudi@gmail.com[1]
madzul@gmail.com[2]

***Abstrak*— *Internet of Things* (IoT) memungkinkan suatu objek menghasilkan data dan bertukar data. Pengaplikasian IoT menggunakan mikrokontroler seperti Arduino belum memiliki fitur untuk menjaga keamanan data di dalamnya. Selain itu, Arduino memiliki kapabilitas komputasi terbatas. Oleh karena itu, perlu diterapkan kriptografi dengan algoritme yang memiliki komputasi rendah pada Arduino untuk menjaga keamanan data. Kriptografi perlu diterapkan pada pengaplikasian IoT menggunakan mikrokontroler untuk menjaga keamanan proses transaksi data dan menjaga keaslian asal suatu data. Penerapan kriptografi pada mikrokontroler juga harus ringan dan dapat berjalan pada mikrokontroler terutama mikrokontroler Arduino. Pada Arduino, kemampuan komputasi bersifat terbatas namun kebanyakan algoritme komputasi keamanan yang ada memiliki komputasi yang tinggi. Oleh karena itu, protokol kriptografi yang diterapkan harus memiliki algoritme yang efisien dan kemampuan komputasi yang rendah. Tujuan penelitian kami adalah untuk memahami apakah pendekatan kriptografi ringan dapat digunakan untuk menumbuhkan IoT yang aman berdasarkan desain. Sebagai langkah pertama dalam proses penelitian kami, kami melakukan tinjauan literatur sistematis pada kriptografi ringan untuk mengumpulkan pengetahuan tentang penggunaan teknologi ini dan untuk mendokumentasikan tingkat saat ini. Kami menemukan 5 kasus penggunaan kriptografi ringan dalam literatur. Kami juga menemukan beberapa masalah dalam Waktu eksekusi algoritme pada Arduino Uno memiliki waktu yang lebih besar dua kali lipat dibandingkan dengan waktu eksekusi algoritme pada PC, Perubahan nilai pada panjang data dan pasangan tanda tangan digital berpengaruh terhadap hasil verifikasi keabsahan tanda tangan digital. Kami mendokumentasikan dan mengkategorikan penggunaan saat ini kriptografi ringan, dan memberikan beberapa rekomendasi untuk masa depan bekerja untuk mengatasi masalah yang disebutkan di atas.**

I. PENDAHULUAN

Internet of things (IoT) tengah menjadi perbincangan dalam dunia teknologi pada saat ini. Istilah IoT umumnya mengacu pada sebuah skenario suatu jaringan internet, kemampuan konektivitas dan komputasi berada dalam sebuah objek yang memungkinkan objek tersebut untuk menghasilkan data, bertukar data, dan mengambil data dengan sedikit campur tangan manusia (Rose et al. 2015) [1]. Pada dasarnya IoT merupakan konstruksi yang saling menghubungkan perangkat umum satu sama lain. Perangkat umum dapat berupa jam tangan, televisi, termostat, mobil, dan lampu. Selain perangkat umum yang disebutkan sebelumnya, pengaplikasian IoT juga biasa digunakan pada mikrokontroler. Salah satu contoh mikrokontroler yang umum digunkan dalam pengaplikasian IoT adalah Arduino. Pengaplikasian IoT pada mikrokontroler ini dapat digunakan sebagai sarana pertukaran data atau pengiriman data. Salah satu contoh pengaplikasiannya adalah sistem location based perangkat berdaya komputasi rendah dengan Arduino. Sistem ini bekerja dengan mengirimkan data berupa longitude, latitude, dan internet *protocol* (IP) *address* dari perangkat mikrokontroler ke suatu *server*. Pengaplikasian dengan sistem tersebut perlu memiliki fitur keamanan di dalamnya. Keamanan yang dimaksud dapat berupa keamanan saat melakukan pengiriman data antar alat elektronik. Jika pada *client server* keamanan proses pengiriman data dilindungi dengan *hypertext transfer protocol secure* (HTTPS), sebaliknya pada IoT belum terdapat keamanan saat proses transaksi data. Keamanan proses transaksi data pada IoT merupakan hal yang penting, bukan hanya pada saat data dikirimkan tetapi juga bagaimana data tidak diubah oleh seseorang atau pihak ketiga sehingga data tersebut bersifat asli dan untuk mengetahui keaslian asal data tersebut. Kriptografi perlu diterapkan pada pengaplikasian IoT menggunakan mikrokontroler untuk menjaga keamanan proses transaksi data dan menjaga keaslian asal suatu data. Penerapan kriptografi pada mikrokontroler juga harus ringan dan dapat berjalan pada mikrokontroler terutama mikrokontroler Arduino. Pada Arduino, kemampuan komputasi bersifat terbatas namun kebanyakan algoritme komputasi keamanan yang ada memiliki komputasi yang tinggi.

Terdapat berbagai penelitian yang telah dilakukan mengenai kriptografi ringan ini, salah satunya penelitian yang dilakukan oleh William J. Buchanan, Shancang Li dan Rameez Asif (2017) dengan judul *Lightweight cryptography methods* [2], penelitian tersebut berfokus untuk metode kriptografi konvensional, seperti untuk AES (enkripsi), SHA-256 (*hashing*) dan RSA/*Elliptic Curve* (penandatanganan), bekerja dengan baik pada sistem yang memiliki daya pemrosesan dan kemampuan memori yang wajar/masuk akal. Lalu penelitian yang dilakukan oleh Selfi Qisthina dan Shelvie Nidya Neyman (2021) dengan judul Pengamanan *Internet of Things* untuk Tanda Tangan Digital Menggunakan Algoritme Elgamal *Signature Scheme* [3], penelitian tersebut berfokus Internet of Things (IoT) memungkinkan suatu objek menghasilkan data dan bertukar data. Pengaplikasian IoT menggunakan mikrokontroler seperti

Arduino belum memiliki fitur untuk menjaga keamanan data di dalamnya. Selain itu, Arduino memiliki kapabilitas komputasi terbatas. Oleh karena itu, perlu diterapkan kriptografi dengan algoritme yang memiliki komputasi rendah pada Arduino untuk menjaga keamanan data. Penjagaan keamanan data terutama pada keaslian asal data, dilakukan dengan menggunakan tanda tangan digital. Penerapan tanda tangan digital dapat dilakukan salah satunya dengan algoritme Elgamal signature scheme. Lalu penelitian yang dilakukan oleh Helmy Rafi Nawawi , Ari Kusyanti dan Rakhmadhany Primananda (2019) dengan judul Implementasi Algoritme AEGIS untuk *Payload* Data Protokol MQTT dan CoAP pada Raspberry Pi 3 [4], penelitian tersebut berfokus Raspberry Pi 3 adalah perangkat yang dapat digunakan sebagai *middleware*. *Middleware* bertanggung jawab dalam memfasilitasi komunikasi dan manajemen informasi dari komponen heterogen. Dalam rancangan sistem middleware penelitian sebelumnya masih terdapat celah keamanan sistem pada pengiriman data yang membuat paket data yang dikirimkan dapat diketahui oleh pihak yang tidak bertanggung jawab. Suatu metode yang dapat menjamin *confidentiality* dan authentication yaitu algoritme AEGIS diajukan sebagai solusi untuk mengatasi permasalahan tersebut. Lalu penelitian yang dilakukan oleh Hala Tawalbeh , Sonia Hashish , Loai Tawalbeh dan Anwar Aldairi (2017) dengan judul *Security in Wireless Sensor Networks Using Lightweight Cryptography* [5], penelitian tersebut berfokus untuk *Wireless Sensor Networks* (WSNs) memiliki banyak penggunaan dan aplikasi di tingkat individu dan organisasi. Sensor terintegrasi di banyak bidang termasuk kesehatan, lalu lintas, pertanian, industri, dan lain-lain. Kegunaan sensor adalah terkait dengan penggunaan banyak teknologi termasuk nirkabel protokol komunikasi, Internet of Things, *Cloud Computing*, komputasi *mobile*, dan teknologi berkembang lainnya. Ini interaksi yang melibatkan transmisi informasi penting dalam banyak kasus memaksakan kebutuhan untuk mengamankan data ini dari semua kemungkinan serangan. Lalu penelitian yang dilakukan oleh Yohanes Heryka Febriarso , Ari Kusyanti dan Primantara Hari Trisnawan (2021), dengan judul Implementasi Algoritme SPECK dalam Sistem *Monitoring* Sumber Daya pada Raspberry Pi [6], penelitian tersebut berfokus Sistem monitoring merupakan sesuatu yang awam dilakukan saat ini. Namun perlu diperhatikan bahwa sistem keamanan juga diperlukan. Bila orang asing menguasai data sistem monitoring, maka dapat membahayakan pemilik sistem monitoring. Aspek yang perlu dijaga sistem monitoring adalah *confidentiality*. Algoritme SPECK dapat memenuhi keamanan *confidentiality* dan memiliki keunggulan pada sistem berspesifikasi rendah. Algoritme SPECK diimplementasikan pada sistem monitoring pada manager dan agent saat melakukan pengiriman dan penerimaan data.

Walaupun pada penelitian tersebut telah dilakukan kriptografi ringan di IoI, namun metode kriptografi konvensional tidak baik untuk berskala dunia dengan sistem tertanam pada jaringan sensor. Dengan demikian, metode kriptografi ringan diusulkan untuk mengatasi banyak masalah kriptografi konvensional ini termasuk kendala yang berkaitan dengan ukuran fisik, persyaratan pemrosesan, keterbatasan memori dan yang menguras energi. Lalu Waktu eksekusi proses tanda tangan digital membutuhkan waktu lebih lama dibandingkan dengan waktu eksekusi proses verifikasi. Algoritme Elgamal signature scheme membutuhkan waktu dua kali lebih lama karena banyaknya perhitungan sistematis pada perangkat Arduino Uno. Lalu terdapat *delay* dalam pembacaan arduino dengan algoritme AEGIS. keamanan dunia maya dan serangan untuk *Wireless Sensor Networks* (WSNs) harus diperhatikan. Lalu hasil pengujian serangan aktif dengan metode known plaintext attack mengunakan algoritme SPECK mampu untuk dapat bertahan dari serangan tersebut selama 24 jam.

II. DESAIN STUDI

A. Konteks

Ilmu dan seni untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain disebut kriptografi (Ariyus, 2008) [7].

1. Dalam kriptografi memiliki tiga tujuan utama yaitu: 1. Confidentiality: kriptografi kerahasiaan data yang bertujuan untuk memastikan informasi hanya pihak-pihak tertentu yang berhak mengakses pesan.
2. Integrity: kriptografi keutuhan data yang bertujuan pesan tidak ada perubahan dalam proses pengiriman atau penerimaan.
3. Authentication: kriptografi keaslian data yang bertujuan memastikan penerima bahwa pesan yang diterima merupakan pesan asli dari sumber yang dapat dipercaya.

Dengan 3 aspek utama tersebut kriptografi dibagi menjadi 2 pengelompokan menurut kunci yang digunakan yaitu algoritme simetris dan asimetris (Ariyus, 2008) [7]. Kriptografi dengan kunci simetris yang memiliki kunci sama pada proses enkripsi dan deskripsinya. Sedangkan kriptografi dengan kunci asimetris yang proses enkripsi dan deskripsi menggunakan kunci yang berbeda (Ariyus, 2008) [7].

B. Tujuan dan Pertanyaan Penelitian

Tujuan dari penelitian kami adalah untuk memahami apakah kriptografi ringan secara umum, dapat bekerja di IoT. langkah pertama dari penelitian ini, kami mengumpulkan kasus kriptografi ringan di IoT untuk mengumpulkan bukti dari literatur tentang tingkat kemampuan pembacaan IoT terhadap algoritme.

Untuk mencapai tujuan itu, kami merumuskan masalah-masalah, sebagai berikut:

Pertanyaan 1 atau P1) Apakah saja persyaratan pemrosesan, keterbatasan memori yang dapat menguras energi pada krptografi ringan di IoT?

Pertanyaan 2 atau P2) Kenapa Waktu eksekusi proses tanda tangan digital membutuhkan waktu lebih lama dibandingkan dengan waktu eksekusi proses

verifikasi untuk Tanda Tangan Digital Menggunakan Algoritme Elgamal Signature Scheme?

Pertanyaan 3 atau P3) Bagaimana *delay* dalam pembacaan arduino dengan algoritme AEGIS?

Pertanyaan 4 atau P4) Bagaimana keamanan dunia maya dan serangan untuk *Wireless Sensor Networks* (WSNs) yang harus diperhatikan?

Pertanyaan 5 atau P5) Bagaimana hasil pengujian serangan aktif dengan metode known plaintext attack mengunakan algoritme SPECK mampu untuk dapat bertahan dari serangan tersebut selama 24 jam?

P1, P2 dan P3 bertujuan untuk menemukan dalam literatur, penggunaan komponen apa saja yang ada pada kriptografi ringan dan IoT yang menjadi penyebab masalah.

P4 dan P5 bertujuan untuk menemukan dalam literatur, keamanan apa saja yang ada dan rentan akan serangan.

C. Proses pencarian

Untuk melakukan penelitian, Kami mengumpulkan 5 penelitian, untuk memutuskan mana dari mereka yang dianalisis secara mendalam. Akhirnya, kami mendapat hasil, dari mana kami mengekstrak informasi yang diperlukan untuk menjawab pertanyaan kami

III. HASIL

Pada bagian ini kami melaporkan hasil yang diambil dari makalah yang dianalisis, disusun berdasarkan pertanyaan penelitian. Diskusi tentang hasil akan mengikuti di Bagian IV.

A. Pertanyaan 1

P1 : Apakah saja persyaratan pemrosesan, keterbatasan memori yang dapat menguras energi pada krptografi ringan di IoT?

Dalam kasus implementasi perangkat lunak, implementasi ukuran dan konsumsi RAM melalui penempatan (*byte* per siklus) adalah metrik yang lebih disukai untuk aplikasi ringan. Semakin kecil semakin baik [8]. Dalam kasus perangkat lunak, *framework Fair Evaluation of Lightweight Cryptographic Systems* (FELICS) diusulkan untuk mengevaluasi kinerja blok ringan atau kinerja *stream cipher* dalam ukuran implementasi, konsumsi RAM dan waktu yang dibutuhkan untuk melakukan operasi tertentu [9].

Tabel 1 *FELICS results for lightweight ciphers*
(Sumber: William J. Buchanan, Shancang Li dan Rameez Asif. Dalam *Journal of Cyber Security Technology*)

General info			AVR (8-bit)			MSP (16-bit)			ARM (32-bit)		
Name	Block	Key	Code	RAM	Time	Code	RAM	Time	Code	RAM	Time
Chaskey	128	128	770	84	1597	490	86	1351	178	80	614
SPECK	64	96	448	53	2829	328	48	1959	256	56	1003
SPECK	64	128	452	53	2917	332	48	2013	276	60	972
Chaskey-LTS	128	128	770	84	413	492	86	2064	178	80	790
SIMON	64	96	600	57	4269	460	56	2905	416	64	1335
SIMON	64	128	608	57	4445	468	56	3015	388	64	1453
LEA	128	128	906	80	4023	722	78	2814	520	112	1171
RECTANGLE	64	128	602	56	4381	480	54	2651	452	76	2432
RECTANGLE	64	80	606	56	4433	480	54	2651	452	76	2338
SPARX	64	128	662	51	4397	580	52	2261	654	72	2338
SPARX	128	128	1184	74	5478	1036	72	3057	1468	104	2935
RC5-20	64	128	1068	63	8812	532	60	15,925	372	64	1919
AES	128	128	1246	81	3408	1170	80	4497	1348	124	4044
HIGHT	64	128	636	56	6231	636	52	7117	670	100	5532
Fantomas	128	128	1712	76	9689	1920	78	3602	2184	184	4550
Robin	128	128	2530	108	7813	1942	80	4913	2188	184	6250

Tabel 1 menunjukkan hasil FELICS dari algoritma cipher ringan populer untuk tiga implementasi yang berbeda: AVR 8-bit, MSP 6- bit dan ARM 32-bit.

Salah satu yang menunjukkan untuk pengganti AES untuk kriptografi ringan adalah PRESENT. Ini menggunakan ukuran blok yang lebih kecil dan potensi kunci yang lebih kecil (seperti kunci 80-bit). PRESENT pengguna baik 80-bit (10 karakter hex) atau kunci enkripsi 128-bit (16 karakter hex). Ini beroperasi pada blok 64-bit dan menggunakan substitusi-permutasi.

Table 2. Lightweight hash functions
(Sumber: William J. Buchanan, Shancang Li dan Rameez Asif. Dalam *Journal of Cyber Security Technology*)

Lightweight hash function	Digest [bits]	Code [bytes]	RAM [bytes]	RAM [bytes]	RAM stack	Cycle (8-byte msg)	Cycle (50-byte msg)	Cycle (100-byte msg)	Cycle (500-byte msg)
SPONGENT-256/256/128	256	364	16	96	5	1 542 923	3 856 916	6 170 900	25 454 100
SPONGENT-160/160/80	160	598	10	60	6	795 294	2 783 241	4 771 186	20 674 746
S-Quark	256	1106	4	60	5	708 783	1 417 611	2 339 023	9 4270 23
D-Quark	176	974	2	42	5	631 871	1 516 685	2 570 035	10 996 835
Keccak[r = 40, c = 160]	160	752	5	45	3	58 063	162 347	278 269	1 205 627
Keccak[r = 144, c = 256]	256	608	18	92	4	90 824	181 466	317 221	1 313 291
PHOTON-160/36/36	160	764	9	39	11	620 921	1 655 364	2 793 265	11 999 914
PHOTON-256/32/32	256	1244	4	68	10	254 871	486 629	787 896	3 105 396

jaringan (SPN). Dengan SPN, seperti halnya AES (Rijndael), kami beroperasi di blok plaintext dan menerapkan kunci dan kemudian menggunakan sejumlah putaran yang kami gunakan substitusi kotak (S-box) dan kotak permutasi (P-box). Operasi yang digunakan biasanya dicapai melalui XOR/rotasi bitwise, dan bagian dari kunci diperkenalkan melalui putaran operasi. Proses dekripsi kemudian kebalikan dari enkripsi putaran, dan S-box/P-box dibalik dalam operasinya.

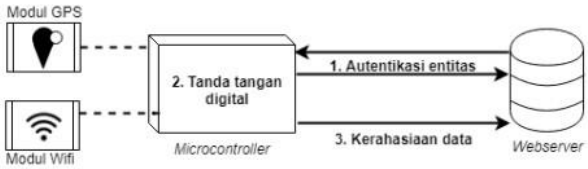
CLEFIA adalah *cipher* blok ringan yang dipelajari dengan baik dan ditulis oleh Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai dan Tetsu Iwata dan dapat diimplementasikan dengan gerbang 6K. Itu didefinisikan oleh Sony dan memiliki kunci 128, 192 dan 256 bit dan ukuran blok 128-bit. Bersamaan dengan ini, termasuk dalam Standar Internasional ISO/IEC 29192 untuk metode cipher blok ringan (ISO/IEC 29192-2:2012). RC5 ('Ron's Cipher 5'), dibuat pada tahun 1994 oleh Ron L. Rivest, juga menunjukkan potensi besar untuk metode kriptografi ringan. Ini adalah cipher blok

yang memiliki ukuran blok variabel (32, 64 atau 128 bit), jumlah putaran variabel dan ukuran kunci variabel (0 hingga 2048 bit). Dengan demikian dapat digunakan untuk mencocokkan enkripsi dengan kemampuan perangkat. Jika itu adalah perangkat berdaya rendah dengan memori terbatas dan jejak fisik yang relatif kecil, kita dapat menggunakan ukuran blok 32-bit dan kunci 80-bit, hanya dengan beberapa putaran. Tetapi kami dapat meningkatkan keamanan jika perangkat dapat mengatasinya dan menggunakan ukuran blok 128-bit dan kunci 128-bit. Itu juga bisa fleksibel, di mana satu perubahan di kedua sisi dapat meningkatkan atau mengurangi persyaratan.

B. Pertanyaan 2

P2 : Kenapa Waktu eksekusi proses tanda tangan digital membutuhkan waktu lebih lama dibandingkan dengan waktu eksekusi proses verifikasi untuk Tanda Tangan Digital Menggunakan Algoritme Elgamal *Signature Scheme*?

Sistem dirancang untuk menyediakan tiga layanan keamanan. Tiga layanan keamanan tersebut adalah autentikasi entitas, tanda tangan digital, dan kerahasiaan data. Lingkungan simulasi pada penelitian ini dibangun pada sistem keamanan IoT yang memiliki tiga tahapan utama seperti pada Gambar 2. Tahapan pertama yang dilakukan adalah Arduino melakukan inisialisasi ke server untuk pertukaran kunci atau autentikasi entitas. Tahapan kedua, yaitu pembuatan tanda tangan digital pada data lokasi mikrokontroler untuk autentikasi asal data. Tahapan ketiga, yaitu Arduino melakukan enkripsi pada data sebelum dikirimkan ke server untuk menjaga kerahasian data.



Gambar 1 Lingkungan sistem simulasi keseluruhan
(Sumber: Selfi Qisthina dan Shelvie Nidya Neyman. Dalam Jurnal Ilmu Komputer dan Agri-Informatika, Volume 8 Nomor 1 halaman 69 - 78)

Proses pembangkitan kunci dilakukan oleh pengirim. Kunci yang dibangkitkan berguna untuk memberikan tanda tangan digital pada data. Kunci yang dibangkitan terdapat dua jenis, yaitu kunci private dan kunci public. Berikut pada Tabel 1 adalah hasil dari proses pembangkitan kunci.

Tabel 3. Hasil pembangkitan kunci
(Sumber: Selfi Qisthina dan Shelvie Nidya Neyman. Dalam Jurnal Ilmu Komputer dan Agri-Informatika, Volume 8 Nomor 1 halaman 69 – 78)

Parameter	Nilai
p	61091
g	60383
k	60719
Kunci *private*	60647
Kunci *public*	35865

Tabel 4. Hasil pembangkitan tanda tangan digital
(Sumber: Selfi Qisthina dan Shelvie Nidya Neyman. Dalam Jurnal Ilmu Komputer dan Agri-Informatika, Volume 8 Nomor 1 halaman 69 - 78)

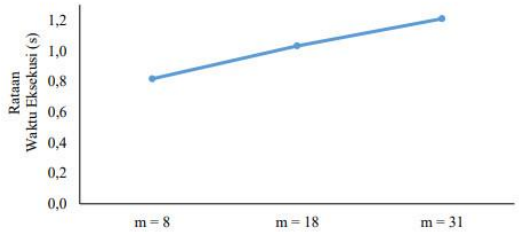
Parameter	Nilai
Data (m)	2147483647
r	32914
s	25611

Verifikasi tanda tangan digital menghasilkan hasil perbandingan dari dua proses komputasi. Berikut pada Tabel 5 adalah hasil verifikasi keabsahan tanda tangan digital. Berdasarkan pada Tabel 5, perbandingan dari hasil dua proses tersebut memiliki nilai yang sama yang artinya verifikasi keabsahan tanda tangan digital berhasil dilakukan.

Tabel 5. Hasil verifikasi tanda tangan digital
(Sumber: Selfi Qisthina dan Shelvie Nidya Neyman. Dalam Jurnal Ilmu Komputer dan Agri-Informatika, Volume 8 Nomor 1 halaman 69 - 78)

Proses komputasi	Hasil
$g^m \bmod p$	16717
$kunci\ public^r\ r^s \bmod p$	16717

Hasil analisis kinerja yang dilakukan pada Arduino Uno dan pada PC. Analisis kinerja yang dilakukan berupa analisis kinerja terhadap waktu eksekusi. Pengukuran kinerja algoritme diukur bedasarkan panjang bilangan acak prima (p) yang digunakan, panjang data (m) yang digunakan, dan lama waktu eksekusi algoritme.



Gambar 2 Panjang Message (m)
(Sumber: Selfi Qisthina dan Shelvie Nidya Neyman. Dalam Jurnal Ilmu Komputer dan Agri-Informatika, Volume 8 Nomor 1 halaman 69 - 78)

Analisis keamanan algoritme dilihat dari hasil verifikasi *signature* saat ada perubahan pada m dan pada pasangan tanda tangan digital (r,s). Pada Tabel 4 dan 5, perubahan pada m ataupun data pasangan tanda tangan digital (r,s) menyebabkan perbedaan nilai pada proses komputasi 1 dan proses komputasi 2. Perbedaan nilai yang dihasilkan menujukkan bahwa proses verifikasi telah gagal. Panjang kunci private dihasilkan pada penelitian ini sebesar 18 bit. Jika dilakukan metode brute force, maka akan menghasilkan 2 18 = 262 144 kemungkinan bilangan untuk menebak nilai dari kunci private yang dihasilkan. Jika brute force dilakukan pada PC penelitian ini membutuhkan waktu sekitar 262 detik atau sekitar empat menit. Idealnya untuk panjang kunci asimetris, minimal 2048 bit seperti pada RSA.

Tabel 6. Data awal dan data sesudah diubah
(Sumber: Selfi Qisthina dan Shelvie Nidya Neyman. Dalam Jurnal Ilmu Komputer dan Agri-Informatika, Volume 8 Nomor 1 halaman 69 - 78)

Parameter	Nilai		
	Data awal	Data pesan diubah	Data (*r,s*) diubah
Pesan (*m*)	2011684	2011876	2011684
r	20965	20965	20432
s	481	481	555

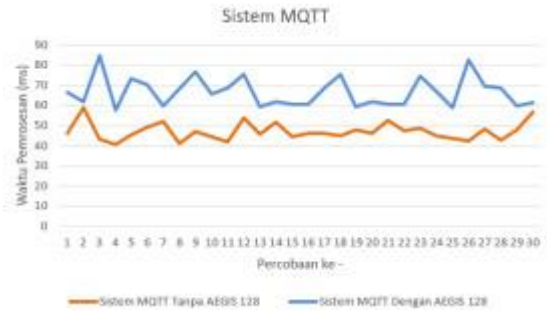
Tabel 7. Hasil verifikasi dengan data awal dan data setelah diubah
(Sumber: Selfi Qisthina dan Shelvie Nidya Neyman. Dalam Jurnal Ilmu Komputer dan Agri-Informatika, Volume 8 Nomor 1 halaman 69 - 78)

Proses verifikasi	Hasil		
	Data awal	Data pesan diubah	Data (*r,s*) diubah
Proses komputasi 1	29154	27223	27223
Proses komputasi 2	29154	29154	39187

C. Pertanyaan 3

P3 : Bagaimana *delay* dalam pembacaan arduino dengan algoritme AEGIS?

Pengujian kinerja waktu sistem bertujuan untuk mengetahui perbedaan signifikan karena implementasi AEGIS 128. Pada sistem *middleware* yang menggunakan protokol MQTT atau menggunakan protokol CoAP akan diuji data sampel waktu pemrosesan saat AEGIS 128 diimplementasikan dan tanpa AEGIS 128. Dari hasil data sampel waktu pemrosesan pada masing-masing sistem akan dibandingkan. Grafik pada Gambar 3 menampilkan perbedaan waktu sistem dengan menggunakan protokol MQTT saat proses pengiriman dilakukan dari perangkat nodeMCU ke *client* tanpa AEGIS 128 dan pada sistem menggunakan AEGIS 128. Dari hasil rata-rata selisih sistem MQTT menggunakan algoritme AEGIS 128 dan tanpa algoritme AEGIS 128 adalah 19,592 milisecond.



Gambar 3. Grafik hasil perbandingan waktu proses sistem MQTT
(Sumber: Helmy Rafi Nawawi, Ari Kusyanti dan Rakhmadhany Primananda. Dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 3, No. 8, hlm. 7879-7887)

Gambar 4. Grafik hasil perbandingan waktu proses sistem CoAP
(Sumber: Helmy Rafi Nawawi, Ari Kusyanti dan Rakhmadhany Primananda. Dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 3, No. 8, hlm. 7879-7887)

Grafik pada Gambar 4 menampilkan perbedaan waktu sistem dengan menggunakan protokol CoAP saat proses pengiriman dilakukan dari perangkat nodeMCU ke *middleware* tanpa AEGIS 128 dan pada sistem menggunakan AEGIS 128. Hasil rata-rata selisih sistem CoAP menggunakan algoritme AEGIS 128 dan tanpa algoritme AEGIS 128 adalah 28,141 milisecond.

Pengujian *Quality of Service* (QoS) merupakan pengujian untuk mengukur kualitas layanan yang diperoleh dari suatu jaringan komunikasi. Dasar pengujian QoS yaitu menganalisis parameter dari *packet loss*, *delay* dan jitter. *Packet loss* merupakan indikasi bahwa terdapat paket yang hilang atau gagal terkirim ke tujuan. Nilai packet loss didapatkan dari nilai *expected* dan nilai actual. Nilai *expected* yaitu nilai harapan dari banyak paket yang terkirim ke tujuan, sedangkan nilai actual adalah nilai dari paket yang berhasil terkirim ke tujuan. *Delay* adalah waktu jeda paket terkirim dari sumber sampai dengan waktu pengiriman ACK dari tujuan. Jitter adalah variasi nilai dari *delay* pengiriman paket. Berdasarkan Tabel 1 hasil pengujian pengiriman data dengan protokol MQTT dari perangkat nodeMCU ESP8266 ke *middleware* tanpa mekanisme keamanan.

Menghasilkan *success rate* 100 % dan packet loss 0%. Dengan nilai rata-rata *delay* adalah 25,7 detik dan jitter adalah 1,41 detik.

Tabel 8. Hasil pengujian skenario MQTT
(Sumber: Helmy Rafi Nawawi , Ari Kusyanti dan Rakhmadhany Primananda. Dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 3, No. 8, hlm. 7879-7887)

Expected	*Actual*	*Success Rate*	*Packet Loss Rate*	*Delay*	*Jitter*
20	20	100 %	0 %	24,8	1,36
20	20	100 %	0 %	26,0	1,43
20	20	100 %	0 %	26,0	1,43
20	20	100 %	0 %	26,0	1,43
20	20	100 %	0 %	26,0	1,43
Rata-rata (detik)				25,7	1,41

Tabel 9. Hasil pengujian scenario CoAP
(Sumber: Helmy Rafi Nawawi, Ari Kusyanti dan Rakhmadhany Primananda. Dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 3, No. 8, hlm. 7879-7887)

20	20	100 %	0 %	14,3	0,0013
20	20	100 %	0 %	14,3	0,0013
20	20	100 %	0 %	14,3	0,0012
20	20	100 %	0 %	14,3	0,0012
20	20	100 %	0 %	14,3	0,0013
Rata-rata (detik)				14,3	0,0013

Berdasarkan Tabel 8 hasil pengujian pengiriman data dengan protokol CoAP dari sensor nodeMCU ESP8266 ke *middleware* tanpa mekanisme keamanan. Menghasilkan *success rate* 100 % dan packet loss 0 %. Dengan nilai rata-rata delay adalah 14,3 detik dan jitter adalah 0,0013 detik.

Berdasarkan Tabel 9 hasil pengujian pengiriman data pada protokol MQTT dari sensor nodeMCU ESP8266 ke middleware dengan algoritme AEGIS 128. Menghasilkan *success rate* 99 % dan *packet loss* 1%. Dengan nilai rata-rata delay adalah 26,3 detik dan jitter adalah 1,46 detik.

Tabel 10. Hasil pengujian skenario MQTT AEGIS 128
(Sumber: Helmy Rafi Nawawi, Ari Kusyanti dan Rakhmadhany Primananda. Dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 3, No. 8, hlm. 7879-7887)

Expected	*Actual*	*Success Rate*	*Packet Loss Rate*	*Delay*	*Jitter*
20	19	95 %	5 %	26,18	1,52
20	20	100 %	0 %	26,37	1,45
20	20	100 %	0 %	26,37	1,45
20	20	100 %	0 %	26,37	1,45
20	20	100 %	0 %	26,37	1,45
Rata-rata (detik)				26,3	1,46

Tabel 11. Hasil pengujian skenario MQTT AEGIS 128
(Sumber: Helmy Rafi Nawawi, Ari Kusyanti dan Rakhmadhany Primananda. Dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 3, No. 8, hlm. 7879-7887)

Expected	*Actual*	*Success Rate*	*Packet Loss Rate*	*Delay*	*Jitter*
20	19	95 %	5 %	14,6	0,0021
20	20	100 %	0 %	14,6	0,0018
20	20	100 %	0 %	14,6	0,0018
20	18	90 %	10 %	14,6	0,0023
20	16	80 %	20 %	14,4	0,0025
Rata-rata (detik)				14,5	0,0021

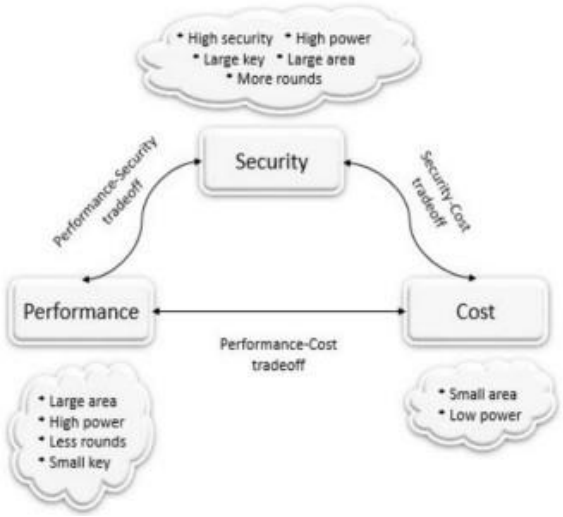
Berdasarkan Tabel 11 hasil pengujian pengiriman data dengan protokol CoAP dari sensor nodeMCU ESP8266 ke middleware dengan algoritme AEGIS 128. Menghasilkan *success rate* 93 % dan *packet loss* 7 %. Nilai rata-rata *delay* adalah 14,5 detik dan jitter adalah 0,0021 detik.

D. Pertanyaan 4

P4 : Bagaimana keamanan dunia maya dan serangan untuk *Wireless Sensor Networks* (WSNs) yang harus diperhatikan?

Kriptografi Ringan mengacu pada satu set algoritma kriptografi simetris dan asimetris yang dirancang untuk memastikan keamanan yang memadai dalam sistem yang memiliki kendala khusus untuk menjalankan lingkungan dan kemampuan daya seperti WNS. Ringan dalam kriptografi dapat dicapai pada tingkat perangkat keras dan perangkat lunak, di mana ukuran chip, energi di tingkat perangkat keras, ukuran kode, dan kompleksitas RAM di tingkat perangkat lunak digunakan untuk mengukur tingkat optimasi yang dipenuhi dengan menerapkan LWC. Pandangan kriptografer adalah bahwa

implementasi perangkat keras dan perangkat lunak dioptimalkan, dengan mengingat bahwa apa yang ringan untuk implementasi perangkat keras mungkin tidak ringan untuk implementasi perangkat lunak dan sebaliknya.



Gambar 5. Desain Algoritma Kriptografi Ringan
(Sumber: Hala Tawalbeh, Sonia Hashish, Loai Tawalbeh dan Anwar Aldairi. Dalam *Journal of Information Assurance and Security*, Volume 12, pp. 118-123)

1. Pendekatan Kriptografi Ringan Simetris untuk persyaratan minimal dan kinerja optimal, Sebagai berikut:
   a) *Advanced Encryption Standard* (AES): Ini adalah cipher blok standar NIST yang beroperasi pada blok 128-bit dan menggunakan ukuran kunci 256 bit, 192 bit, atau 128 bit. Enkripsi dapat dilakukan dengan sepuluh, 12, atau 14 putaran tergantung pada ukuran input. Setiap putaran AES melewati empat langkah dasar: sub byte, shift rows, MixedColumns, dan AddRoundKey. Array byte diperlakukan sebagai matriks 4 x 4. Implementasi AES 128-bit yang ringan membutuhkan 3100 GE (gate setara) dan menawarkan *throughput* 80 Kb/s pada daya 100 KHz. Namun, serangan Man-in-the-Middle (MITM) masih mengancam AES Ringan [10].
   b) DESLX: varian DES (Data Encryption Standard) ringan yang beroperasi pada blok 64-bit menggunakan kunci 184-bit. Enkripsi menggunakan DESLX terjadi dengan 16 putaran. Perubahan signifikan diterapkan pada DES standar untuk mencapai tampilan DESLX yang ringan, seperti hanya menggunakan satu S-box, bukan delapan Sbox dan menerapkan pendekatan pemutihan esensial yang memerlukan dua gerbang XOR untuk ditambahkan ke desain: satu untuk pra - whitening pada plaintext dan satu untuk post-whitening pada ciphertext [11]. Implementasi DES yang ringan ini membutuhkan 2168 GE (gate setara) dan menawarkan throughput 44,4 Kb/s pada daya 100 KHz [28]. Serangan yang mengancam DES tidak serta merta mengancam DESLX karena properti dari satu S-box yang digunakan dalam DESLX berbeda dengan properti dari DES S-box.
   c) Keamanan Tinggi dan Ringan *HIGHT*: cipher blok ringan beroperasi pada blok 64-bit menggunakan kunci 128-bit yang dihasilkan pada enkripsi dan dekripsi menggunakan struktur GFS. Enkripsi menggunakan *HIGHT* terjadi dengan 32 putaran. Implementasi ringan ini membutuhkan 3048 GE (setara gerbang) dan menawarkan throughput 188,2 Kb/s pada daya 100 KHz. Block cipher ini rentan terhadap serangan saturasi [12].
   d) *PRESENT*: Ini dianggap sebagai algoritma kriptografi ultra-ringan yang beroperasi pada blok 64-bit menggunakan ukuran kunci 80-bit atau 128-bit. Enkripsi menggunakan PRESENT terjadi dalam 32 putaran. Lapisan substitusi SPN saat ini menggunakan S-box 4-bit untuk input dan output. Block cipher ini dikenal dengan tingkat keamanan yang tinggi dan kesederhanaannya. *PRESENT* membutuhkan hingga 1570 GE (setara gerbang) saat menggunakan kunci 80-bit dan hingga 1884 GE saat menggunakan kunci 128-bit, dan menawarkan throughput hingga 200 Kb/s pada daya 100 KHz. *PRESENT* rentan terhadap serangan diferensial [13].
   e) KATAN dan KTANTAN: Ini adalah stream cipher yang beroperasi pada blok ukuran 32-bit, 48-bit atau 64-bit dan keduanya menggunakan kunci 80-bit. Enkripsi oleh KATAN dan KTANTAN terjadi dalam 254 putaran. Perbedaan utama antara desain adalah bahwa KTANTAN membutuhkan sekitar setengah dari ekuivalen gerbang yang dibutuhkan KATAN untuk implementasi perangkat keras dengan semua ukuran blok. Untuk blok berukuran 32-bit, KTANTAN membutuhkan 462 EG, sedangkan KATAN membutuhkan 802 EG. Untuk blok berukuran 48- bit, KTANTAN membutuhkan 588 EG, sedangkan KATAN membutuhkan 927 EG. Akhirnya, untuk blok 64- ukuran bit, KTANTAN membutuhkan 688 EG, sedangkan KATAN membutuhkan 1054 EG [14]. KATAN dan KTANTAN menawarkan throughput 12,5, 18,8 dan 25,1 untuk Kb/s pada daya 100 KHz saat mengenkripsi blok masing-masing berukuran 32-bit, 48-bit, dan 64-bit [14]. Kedua cipher rentan terhadap serangan diferensial.
   f) *PRINCE*: sandi ringan yang dikenal menggunakan properti refleksi: enkripsi dan dekripsinya sama, tetapi masing-masing

menggunakan kunci yang berbeda untuk mengurangi persyaratan desain. *Cipher* ini beroperasi pada blok berukuran 64-bit menggunakan kunci 128-bit, dan menggunakan struktur SPN. RSA: pendekatan boros sumber daya yang tidak terlalu fungsional untuk pengoptimalan ringan. RSA bergantung pada pemilihan dua bilangan prima besar untuk menemukan kunci publik dan kunci privatnya, yang biasanya antara 1024 dan 4096 bit. 121 Enkripsi oleh *PRINCE* terjadi dalam 12 putaran *PRESENT* membutuhkan hingga 3491 GE (setara gerbang) dan menawarkan *throughput* hingga 533,3 Kb/s pada daya 100 KHz. *PRINCE* rentan terhadap serangan refleksi [15].

Tabel 12. Pendekatan LWC Simetris
(Sumber: Hala Tawalbeh, Sonia Hashish, Loai Tawalbeh dan Anwar Aldairi. Dalam *Journal of Information Assurance and Security*, Volume 12, pp. 118-123)

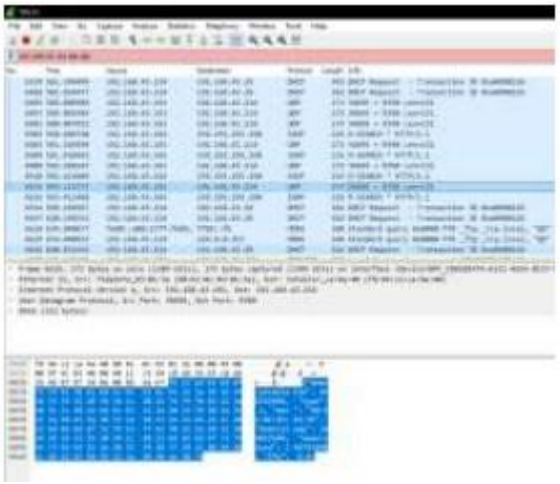
Cipher name	*Block Size*	*Key Size*	*# of rounds*	*Structure*	*# of GEs*	*Throughput Kb/s at 100 KHz*	*Vulnerable Attacks*
AES	128-bit	128-bit	12	SPN	3100	80	MITM attack
		192-bit	14				
		256-bit	16				
DESLX	64-bit	184-bit	64	Feistel	2168	44.4	-
HIGHT	64-bit	128-bit	32	GFS	3048	188.2	Saturation attack
PRESENT	64-bit	80-bit	31	SPN	up to 1570	up to 200	Differential attack
		128-bit			up to 1884		
PRINCE	64-bit	128-bit	12	SPN	up to 3491	up to 533.3	Reflection attack
KATAN	32-bit	80-bit	254	Stream-cipher	802	12.5	Deferential attack
	48-bit				927	18.8	
	64-bit				1054	25.1	
KTANTAN	32-bit	80-bit	254	Stream-cipher	462	12.5	Deferential attack
	48-bit				588	18.8	
	64-bit				688	25.1	
TWIN	64-bit	80-bit	36	GFN	1799	178	Biclique attack
		128-bit			2285		

2. Pendekatan Kriptografi Ringan Asimetris, Sebagai berikut:
   a) Kriptografi Kurva Elliptik ECC: ECC ringan dianggap sebagai metode kunci publik yang paling efisien karena memerlukan konsumsi daya yang lebih sedikit, area yang lebih kecil, dan siklus clock yang lebih sedikit [16]. Ini didasarkan pada sistem aljabar dan menggunakan kunci kecil yang dihasilkan menggunakan algoritma diskrit. Beberapa implementasi yang dioptimalkan diusulkan untuk ECC ringan. Sebagian besar desain membutuhkan tidak kurang dari 10.000 GE [17]. Lima operasi disediakan dalam aritmatika medan utama: Perkalian, Penambahan, Pengurangan, Bagi-bagi, dan Pengurangan. Mengurangi kompleksitas operasi Perkalian adalah yang paling layak dari operasi ini karena dapat dilakukan untuk implementasi hingga 30 k GE. Optimalisasi terjadi dengan menggunakan bit shifting dan penambahan untuk perkalian daripada menggunakan mikroprosesor
   b) RSA: pendekatan boros sumber daya yang tidak terlalu fungsional untuk pengoptimalan ringan. RSA bergantung pada pemilihan dua bilangan prima besar untuk menemukan kunci publik dan kunci privatnya, yang biasanya antara 1024 dan 4096 bit. Pendekatan konvensional kriptografi asimetris masih belum menjanjikan untuk bobot ringan karena tidak ada implementasi yang diusulkan memiliki waktu eksekusi yang wajar [18].

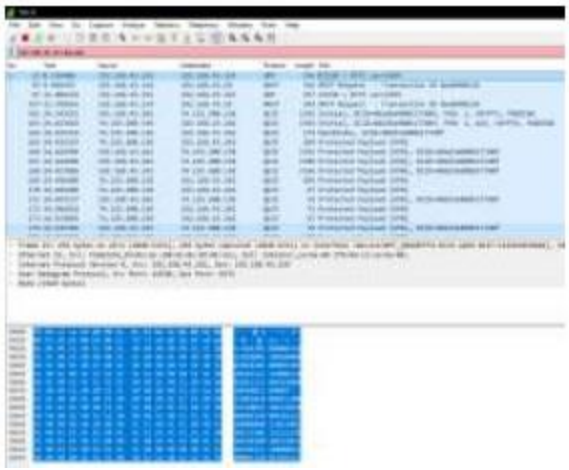
E. Pertanyaan 5

P5 : Bagaimana hasil pengujian serangan aktif dengan metode *known plaintext attack* mengunakan algoritme SPECK mampu untuk dapat bertahan dari serangan tersebut selama 24 jam?

pengujian serangan yang dilakukan pada penelitian ini. Pada Gambar 6 dan Gambar 7 merupakan hasil dari pengujian serangan pasif menggunakan alat bantu yaitu aplikasi *Wireshark*. Pada Gambar 6 memiliki skenario pihak ketiga melakukan serangan sniffing dengan aplikasi Wireshark dengan kondisi sistem monitoring belum dilakukan implementasi algoritme SPECK. Dapat dilihat bahwa pada Gambar 7 memperlihatkan komunikasi antar manager dan agent dapat dilihat dengan jelas oleh aplikasi Wireshark. Sedangkan Gambar 6 memiliki kondisi sistem monitoring sudah diimplementasikan algoritme SPECK. Dapat dilihat bahwa pada Gambar 8 aplikasi Wireshark hanya melihat angka biner acak saja. Pada Gambar 8 menunjukkan pengujian serangan aktif dengan cara known plaintext attack untuk mendapatkan pasangan key dan *initialization vector* (IV) dengan kondisi penyerang sudah memiliki plaintext. Serangan dilakukan selama 24 jam, akan tetapi penyerang masih tidak mendapatkan pasangan key dan IV yang benar.



Gambar 6. Hasil capture pada Wireshark sistem monitoring sebelum implementasi algoritme SPECK (Sumber: Yohanes Heryka Febriarso, Ari Kusyanti dan Primantara Hari Trisnawan. Dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 5, No. 5, hlm. 1793-1801)



Gambar 7. Hasil capture pada Wireshark sistem monitoring sesudah implementasi algoritme SPECK (Sumber: Yohanes Heryka Febriarso, Ari Kusyanti dan Primantara Hari Trisnawan. Dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 5, No. 5, hlm. 1793-1801)

Gambar 8. Hasil capture pada command line pada Windows 10 ketika melakukan serangan known plaintext attack selama 24 jam
(Sumber: Yohanes Heryka Febriarso, Ari Kusyanti dan Primantara Hari Trisnawan. Dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 5, No. 5, hlm. 1793-1801)

IV. DISKUSI

Penggunaan komponen apa saja yang ada pada kriptografi ringan dan IoT yang menjadi penyebab masalah. Dalam kasus implementasi perangkat lunak, implementasi ukuran dan konsumsi RAM melalui penempatan (byte per siklus) adalah metrik yang lebih disukai untuk aplikasi ringan. Semakin kecil semakin baik. Dalam kasus perangkat lunak, framework *Fair Evaluation of Lightweight Cryptographic Systems* (FELICS) diusulkan untuk mengevaluasi kinerja blok ringan atau kinerja stream cipher dalam ukuran implementasi, konsumsi RAM dan waktu yang dibutuhkan untuk melakukan operasi tertentu. CLEFIA adalah cipher blok ringan yang dipelajari dengan baik dan ditulis oleh Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai dan Tetsu Iwata dan dapat diimplementasikan dengan gerbang 6K. Dengan demikian dapat digunakan untuk mencocokkan enkripsi dengan kemampuan perangkat. Jika itu adalah perangkat berdaya rendah dengan memori terbatas dan jejak fisik yang relatif kecil, kita dapat menggunakan ukuran blok 32-bit dan kunci 80-bit, hanya dengan beberapa putaran. Tetapi kami dapat meningkatkan keamanan jika perangkat dapat mengatasinya dan menggunakan ukuran blok 128-bit dan kunci 128-bit. Itu juga bisa fleksibel, di mana satu perubahan di kedua sisi dapat meningkatkan atau mengurangi persyaratan.

Waktu eksekusi proses tanda tangan digital membutuhkan waktu lebih lama dibandingkan dengan waktu eksekusi proses verifikasi untuk Tanda Tangan Digital Menggunakan Algoritme Elgamal Signature Scheme. Proses pembangkitan kunci dilakukan oleh pengirim. Kunci yang dibangkitkan berguna untuk memberikan tanda tangan digital pada data. Verifikasi tanda tangan digital menghasilkan hasil perbandingan dari dua proses komputasi. Pada Tabel 5 adalah hasil verifikasi keabsahan tanda tangan digital. Berdasarkan pada Tabel 5, perbandingan dari hasil dua proses tersebut memiliki nilai yang sama yang artinya verifikasi keabsahan tanda tangan digital berhasil dilakukan. Analisis keamanan algoritme dilihat dari hasil verifikasi *signature* saat ada perubahan pada m dan pada pasangan tanda tangan digital (r,s).Namun pada Tabel 4 dan 5, perubahan pada m ataupun data pasangan tanda tangan digital (r,s) menyebabkan perbedaan nilai pada proses komputasi 1 dan proses komputasi 2. Perbedaan nilai yang dihasilkan menujukkan bahwa proses verifikasi telah gagal. Panjang kunci private dihasilkan pada penelitian ini sebesar 18 bit. Jika dilakukan metode brute force, maka akan menghasilkan 2 18 = 262 144 kemungkinan bilangan untuk menebak nilai dari kunci private yang dihasilkan. Jika brute force dilakukan pada PC penelitian ini membutuhkan waktu sekitar 262 detik atau sekitar empat menit. Idealnya untuk panjang kunci asimetris, minimal 2048 bit seperti pada RSA.

***Delay* dalam pembacaan arduino dengan algoritme AEGIS.** Pengujian kinerja waktu sistem bertujuan untuk mengetahui perbedaan signifikan karena implementasi AEGIS 128. Pada sistem *middleware* yang menggunakan protokol MQTT atau menggunakan protokol CoAP akan diuji data sampel waktu pemrosesan saat AEGIS 128 diimplementasikan dan tanpa AEGIS 128. Pengujian *Quality of Service* (QoS) merupakan pengujian untuk mengukur kualitas layanan yang diperoleh dari suatu jaringan komunikasi. Dasar pengujian QoS yaitu menganalisis parameter dari *packet loss*, *delay* dan jitter. *Packet loss* merupakan indikasi bahwa terdapat paket yang hilang atau gagal terkirim ke

tujuan. Nilai packet loss didapatkan dari nilai *expected* dan nilai actual. Nilai *expected* yaitu nilai harapan dari banyak paket yang terkirim ke tujuan, sedangkan nilai actual adalah nilai dari paket yang berhasil terkirim ke tujuan. *Delay* adalah waktu jeda paket terkirim dari sumber sampai dengan waktu pengiriman ACK dari tujuan. Jitter adalah variasi nilai dari *delay* pengiriman paket. Namun Berdasarkan Tabel 8 hasil pengujian pengiriman data dengan protokol CoAP dari sensor nodeMCU ESP8266 ke *middleware* tanpa mekanisme keamanan. Menghasilkan *success rate* 100 % dan packet loss 0 %. Dengan nilai rata-rata delay adalah 14,3 detik dan jitter adalah 0,0013 detik. Berdasarkan Tabel 9 hasil pengujian pengiriman data pada protokol MQTT dari sensor nodeMCU ESP8266 ke *middleware* dengan algoritme AEGIS 128. Menghasilkan *success rate* 99 % dan *packet loss* 1%. Dengan nilai rata-rata delay adalah 26,3 detik dan jitter adalah 1,46 detik.

Keamanan dunia maya dan serangan untuk *Wireless Sensor Networks* (WSNs). Kriptografi Ringan mengacu pada satu set algoritma kriptografi simetris dan asimetris yang dirancang untuk memastikan keamanan yang memadai dalam sistem yang memiliki kendala khusus untuk menjalankan lingkungan dan kemampuan daya seperti WNS. Ringan dalam kriptografi dapat dicapai pada tingkat perangkat keras dan perangkat lunak, di mana ukuran chip, energi di tingkat perangkat keras, ukuran kode, dan kompleksitas RAM di tingkat perangkat lunak digunakan untuk mengukur tingkat optimasi yang dipenuhi dengan menerapkan LWC. Namun, serangan *Man-in-the-Middle* (MITM) masih mengancam AES Ringan. Serangan yang mengancam DES tidak serta merta mengancam DESLX karena properti dari satu S-box yang digunakan dalam DESLX berbeda dengan properti dari DES S-box. *HIGHT* Block cipher ini rentan terhadap serangan saturasi. *PRESENT*, KATAN dan KTANTAN rentan terhadap serangan diferensial. *PRINCE* rentan terhadap serangan refleksi.

Hasil pengujian serangan aktif dengan metode *known plaintext attack* mengunakan algoritme SPECK mampu untuk dapat bertahan dari serangan tersebut selama 24 jam. hasil dari pengujian serangan pasif menggunakan alat bantu yaitu aplikasi Wireshark. Pada Gambar 6 memiliki skenario pihak ketiga melakukan serangan sniffing dengan aplikasi Wireshark dengan kondisi sistem monitoring belum dilakukan implementasi algoritme SPECK. Dapat dilihat bahwa pada Gambar 7 memperlihatkan komunikasi antar manager dan agent dapat dilihat dengan jelas oleh aplikasi Wireshark. Sedangkan Gambar 6 memiliki kondisi sistem monitoring sudah diimplementasikan algoritme SPECK. Dapat dilihat bahwa pada Gambar 8 aplikasi Wireshark hanya melihat angka biner acak saja. Pada Gambar 8 menunjukkan pengujian serangan aktif dengan cara known plaintext attack untuk mendapatkan pasangan key dan *initialization vector* (IV) dengan kondisi penyerang sudah memiliki plaintext. Serangan dilakukan selama 24 jam, akan tetapi penyerang masih tidak mendapatkan pasangan key dan IV yang benar. Namun apabila adanya serangan aktif selama 24 jam perlu adanya uji coba kembali.

V. KESIMPULAN

Kami melakukan Tinjauan Literatur Sistematik untuk menyelidiki kasus penggunaan kriptografi ringan dalam literatur dan faktor mana yang memengaruhi metode kriptografi konvensional tidak baik untuk berskala dunia dengan sistem tertanam pada jaringan sensor. Dengan demikian, metode kriptografi ringan diusulkan untuk mengatasi banyak masalah kriptografi konvensional ini termasuk kendala yang berkaitan dengan ukuran fisik, persyaratan pemrosesan, keterbatasan memori dan yang menguras energi. Lalu Waktu eksekusi proses tanda tangan digital membutuhkan waktu lebih lama dibandingkan dengan waktu eksekusi proses verifikasi. Algoritme Elgamal signature scheme membutuhkan waktu dua kali lebih lama karena banyaknya perhitungan sistematis pada perangkat Arduino Uno. Lalu terdapat *delay* dalam pembacaan arduino dengan algoritme AEGIS. keamanan dunia maya dan serangan untuk *Wireless Sensor Networks* (WSNs) harus diperhatikan. Lalu hasil pengujian serangan aktif dengan metode known plaintext attack mengunakan algoritme SPECK mampu untuk dapat bertahan dari serangan tersebut selama 24 jam.

Dalam penelitian ini dilakukan untuk mengatasi masalah di masa depan akan terdiri dari pengujian kriptografi yang aman dan paling cocok dalam merancang arsitektur berlapis untuk aplikasi IoT.

DAFTAR PUSTAKA

[1] Rose K, Chapin L, Eldridge S. 2015. The Internet of Things: an Overview. Geneva (CH): Internet Society

[2] Buchanan, William J., Li, Shancang., dan Asif, Rameez. 2017. *Lightweight cryptography methods. Journal of Cyber Security Technology*, ISSN: 2374-2917 (Print) 2374-2925.

[3] Qisthina, Selfi., dan Neyman. Shelvie Nidya. 2021. Pengamanan Internet of Things untuk Tanda Tangan Digital Menggunakan Algoritme Elgamal Signature Scheme. Jurnal Ilmu Komputer dan Agri-Informatika, Volume 8 Nomor 1 halaman 69 - 78, e ISSN: 2654-9735, p ISSN: 2089-6026.

[4] Nawawi, Helmy Rafi., Kusyanti, Ari., dan Primananda, Rakhmadhany. 2019. Implementasi Algoritme AEGIS untuk *Payload* Data Protokol MQTT dan CoAP pada Raspberry Pi 3. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 3, No. 8, Agustus 2019, hlm. 7879-7887.

[5] Tawalbeh, Hala., Hashish, Sonia., Tawalbeh, Loai., dan Aldairi, Anwar. 2017. *Security in Wireless Sensor Networks Using Lightweight Cryptography*. *Journal of Information Assurance and Security*. ISSN 1554-1010 Volume 12 (2017) pp. 118-123.

[6] Febriarso, Yohanes Heryka., Kusyanti, Ari., dan Trisnawan, Primantara Hari. 2021. Implementasi Algoritme SPECK dalam Sistem Monitoring Sumber Daya pada Raspberry Pi. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN:

2548-964X Vol. 5, No. 5, Mei 2021, hlm. 1793-1801.

[7] Ariyus, D., 2008. Pengantar Ilmu Kriptografi: Teori, Analisis, dan Implementasi. Yogyakarta: ANDI.

[8] Biryukov A, Perrin L. *State of the art in lightweight symmetric cryptography*. IACR *Cryptology ePrint Archive*, 2017:511.

[9] Dinu D, Biryukov A, Großschädl J, et al. FELICS – *fair evaluation of lightweight cryptographic systems. In: NIST Workshop on Lightweight Cryptography* 2015, *National Institute of Standards and Technology* (NIST) (2015)

[10] Ahmed, E.G., Shaaban, E. and Hashem, M. "*Lightweight mix columns implementation for AES*." *In Proceedings of the 9th* WSEAS *international conference on Applied informatics and communications*, pp. 253- 258. 2009.

[11] Leander, G., Paar, C., Poschmann, A. and Schramm, K. "*New lightweight* DES *variants*." *In International Workshop on Fast Software Encryption*, pp. 196-210. Springer, Berlin, Heidelberg, 2007.

[12] Jana, S., Bhaumik, J. and Maiti, M.K. "Survey on lightweight block cipher." International Journal of Soft Computing and Engineering 3 (2013): 183-187.

[13] Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J., Seurin, Y. and Vikkelsoe, C. "PRESENT: An ultra-lightweight block cipher." In CHES, vol. 4727, pp. 450-466. 2007.

[14] De Canniere, C., Dunkelman, O. and Knežević, M. "KATAN and KTANTAN—a family of small and efficient hardware-oriented block ciphers." In Cryptographic Hardware and Embedded Systems-CHES 2009, pp. 272-288. Springer, Berlin, Heidelberg, 2009.

[15] Borghoff, J., Canteaut, A., Güneysu, T., Kavun, E.B., Knezevic, M., Knudsen, L.R., Leander, G., Nikov, V., Paar, C., Rechberger, C. and Rombouts, P. "PRINCE–a low-latency block cipher for pervasive computing applications." In International Conference on the Theory and Application of Cryptology and Information Security, pp. 208-225. Springer, Berlin, Heidelberg, 2012.

[16] Tawalbeh, L. A., Tenca, A., Park, S., & Koc, C. (2005, August). An efficient hardware architecture of a scalable elliptic curve crypto-processor over GF (2n). In Optics and Photonics 2005 (pp. 59100Q-59100Q). International Society for Optics and Photonics.

[17] Katagi, M. and Moriai, S. "Lightweight cryptography for the internet of things." Sony Corporation (2008): 7-10.

[18] Tawalbeh, L. A. A., & Sweidan, S. (2010). Hardware design and implementation of ElGamal public-key cryptography algorithm. Information Security Journal: A Global Perspective, 19(5), 243-252.