Artikel ini telah dipresentasikan dalam *Innovative and Creative Information Technology Conference* (ICITech) dengan tema "*E-Transaction and Power Play"* yang diselenggarakan oleh Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana – Salatiga, pada 24 November 2016.

Keamanan Data Pada Perangkat *Internet Of Things* Menggunakan Metode *Public-Key Cryptography*

Marsel Sampe Asang[1], Irwan Sembiring[2]
[12]Universitas Kristen Satya Wacana
Jl. Diponegoro 52-60, Salatiga 50711, Indonesia
Email : [1)] marselsampe@gmail.com, [2)]irwan@staff.uksw.edu

Abstrak

Developments of IoT in every sector gives a new problem, it is information security. There are many issues about data theft and hacking in IoT system, this is because the weak of system security. IoT device is a part of system that needs special care for that issues. This research focuses on data security in IoT decive like microcontroller that only has a limited resources. Firstly is implement public-key cryptography using RSA algorithm in the device. Then, make an analysis process to see the effectiveness and efficiency of the method. Finally result of this paper shows that implementation of public-key cryptography in IoT device for data security still make the device effective and efficient to work only for small key.

Keywords : IoT, Keamanan, *Public-Key Cryptography, RSA*

1. Pendahuluan

Internet of Things merupakan teknologi komunikasi masa depan, dimana setiap objek yang ada di sekitar kita akan ditanamkan mikrokontroler, yang akan membuat objek tersebut dapat berkomunikasi dengan objek yang lain dan juga dengan pengguna [1]. Dalam *IoT,* objek-objek tersebut dalam disebut sebagai *things*. *IoT* menggunakan internet sebagai konektivitasnya sehingga dapat memberikan informasi secara *real-time* tanpa batasan jarak dan tanpa campur tangan manusia, sering disebut komunikasi mesin-ke-mesin. Contoh pengembangan *IoT* yaitu hadirnya sistem *smart*, seperti *smart home, smart energy, smart governance*, *smart infrastructure,* dll.

Penggunaan *IoT* yang semakin berkembang memunculkan masalah baru yaitu faktor keamanan, Keamanan yang baik merupakan salah satu tantangan terbesar dalam membangun *IoT*. Secara umum, kerangka *IoT* terdiri dari 3 layer yaitu layer sensor, layer jaringan dan layer aplikasi [2]. Setiap layer harus memiliki protokol keamanannya agar dapat menjaga integritas, ketersediaan dan kerahasiaan data, terutama saat seseorang melakukan serangan ke sistem *IoT* tersebut. Jenis-jenis serangan yang umum dilakukan yaitu *Botnet, Man-In-The-Middle, Data & Identity Theft, Social Engineering,* dan *Denial of Service* [3].

Pada tahun 2014, pakar-pakar keamanan mendemonstrasikan bagaimana mereka dapat menyerang jaringan yang digunakan untuk menyalakan lampu otomatis dan mendapatkan *username* dan *password Wi-Fi* dari pemilik rumah yang menggunakan lampu tersebut [4]. Jika melihat cara kerja *IoT*, mikrokontroler akan mengirim dan menerima data dari server melalui jaringan umum atau internet. Data yang dikirim tersebut masih berupa *plaintext*, artinya format data tersebut masih dapat dibaca oleh siapa saja. Apabila terjadi serangan *Data & Identity Theft* dalam

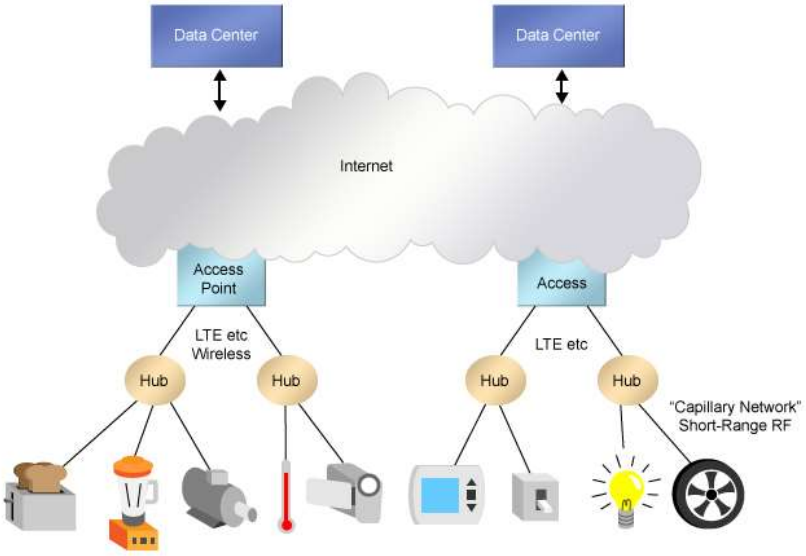
hal ini pelaku melakukan *sniffing* ke jaringan yang digunakan oleh mikrokontroler, maka pelaku dapat membaca data-data perangkat tersebut. Kerahasiaan data sudah tidak lagi terjamin.

Salah satu solusi yang dapat digunakan untuk mengamankan data di *IoT* yaitu menggunakan metode enkripsi data. Enkripsi data merupakan bagian dari ilmu kriptologi, ilmu tentang menyembunyikan dan merahasiakan pesan. Ilmu ini mempelajari tentang kriptografi dan kriptoanalisis [5]. Dalam kriptografi, pesan awal yang dinamakan *plaintext* akan dienkripsi menggunakan algoritma tertentu dan menggunakan kunci rahasia. Pesan yang sudah dienkripsi akan disebut *chipertext*, pesan ini sudah tidak dapat dibaca jika seseorang tidak mengetahui algoritma dan kunci untuk membukanya. Proses untuk merubah pesan dari *chipertext* ke *plaintext* sehingga dapat dibaca kembali disebut proses dekripsi. Enkripsi yang kuat dapat menjamin kerahasiaan data.

Penelitian ini akan berfokus pada implementasi dan analisis metode *public-key cryptography* di perangkat *IoT* untuk pengamanan data, khususnya dalam menenkripsi data yang akan dikirim dari mikrokontroler ke server. Algoritma enkripsi yang digunakan yaitu algoritma *RSA*. Setelah mengimplementasikan metode tersebut, akan dilakukan analisis untuk menguji efektifitas dan efisiensi waktu proses serta penggunaan memori dari perangkat yang digunakan.

2. *Internet of Things*

Internet of Things merupakan paradigma baru dalam penggunaan teknologi komunikasi *wireless* yang terus berkembang. Konsep utama dalam *IoT* yaitu setiap benda atau objek, seperti *Radio-Frequency Identification (RFID) tag*, sensor, aktuator, ponsel pintar, mikrokontroler, dll. dapat saling berinteraksi untuk mencapai suatu tujuan tertentu. Setiap objek akan memiliki alamat penanda yang unik sehingga dapat dibedakan satu sama lain [1].



Gambar 1. Layer *Internet of Things* [6]

IoT terdiri dari 3 layer yaitu layer sensor, layer jaringan, dan layer aplikasi. Layer sensor merupakan layer terluar dari *IoT*, layer ini sudah berbentuk fisik dan dapat disentuh. Disinilah ditempatkan *RFID tags*, sensor, aktuator, dll. Layer jaringan meliputi jaringan *wireless* atau kabel, media jaringan dan jaringan utama. Layer jaringan merupakan jalur pengiriman data dari layer sensor ke layer aplikasi. Layer aplikasi merupakan sistem yang memiliki antarmuka untuk menampilkan semua data yang telah diolah menjadi informasi kepada pengguna [2].

IoT bisa dikategorikan sebagai sistem yang sangat kompleks dan juga rentan terhadap serangan *cyber*. Semua layer *IoT* rentan terhadap serangan, terutama pada layer sensor. Perangkat-perangkat tersebut terkoneksi melalui jaringan publik atau internet, sehingga pelaku serangan dapat menyerangnya dengan menggunakan teknik tertentu. Salah satunya isu serangan yang marak terjadi yaitu *Data & Identity Theft*, disini pelaku mencuri data yang sementara dikirim dari perangkat menuju ke server data dan sebaliknya. Pelaku dapat membaca data yang berhasil dicuri karena data tersebut belum diamankan sebelum dikirim.

3. Enkripsi *Public-Key*

Dalam dunia kriptografi terdapat dua metode enkripsi yang dapat digunakan untuk menjaga kerahasiaan data, yaitu metode enkripsi simetris dan asimetris. Metode enkripsi simetris sering disebut sebagai *single-key cryptography* dan metode enkripsi asimetris sering disebut *public-key cryptography*.

Enkripsi simetris disebut *single-key cryptography* karena metode ini memakai kunci yang sama untuk kegiatan enkripsi dan dekripsi. Bila mengirim pesan dengan menggunakan metode ini, si penerima pesan harus diberitahu kunci dari pesan tersebut agar bisa mendekripsi pesan yang diterima. Keamanan enkripsi dari metode ini tergantung pada kunci. Jika kunci tersebut diketahui oleh orang lain maka orang tersebut akan dapat melakukan enkripsi dan dekripsi terhadap pesan [7]. Terdapat banyak jenis algoritma enkripsi simetris diantaranya yang paling populer adalah algoritma *DES (Data Encryption Standard)*. Dua kelemahan utama pada metode enkripsi simetris, yaitu pada saat proses distribusi kunci dan otentikasi pengirim. Oleh karenanya untuk mengatasi masalah tersebut maka diciptakanlah metode enkripsi asimetris.

Metode enkripsi asimetris sering juga disebut *public-key cryptography*, artinya kunci yang digunakan untuk melakukan enkripsi dan dekripsi berbeda [7]. Dalam enkripsi asimetris kunci terbagi menjadi dua bagian yaitu : kunci publik dan kunci privat. Kunci publik merupakan kunci yang dibuat untuk disebarkan kepada umum dan digunakan untuk melakukan enkripsi. Kunci privat harus dirahasikan karena kunci ini yang akan digunakan untuk mendekripsi data. Algoritma yang paling populer digunakan pada metode ini adalah algoritma *RSA*. Hadirnya metode ini menjawab dua permasalahan utama dalam metode simetris. Tetapi metode ini juga memiliki kelemahan yaitu pada efisiensi waktu. Enkripsi dan dekripsi data umumnya lebih lambat daripada metode simetris, karena dalam prosesnya

menggunakan bilangan yang besar dan melibatkan operasi perpangkatan yang besar.

4. Algoritma *RSA*

Algoritma *RSA* termasuk dalam kategori metode enkripsi asimetris. Kata *RSA* merupakan singkatan dari 3 nama penemu algoritma tersebut, Ron Rivest, Adi Shamir dan Len Adleman, yang ditemukan pada tahun 1977. Algoritma *RSA* merupakan algoritma enkripsi asimetris yang paling banyak digunakan di dunia. Algoritma *RSA* juga cocok digunakan untuk membuat tanda tangan digital [8]. Keamanan algoritma *RSA* terletak pada sulitnya memfaktorkan bilangan yang sangat besar menjadi faktor-faktor prima. Pemfaktoran dilakukan untuk memperoleh kunci privat yang menjadi kunci untuk melakukan dekripsi pesan. Cara kerja algoritma *RSA* dimulai dari proses pembangkitan kunci, proses enkripsi pesan, dan proses dekripsi pesan.

4.1 Pembangkitan Kunci

Proses pembangkitan kunci merupakan proses untuk membuat pasangan kunci publik dan kunci privat. Proses ini dilakukan di sisi A.

1. Pilih 2 bilangan prima yang besar, p dan q.
2. Hitung N = p*q dan phi (φ) = (p-1)(q-1).
3. Pilih bilangan bulat e antara satu dan phi ($1 < e < \varphi$) yang juga merupakan koprima dari phi.
4. Hitung d hingga $d*e \equiv 1 \pmod{\varphi}$.

Kunci publik adalah (n, e) dan nilainya akan dikirimkan ke B, sedangkan kunci privat adalah (d, p, q). Nilai dari d, p, q, dan phi harus selalu dirahasiakan. Sampai saat ini, panjang kunci *RSA* yang bisa dikatakan aman yaitu 1024 bit ke atas.

Kode Program 1. Contoh kunci n 1024 bit dalam format *hexadecimal*

```
0A 66 79 1D C6 98 81 68 DE 7A B7 74 19 BB 7F B0
C0 01 C6 27 10 27 00 75 14 29 42 E1 9A 8D 8C 51
D0 53 B3 E3 78 2A 1D E5 DC 5A F4 EB E9 94 68 17
01 14 A1 DF E6 7C DC 9A 9A F5 5D 65 56 20 BB AB
```

4.2 Enkripsi Pesan

Proses enkripsi pesan dilakukan dilakukan di sisi B menggunakan kunci publik milik A.

1. A mengirim kunci publik ke B (n, e).
2. B menerima kunci publik dari A.
3. Pesan *plaintext* diubah menjadi angka positif m ($1 < m < n$).
4. Hitung *chipertext* $c = m^e \bmod n$.
5. B mengirim *chipertext* c ke A.

4.3 Dekripsi Pesan

Proses dekripsi pesan dilakukan di sisi A menggunakan kunci privat milik A.

1. A menerima *chipertext* dari B.
2. Menggunakan kunci privat (n, d), hitung $m = c^d \bmod n$.
3. Bentuk kembali *plaintext* menggunakan m.

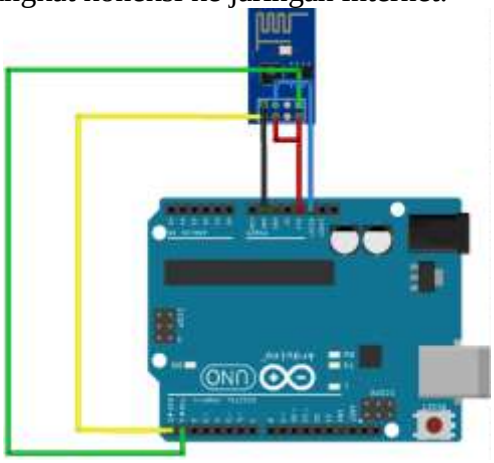
Selama kunci privat masih aman dan tidak diketahui pihak lain termasuk sisi B, maka pesan *chipertext* tidak mungkin didekripsi oleh pihak lain sekalipun dengan menggunakan teknik *bruteforce* yang akan memakan waktu puluhan tahun sampai menemukan kombinasi kunci yang tepat. Tetapi kelemahan *RSA* dibanding algoritma enkripsi yang lain terletak pada saat proses perhitungannya yang sangat lambat. Oleh karena itu, *RSA* lebih cocok digunakan untuk menenkripsi pesan-pesan yang kecil.

5. *Public-Key Cryptography* pada Perangkat *IoT*

Perangkat *IoT* yang dibahas dalam penelitian ini lebih fokus pada perangkat mikrokontroler. Umumnya data yang dikirim dari mikrokontroler ke server masih berbentuk *plaintext*, artinya data tersebut masih dapat dibaca oleh siapa saja. Metode enkripsi dapat digunakan untuk mengamankan data tersebut. Metode enkripsi yang sesuai digunakan yaitu metode enkripsi simetris, karena metode ini lebih cepat dan efisien penggunaan memorinya dibanding metode enkripsi asimetris. Tetapi kunci enkripsi simetris yang tersimpan di perangkat masih rentan diketahui oleh pihak lain, dalam hal ini pihak tersebut membuka *source code* yang tersimpan di memori perangkat. Oleh karena itu, metode yang lebih cocok digunakan yaitu metode enkripsi asimetris. Sekalipun pihak lain mengetahui kunci publik yang tersimpan di memori perangkat, mereka tetap tidak dapat mendekripsi pesan *chipertext* yang dicuri. Algoritma enkripsi asimetris yang digunakan yaitu algoritma *RSA*.

5.1 *Prototype* Perangkat *IoT*

Prototype pada penelitian ini dirancang menggunakan *board Arduino UNO R3* sebagai basis mikrokontrolernya. *Prototype* ini menggunakan modul *wireless ESP8266* sebagai perangkat koneksi ke jaringan internet.



Gambar 2. *Prototype* Perangkat *IoT*

Tabel 1. Spesifikasi *Arduino UNO R3*

Hardware	**Tipe / Besar**
Mikrokontroler	ATmega328P
Flash Memory	32 *KB*
SRAM	2 *KB*
EEPROM	1 *KB*
Clock Speed	16 *MHz*

Pada *prototype* ini, spesifikasi memori untuk proses data (*SRAM*) hanya sebesar 2 *KB*, ini sudah termasuk untuk alokasi memori yang nantinya digunakan oleh modul *wireless ESP8266*. Dari segi kecepatan pemrosesan perhitungan, *prototype* ini hanya memiliki *clock speed* sebesar 16 *MHz.*

5.2 Implementasi Algoritma *RSA*

Tantangan dalam mengimplementasikan enkripsi *RSA* pada *prototype* ini terletak pada efisiensi penggunaan memori dan lama proses enkripsi. Proses enkripsi data akan dibuat lebih ringkas agar sesuai dengan kondisi sumber daya yang dimiliki perangkat. Walaupun enkripsi dibuat ringkas tetapi kekuatan enkripsi tetap dipertahankan. Pada arsitektur sistem berikut, perangkat *IoT* akan disebut sisi Klien sedangkan server data akan disebut sisi Server. Klien dan Server berkomunikasi melalui jaringan internet dengan menggunakan protocol *TCP*.

Gambar 3. Arsitektur Sistem

Berikut merupakan tahapan proses enkripsi *RSA* pada arsitektur sistem :

1. Klien membuka koneksi ke Server untuk mengambil kunci publik (n, e).
2. Server mengirim kunci publiknya.
3. Klien menerima kunci publik.
4. Klien menenkripsi *plaintext* dengan kunci publik.
5. Klien mengirim *chipertext* yang sudah dienkripsi ke Server.
6. Server menerima *chipertext* dan mendekripsinya menggunakan kunci privat (n, d).

Tahap 1 sampai 3 merupakan proses distribusi kunci publik dari Server ke Klien, tahapan ini hanya dilakukan sekali saja sesaat setelah perangkat dijalankan. Tahap 4 sampai 6 merupakan proses pengiriman data dari Klien ke Server. Proses ini akan berulang sesuai dengan interval waktu yang telah ditentukan di perangkat.

5.3 Analisis Proses Enkripsi

Analisa hanya dilakukan pada sisi Klien atau perangkat *IoT*. Efektifitas proses enkripsi akan diukur berdasarkan waktu yang dibutuhkan untuk melakukan enkripsi, sedangkan efisiensi akan diukur berdasarkan jumlah penggunaan memori (*SRAM*). Panjang kunci publik *RSA* yang digunakan saat proses pengujian hanya sebesar 16 bit. Panjang data yang dienkripsi sebesar 18 *bytes*.

Kunci n B3 46 (16 bit)
Kunci d 6A (8 bit)
Pesan *Plaintext* (m) marsel sampe asang (18 bytes)
Pesan *Chipertext* (c) C9 7A BA 06 0A DC D5 0B 7A 29 56 65 BF 7A 31 7A 01 F5 (18 *bytes*)
Waktu enkripsi 1.97 detik
Penggunaan Memori 1018 bytes

Dari hasil diatas terlihat jumlah waktu yang dibutuhkan untuk mendekripsi pesan *plaintext* menggunakan algoritma *RSA* adalah 1.97 detik. Memori yang digunakan sebanyak 1018 *bytes*, ini sudah termasuk alokasi memori untuk variabel pendukung dan perhitungan proses enkripsi. Setelah melakukan enkripsi, hasil enkripsi akan dimasukkan ke dalam format *HTTP Header* dan dikirim ke Server.

Kode Program 2. Format *HTTP Header* untuk *request* simpan data ke Server

```
POST /save.php HTTP/1.0
Host: iot.marselsampeasang.web.id
Content-type: application/x-www-form-urlencoded
Content-Length: 28

C9 7A BA 06 0A DC D5 0B 7A 29 56 65 BF 7A 31 7A 01 F5
```

Hasil analisa jumlah waktu dan penggunaan memori diatas murni untuk proses perhitungan enkripsi *RSA*. Ini belum termasuk untuk proses menjalankan modul *wireless ESP8266*, koneksi ke Server dan variabel-variabel pendukung lainnya yang tidak digunakan untuk proses enkripsi. Khusus untuk koneksi ke Server jumlah waktu yang digunakan untuk *request* dan *response* data dari atau ke server tidak dapat diukur, ini disebabkan karena waktu yang dibutuhkan untuk proses tersebut bergantung pada kecepatan internet yang digunakan.

Terlihat bahwa perhitungan enkripsi *RSA* menggunakan panjang kunci 16 bit dan panjang data 18 *bytes*, masih relatif cepat. Tetapi perlu diketahui bahwa panjang kunci *RSA* yang dikatakan aman yaitu sekitar 1024 bit ke atas. Bisa disimpulkan

bahwa memori yang terdapat pada perangkat di penelitian ini tidak mungkin melakukan perhitungan dengan nilai sebesar itu.

6. Kesimpulan

Dalam penelitian ini, metode *public-key cryptography* menggunakan algoritma *RSA* dapat diimplementasikan pada perangkat *IoT* seperti mikrokontroler. Hasil uji coba menunjukkan proses enkripsi menggunakan algoritma *RSA* dapat berjalan dengan baik walaupun memakan waktu dan memori yang tidak sedikit. Panjang data yang dienkripsi sebesar 18 *bytes*, sedangkan kunci yang digunakan enkripsi *RSA* sebesar 16 bit. Kunci ini masih jauh dari panjang kunci *RSA* yang dianggap aman yaitu sekitar 1024 bit ke atas, ini menunjukkan perangkat yang digunakan tidak mungkin melakukan perhitungan untuk kunci sebesar itu sekalipun dipaksakan pasti akan membuat kinerja dari mikrokontroler menjadi sangat lambat. Dilain sisi, perangkat tersebut harus selalu memberikan informasi yang *real-time* sehingga memori dan kinerja dari mikrokontroler memang perlu diefisienkan penggunaannya. Saran ke depan yaitu menggunakan metode enkripsi hybrid untuk kasus diatas. Metode enkripsi hybrid merupakan metode yang mengkombinasikan enkripsi simetris dan asimetris. Enkripsi dapat menjamin keamanan data pada perangkat *IoT*.

7. Daftar Pustaka

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] T. Li, "Principle, Framework and Application of Internet of Things," *Appl. Mech. Mater.*, vol. 303–306, pp. 2144–2148, 2013.

[3] L. Toms, "5 Common Cyber Attacks in the IoT - Threat Alert on a Grand Scale," *GlobalSign Blog*, 2016. [Online]. Available: https://www.globalsign.com/en/blog/five-common-cyber-attacks-in-the-iot. [Accessed: 03-Jul-2016].

[4] M. O'Neill, "Insecurity by Design: Today's IoT Device Security Problem," *Engineering*, vol. 2, no. 1, pp. 48–49, 2016.

[5] H. J. Highland, "Data encryption: A non-mathematical approach," *Comput. Secur.*, vol. 16, no. 5, pp. 369–386, 1997.

[6] R. Wilson, "A Clearing Picture of the Internet of Things," *Altera Corporation*, 2013. [Online]. Available: http://systemdesign.altera.com/a-clearing-picture-of-the-internet-of-things-2.

[7] D. Ariyus, *Pengantar Ilmu Kriptografi*. Yogyakarta: Penerbit Andi, 2008.

[8] D. Ireland, "RSA Algorithm," *DI Management*, 2011. [Online]. Available: http://www.di-mgt.com.au/rsa_alg.html. [Accessed: 25-Oct-2016].