Bab II
Tinjauan Pustaka

2.1 FOTA *Object*

Menurut Doddapaneni, FOTA *object* digunakan untuk membuat sebuah notasi yang umum dan *reusable* objek untuk melakukan FOTA *update* [2]. FOTA *object* berisikan informasi yang digunakan untuk melakukan FOTA *update*. Informasi-informasi inilah yang menjadi standard untuk membuat sebuah FOTA *packet*. Dengan FOTA *object* ini perangkat dapat mengetahui apa yang harus dilakukan terhadapat *packet* tersebut. Oleh karena itu, objek ini dapat digunakan untuk menjadi standard komunikasi antar perangkat dan *client*.

2.2 HMAC-SHA256

SHA256 adalah sebuah *secure hash algorithm* yang menghasilkan *digest message* dengan panjang yang tetap. SHA256 menghasilkan *digest* dengan panjang 256 bit atau sama dengan 32 byte. HMAC-SHA256 digunakan untuk membuat sebuah *message* yang lebih sulit untuk di rekonstruksi atau dibuat ulang. Algoritma tersebut menggunakan sebuah *secret key* yang ditambahkan di depan dan di belakang *plain text*. Setelah *secret key* ditambahkan maka *message* tersebut di *digest* menggunakan SHA256. Menurut Cheng, HMAC-SHA256 memiliki bit yang lebih panjang dibandingkan dengan HMAC-SHA1 sehingga algoritma tersebut dapat menghasilkan lebih banyak *message* unik [5]. HMAC biasa digunakan untuk menjaga integritas sebuah data dan menjaga konsistensinya. Algoritma tersebut umum digunakan untuk melakukan *signing* pada data. Dengan adanya *signing* tersebut data dapat dikirimkan kemana pun tanpa kehilangan integritas yang dimiliki oleh data tersebut. Sebagai contoh dalam *paper*-nya, Cheng menggunakan algoritma tersebut untuk men-*signing* XML.

2.3 Token Based Authentication

JSON Web Token (JWT) adalah sebuah standard untuk autentikasi berbasis token yang umum digunakan dalam *web technology*. Bhawiyuga menggunakan JWT untuk memverivikasi komunikasi MQTT dalam arsitektur IOT [6]. Dia menggunakan NodeMCU untuk implementasinya. Perangkat akan mengirimkan *credential* ke server. Jika *credential* tersebut benar, server akan membalas dengan memberikan sebuah JWT. JWT dibuat menggunakan HMAC-SHA256. Dia menggunakan Oauth untuk autentikasinya. Oauth adalah sebuah *service* yang menyediakan platform untuk JWT *authentication*. Setiap komunikasi dengan MQTT, perangkat dan server akan memverifikasi token ke Oauth yang membuat token tersebut

2.4 Cryptography for Low Power Device

Menurut Kaps, SHA-1 dan AES dapat memiliki *ultra-low power consumption* sekitar 30 uW [7]. Dalam grafik yang ditunjukan pada *paper*-nya, SHA-1 memiliki *power consumption* yang lebih rendah secara keseluruhan untuk bytes yang lebih tinggi dibandingkan AES. AES akan membutuhkan *power* yang lebih banyak ketika bytes meningkat. *Power consumption* AES cukup linear terhadap penggunaan bytes. Disisi lain, penggunaan *power* SHA-1 tidak setajam AES ketika bytes meningkat. Penggunaan *power* ini diukur dalam melakukan enkripsi dan autentikasi *message*

2.5 Penelitian Terkait

A Secure Vehicle ECUs Updates Over The Air

Daimi menggunakan sebuah metode untuk mengamankan FOTA *update*. Mereka menggunakan *firmware distribution authority* dan *certificate authority* [8]. Setiap komponen adalah arsitektuk FOTA akan mendapatkan *firmware distribution authority certificate* dari *firmware distribution authority*. *Firmware distribution authority* adalah sebuah bagian penting untuk mendistribusikan *firmware*. *Firmware* akan disimpan pada sebuah *firmware repository*. Ketika update *firmware* akan dilakukan, *firmware repository* akan menyediakan *firmware*

yang dibutuhkan ke *firmware distribution authority*. Mekanisme ini dapat menjaga integritas *update* tetapi disisi lain, mekanisme ini membutuhkan lebih banyak bagian dibandingkan mekanisme *client-server* yang umum digunakan.

B Simple Implementation of Secure FOTA Updates

Untuk implementasi sederhana pada FOTA *update*, terdapat implementasinya pada mikrokontroller ESP32. Microcontroller tersebut akan menyimpan *public key* dari server [9]. Mekanisme yang digunakan untuk autentikasi seperti SSL/TLS. Server akan mengirimkan *firmware* yang terenkripsi menggunakan *private key* dari server tersebut. Metode ini mudah dilakukan tetapi sulit di implementasikan pada sistem yang perangkatnya dapat berubah secara dinamis ataupun diganti pada *runtime*.

C Secure Firmware Validation and Update for Consumer Devices in Home Networking

Byung-Chul Choi memiliki sebuah metode untuk melakukan *secure FOTA update* menggunakan sebuah *server, manager,* dan *device*. Dia menggunakan *session key derivation*. *Server* memiliki *public key* dan *private key*. *Private key* dan *public key* tersebut digunakan untuk membuat sebuah session key untuk *manager* [10]. Dengan cara yang sama, *manager* menggunakan *session key* dari server untuk membuat *session key* untuk *device* menggunakan *public key* dan *private key manager*. *Manager* memecah *firmware* menggunakan *hash chaining*. Setiap bagian *firmware* dikirimkan ke perangkat dengan masing-masing bagian tersebut memiliki *session key* masing-masing. Cara ini dapat menjaga integritas dan ke rahasiaan data. Setiap koneksi memiliki *key* masing-masing untuk meng-autentikasi *request*. Hal tersebut memastikan *key* tersebut tidak dapat di pakai ulang. Cara ini dapat memvalidasi FOTA *request* tetapi untuk melakukannya diperlukan *bandwidth* dan kemampuan komputasi yang cukup besar. Dalam metode ini, terdapat banyak pertukaran *public key* dan *session key*. Untuk pertukarang *public key* dan membuat sebuah *session key* setidaknya dibutuhkan 3 *round trip time*.

D A Hardware-based Framework for Secure Firmware Updates on Embedded Systems

Solon Falas menggunakan *hardware base framework* untuk mengamankan *firmware updates* pada *embedded system*. Dia mengamankan komunikasi menggunakan PPUF (Public Physical Uncloneable Function) [11]. Vendor menaruh *pre-shared key* pada perangkat. Perangkan menggunakan *key* ini untuk menghasilkan sebuah *response* menggunakan *challenge* yang didapatkan dari vendor *firmware*. Jika *hash string* dari vendor *firmware* sama dengan *hash* dari *response* maka FOTA *update* akan dilakukan. Mekanisme ini dapat memastikan komunikasi perangkat ke vendor autentik. Metode ini juga akan memiliki waktu proses yang cepat karena autentikasi dilakukan menggunakan apa yang sudah dimiliki oleh perangkat yaitu PPUF. Meskipun metode ini dapat menjaga integritas *firmware* tetapi terdapat kekurangan yaitu membuat metode ini dapat bekerja dengan arsitektur dan perangkat IOT yang sudah ada. Arsitektur IOT bisa saja memiliki lebih dari satu vendor dan pihak yang melakukan FOTA bisa saja tidak cuma dari vendor tetapi juga pengguna. Mekanisme ini sangat terikat pada manufakturnya. Hal tersebut membuat metode ini sulit untuk di *scaling* jika *developer* menggunakan beberapa vendor.

E I Can Detect You: Using Intrusion Checkers to Resist Malicious Firmware Attacks

Devu Manikantan Shila membuat sebuah *intrusion checker* untuk mencegah *malicious firmware* yang bernama I Can Detect You [12]. Mereka mengecek sifat dari *firmware procedure call*. Pada tahap *offline*, mereka melatih sebuah model untuk mendeteksi kejanggalan kondisi pada *firmware*. Model tersebut akan mempelajari *firmware signature* yang normal dan digunakan untuk menentukan kondisi yang janggal. *Procedure signature* dapat diambil dari *time interval* dan panjang *stack*. *Time interval* adalah lama waktu *procedure* dilakukan. Panjang *stack* adalah urutan *function* pada *procedure*. Cara ini tidak akan membebani perangkat *embedded*. Hal tersebut dapat menambah kedalaman

keamanan tetapi akan sulit diimplementasikan jika perangkat ada dibalik jaringan yang sulit didefinisikan terutama jalurnya.

2.6 Kesimpulan

Bedasarkan tinjauan pustaka diatas, Untuk membuat *secure FOTA update* sebuah protokol yang memungkinkan perangkat dapat mengenali pengguna dan menerima *firmware* dari pengguna tersebut dengan aman. Seperti yang dilakukan Daimi, setiap bagian yang terlibat dalam verifikasi dan autentikasi FOTA *update* memiliki tugas yang spesifik tetapi harus tetap dapat berkomunikasi dan bekerja sama untuk melakukan autentikasi. Protokol ini perlu diketahui oleh semua pihak yang terlibat dalam FOTA *update*. Hal ini dapat membuat semua pihak berkerja dengan perilaku yang sudah ditentukan dan dapat di prediksi. Selain itu, sistem yang baru harus dapat mengakomodasi dan bekerja bersama dengan sistem yang sudah ada. Dengan ini dapat muncul sebuah masalah. Salah satu contohnya adalah sebuah sistem yang sudah berjalan bisa saja tidak *scaleable*. Sistem dibutuhkan untuk *scaleable* untuk mengakomodasi jika terjadi perubahan pada jumlah perangkat atau perilaku perangkat. Oleh karena itu, dibutuhkan satu pihak yang memantau sistem secara keseluruhan.

Berdasarkan survey yang dilakukan oleh Kaps, *Low power device* dapat menggunakan SHA sebagai algoritma untuk enkripsi dan autentikasi. SHA dapat mengakomodasi bytes yang meningkat tanpa menambah *power consumption*. Hal ini menunjukan dengan SHA, sistem dapat memiliki jumlah bytes yang cukup besar tanpa harus menambahkan *power*.

Bagian terakhir yang perlu dilakukan untuk mencapai *secure* FOTA *update* adalah pertukaran kunci. Ketika ingin melakukan komunikasi yang aman, setiap pihak yang saling berkomunikasi harus memiliki *key* yang digunakan untuk verifikasi tetapi ketika mengirimkan *key* tersebut, *key* tersebut akan melewati *untrusted media*. Seperti pada percobaan Byung-Chul Choi, agar dapat mengirimkan *key* dengan aman, dibutuhkan *public key* dan *private key* untuk mengenkripsi komunikasi.