# BABI PENDAHULUAN

## 1.1 Latar Belakang

*Internet Of Things* (IoT) menjadi salah satu teknologi yang ramai diperbincangkan saat ini. Kemajuannya yang pesat dan penerapannya yang bisa digunakan di semua bidang menjadikan IoT menjadi salah satu teknologi yang paling berkembang. Menurut hasil studi dari Gartner [1], perusahaan riset dan teknologi dari Amerika Serikat. Pada tahun 2017 ini akan ada 1,5 miliar perangkat baru yang terhubung ke internet. Jumlah tersebut akan meningkat hingga 20 miliar perangkat pada tahun 2020. Perangkat IoT sendiri dapat dibedakan menjadi 3 kategori, yaitu *Wearables*, Perangkat *Smart Home* dan Perangkat M2M *(Machine to Machine)*.

*Wearables* merupakan perangkat yang selalu dibawa oleh pengguna. Biasanya terhubung melalui koneksi *Bluetooth* ke perangkat seluler yang kemudian tersambung ke Internet. Perangkat di kategori ini termasuk jam pintar dan fitness band. Perangkat *Smart Home* merupakan perangkat yang bisa ditemukan di dalam rumah. Perangkat ini meliputi *motion sensor*, pintu dan saklar otomatis hingga oven. Kategori ketiga yaitu M2M merupakan perangkat yang langsung terhubung ke jaringan seperti mobil yang mampu memberitahu lokasinya saat terjadi kecelakaan, atau sebuah kulkas yang mampu memesan sendiri ketika stok buah yang disimpannya habis.

Dengan semakin banyaknya perangkat IoT yang terhubung secara online, pekerjaan manusia tentu akan terbantu. Namun dilain pihak, juga akan menimbulkan masalah baru ketika perangkat-perangkat yang terhubung tersebut memiliki tingkat keamanan yang rendah. Perangkat IoT yang dikuasai hacker dapat diubah menjadi botnet. Botnet ini mampu dikendalikan secara remote oleh hacker untuk melakukan serangan *Distributed Denial Of Service Attacks* (DDOS) ke jaringan tertentu. Pada akhir tahun 2016 kemarin, Dyn sebuah perusahaan provider *Domain Operated System* (DNS) mengalami serangan DDOS pada server mereka. Hal ini mengakibatkan situs-situs besar yang menggunakan layanan DSN Dyn seperti Amazon, Airbnb, CNN, Netflix dan Spotify tidak dapat diakses oleh pengguna. Setelah dilakukan investigasi menyeluruh, ditemukan bahwa serangan tersebut dilakukan lebih dari 100.000 perangkat IoT yang terkena malware Mirai botnet [2].

Berkaca dari hal diatas, sisi keamanan dari IoT perlu ditingkatkan agar kasus tersebut tidak terulang kembali. Baik dari sisi perangkat itu sendiri, komunikasi data ataupun dari sisi *Application Programming Interface* (API). API merupakan penghubung antara bagian *backend* dan *frontend*. Di pengembangan aplikasi berbasis IoT, API menjadi sistem penghubung penghubung antara sensor dengan database, ataupun database dengan antarmuka aplikasi. Penggunaan API diimplementasikan dalam bentuk web service. Generasi pertama web service yang diperkenalkan adalah Simple Object Acces Protocol (SOAP) namun karena perkembangan perangkat IoT yang semakin banyak dan SOAP tidak mampu menghandle perangkat yang berbeda standar maka penggunaan SOAP mulai ditinggalkan.

Representational State Transfer (REST) menjadi pengembangan selanjutnya dari web service. REST web service / RESTful memiliki keunggulan mampu menghandle perangkat-perangkat yang berbeda standar karena menggunakan basis Resource Oriented Architecture (ROA) [3]. REST sendiri memiliki 3 attribute yaitu :

1. Addresssability

Semua resource akan diimplementasikan menggunakan Uniform Resource Identifiers (URI). Setiap resource tersebut akan memiliki alamat URI sendiri. Ketika alamat URI dipanggil dia akan mengembalikan respon dalam bentuk JSON atau XML.

1. Connectedness

Resource yang ada dalam REST harus memiliki relasi dengan resource yang lain agar dapat dipresentasikan melalui URI.

1. Homogeneus Interface

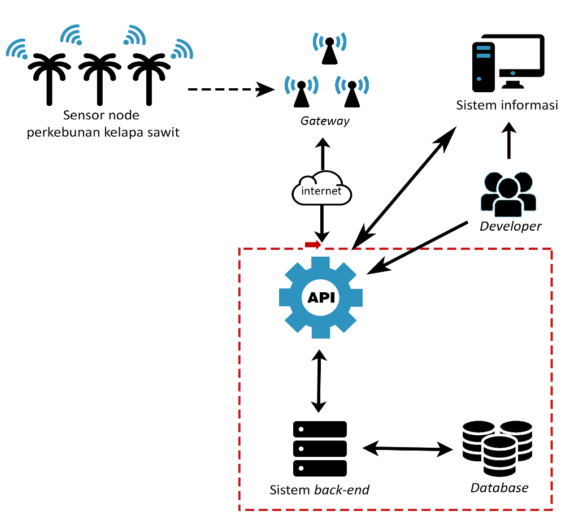
Resource akan dipanggil menggunakan 4 metode HTTP yaitu GET, PUT, POST dan DELETE dengan 2 tambahan metode yaitu HEAD dan OPTION. HEAD digunakan untuk menunjukkan Metadata sedang OPTION digunakan untuk memeriksa metode yang ada

1. Statelessness

Stateles menunjukkan bahwa server tidak menyimpan data dari klien dari setiap koneksi yang terbentuk.

Protokol REST ini banyak digunakan sebagai standar komunikasi data pada sistem-sistem IoT. Salah satu sistem IoT yang menggunakan RESTful adalah Smart Farm[4]. Pada Smart Farm, perangkat IoT digunakan sebagai alat monitoring pada perkebunan kelapa sawit yang dapat diakses melalui <http://smartcity.wg.ugm.ac.id/webapp/smartfarm/index.php> . Ilustrasi sistem ini dapat dilihat pada Gambar 1.1. Ada 4 bagian utama didalam sistem ini yaitu : Sensor Node, Gateway, *Back End* dan *Front End*. Sensor Node akan memonitoring kondisi lingkungan dan mengambil data berupa suhu udara, suhu tanah, kelembapan tanah, kelembapan udara, intensitas cahaya matahari, curah hujan, arah angin, kecepatan angin, ketinggian air dan kelembapan daun tanaman kelapa sawit. Data-data tersebut akan dikirimkan secara nirkabel melalui gateway yang akan disimpan kedalam Back End atau Database. Agar data dari database tersebut dapat divisualisasikan dengan baik maka perlu adanya layanan Front End atau Sistem Informasi. Untuk menjembatani antara Back End dan Front End maka diperlukan pengembangan API menggunakan protokol REST. API didalam sistem Smart Farm ini memiliki 2 fungsi yaitu sebagai jalur penghubung antara sensor node dengan database dan penghubung antara database dengan Sistem informasi.

Di dalam sistem Smart Farm ini, REST sudah dikembangkan secara baik namun masih memiliki beberapa kekurangan karena dalam pengembangan sistemnya aspek keamanan data belum menjadi prioritas. Sebagai contoh, belum adanya otentikasi user saat mengakses API tersebut, sehingga jika alamat URI dari API tersebut diketahui maka dapat diakses oleh semua orang. Masalah yang lain yaitu setiap user bisa melihat semua data dalam database. Baik itu data dari sensor maupun data yang bersifat private seperti username dan password user. Untuk itu diperlukan suatu layanan manajemen user.



Gambar 1.1 Konsep sistem Smart Farm pemantauan perkebunan Kelapa Sawit

## 1.2 Perumusan masalah

Berdasar latar belakang diatas maka maka dirumuskan masalah sebagai berikut :

1. Data API pada system Smart Farm masih bisa diakses oleh semua user
2. Belum ada pembagian user berdasarkan hak akses
3. Perlunya penambahan metode autentifikasi didalam protokol REST yang digunakan

## 1.3 Keaslian penelitian

Penelitian - penelitian di bidang Internet of Things yang menggunakan arsitektur metode komunikasi REST sudah banyak dipublikasikan. Di bidang kesehatan ada Health Monitoring Management (HMM) yang digunakan untuk memantau tanda-tanda vital pasien [5], Di bidang property, IoT sudah diterapkan pada Smart Home untuk melakukan kontrol dan pemantauan alat-alat rumah tangga [6] dan juga sebagai pemantau suhu dan temperature [7]. Di bidang pertanian, ada Smart Farm yang digunakan untuk memantau kondisi perkebunan kelapa sawit [4].

Namun dari beberapa penelitian yang menggunakan REST sebagai metode komunikasinya, sebagian besar masih fokus dalam pemanfaatan metode tersebut sebagai penghubung antara sensor dengan aktuator maupun sensor dengan database. Faktor keamanan pada penelitian-penelitian tersebut masih belum menjadi perhatian. Dalam Tabel 1.1 dibawah ini, beberapa penelitian yang berhubungan dengan keamanan penggunakan metode komunikasi REST antara lain:

Tabel 1.1 Tabel Keaslian Penelitian

|  |  |  |  |
| --- | --- | --- | --- |
| No | Judul | Pengarang | Metode Otentikasi |
| 1 | HTTP Authentication : Basic dan Digest Access Authentication (1999) | Lawrence et al[8] | Username dan Password |
| 2 | An Extended Username Token-based Approach for REST-style Web Service Security Authentication (2009) | Dunlu Peng et al[9] | Username dan Password + Hash |
| 3 | A token-based user authentication mechanism for data exchange in RESTful API (2015) | Xiang-Wen Huang et al [10] | Token |
| 4 | A Method for Secure RESTful Web Service (2015) | Sungchul Lee et al[3] | Token dan ID Based Encryption |
| 5 | Study on Access Permission Control for the Web of Things (2015) | Se Won Oh et al[11] | Manajemen Akses Kontrol berdasar Web of Thing |
| 6 | Security Analysis and proposal of new Access Control model in the Internet of Thing (2015) | Ouaddah et al [12] | Manajemen Akses Kontrol berdasar Organization Based Access Control (OrBAC) |

Dari Tabel 1.1 diketahui penggunaan username menjadi salah satu metode otentikasi pada REST. Namun dalam perkembangannya metode tersebut mulai ditinggalkan dan beralih ke penggunaan Token. Manajemen hak akses juga sudah mulai diperkenalkan dalam sebuah peneltian namun belum secara spesifik membahas implementasinya didalam aplikasi IoT.

Pada penelitian ini akan digunakan 2 metode untuk meningkatkan keamanan metode komunikasi REST yaitu penggunaan token sebagai otentikasi serta pembatasan hak akses URI berdasar dari peran user. Penerapan kedua metode tersebut dalam bidang IoT berserta tantangan-tantangannya akan menjadi bahasan utama dalam penelitian ini. Hasil dari penelitian ini diharapkan dapat memberikan kontribusi dalam bidang keamanan Internet of Things, serta memberi arah pengembangan aplikasi-aplikasi Internet of Things di masa mendatang.

## 1.4 Tujuan Penelitian

Penelitian ini bertujuan :

1. Mengembangkan metode otentikasi pada API Smart Farm
2. Membuat manajemen user berdasar peran user dalam sistem Smart Farm

## 1.5 Batasan Masalah

Penelitian ini memiliki batasan masalah sebagai berikut :

1. Difokuskan pada sistem API yang telah ada
2. Fokus pada pengembangan otentikasi dan layanan hak ases user dalam mengakses URI
3. API penghubung sensor node dengan database tidak menjadi perhatian
4. Hardware dan *sensor node* yang digunakan tidak menjadi perhatian dalam penelitian ini
5. Perancangan sistem informasi dan *user interface* tidak menjadi perhatian dalam penelitian ini

## 1.6 Manfaat Penelitian

Dengan penelitian ini, maka diharapkan akan memberikan manfaat antara lain

1. Sistem Smart Farm yang telah dibuat akan lebih aman
2. Sistem memiliki pembagian user, dan setiap user memiliki hak akses sendiri
3. Sistem keamanan API yang dibuat dapat dikembangkan lagi untuk penelitian selanjutnya

# BAB II TINJAUAN PUSTAKA DAN LANDASAN TEORI

## 2.1 Tinjauan Pustaka

Metode komunikasi Web Service sudah semakin banyak digunakan sekarang ini. Terutama dengan semakin banyaknya peralatan-peralatan yang berbasis IoT membuat penerapan Web Service menjadi semakin masif. Simple Object Access Protocol (SOAP) merupakan generasi pertama web service yang diperkenalkan. Namun karena perkembangan perangkat IoT yang semakin banyak dan SOAP tidak mampu menghandle perangkat yang berbeda standar maka penggunaan SOAP mulai ditinggalkan. Pengembangan web service selanjutnya yaitu Representational State Transfer (REST). REST ini memiliki keunggulan mampu menghandle perangkat-perangkat yang berbeda standar karena dalam komunikasinya menggunakan metode HTTP umum yaitu GET, PUT, POST, DELETE, HEAD dan OPTION.

Penerapan metode REST juga sudah banyak digunakan di penelitian - penelitian di bidang Internet of Things. Di bidang kesehatan ada Health Monitoring Management (HMM) yang digunakan untuk memantau tanda-tanda vital pasien [5], Di bidang property, IoT sudah diterapkan pada Smart Home untuk melakukan kontrol dan pemantauan alat-alat rumah tangga [6] dan juga sebagai pemantau suhu dan temperature [7]. Di bidang pertanian, ada Smart Farm yang digunakan untuk memantau kondisi perkebunan kelapa sawit [4]. Namun dari beberapa penelitian yang menggunakan REST sebagai metode komunikasinya, sebagian besar masih fokus dalam pemanfaatan metode tersebut sebagai penghubung antara sensor dengan aktuator maupun sensor dengan database. Faktor keamanan pada penelitian-penelitian tersebut masih belum menjadi perhatian.

Karena menggunakan dasar HTTP sebagai komunikasinya, maka metode REST memiliki kerentanan yang sama dengan beberapa aplikasi web yang lain. Menurut Femke [13], beberapa serangan yang bisa dilakukan terhadap komunikasi REST, yaitu :

1. SQL Injection

Metode komunikasi REST sangat bergantung pada komunikasi HTTP. Jika tidak ada mekanisme untuk melakukan validasi pada data yang masuk, maka sistem akan rentan terhadap serangan SQL Injection

1. Serangan Man-in-the-Middle (MITM)

Serangan yang terjadi ketika penyerang membuat sebuah koneksi baru kepada user, membuat user seolah-seolah sedang berkomunikasi dengan server yang asli.

1. Replay Attack

Serangan yang terjadi pada jaringan dimana penyerang mengambil informasi yang bersifat rahasia seperti otentifikasi , lalu penyerang menggunakan informasi tersebut untuk pura-pura menjadi klien yang ter-otentifikasi

1. Spoofing

Serangan dimana penyerang berpura-pura menjadi host yang dapat dipercaya pada suatu jaringan. Teknik ini dapat digunakan oleh penyerang untuk memalsukan data yang diminta oleh klien

1. Cross-Site Scripting (XSS) dan Cross-Site Request Forgery (CSRF)

XSS dan CSRF merupakan 2 serangan yang sama-sama ditujukan melalui browser kepada klien. Serangan ini mampu mencuri otentikasi dari klien atau memanipulasi konten yang dikirim dari server

REST merupakan metode komunikasi yang stateless dimana setiap request yang terjadi bersifat independen sehingga server harus melalukan otentikasi klien setiap kali request terjadi. Stateless juga berarti tidak ada session yang disimpan selama otentikasi dilakukan. Hal ini mengakibatkan otentikasi berdasar protokol HTTP menjadi tidak memadai. Beberapa penelitian mulai dilakukan untuk menemukan metode yang relevan untuk menghindari serangan-serangan terhadap komunikasi REST seperti diatas.

Penelitian pertama menggunakan metode HTTP Basic Authentication[8]. Metode ini menggunakan server HTTP untuk melakukan otentikasi terhadap Web Browser. Ketika klien melakukan request terhadap resource maka server akan meminta identitas dari klien tersebut. Identitas ini berupa username dan password. Ketika klien memberikan identitas yang sesuai maka server akan memberikan response berupa resource yang diminta. Namun metode ini memiliki kelemahan yaitu tidak ada enkripsi terhadap request yang dilakukan.

Metode HTTP Basic Authentication kemudian dikembangkan menjadi HTTP Digest Authentication. Proses otentikasi yang dilakukan sama dengan Basic Authentication namun mekanisme yang dilakukan lebih komplek. Ketika server meminta identitas dari klien, maka klien akan memberikan username dan password yang ditambah dengan hash string seperti MD5 [9]. Proses otentikasi pada HTTP Digest Authentication seperti berikut. Pertama klien akan melakukan request kepada server dan server akan memberikan nonce (kata acak) kepada klien. Kemudian klien akan menggabungkan username, pasword dan nonce tersebut untuk membuat hash. Hash tersebut akan dikirim kembali dari klien ke server. Oleh server, hash dari akan dibandingkan dengan hash yang dibuat sendiri oleh server berdasar username dan password klien. Jika hash tersebut memiliki nilai yang sama maka klien akan diberikan akses terhadap resource. Metode HTTP Digest Authentication ini lebih aman jika dibandingkan dengan HTTP Basic Authentication namun memiliki kelemahan karena penyerang dapat melakukan serangan MITM [3].

Pengembangan selanjutnya yaitu penggunaan token sebagai otentikasi. Metode ini diklaim lebih aman karena tidak menggunakan username dan password. Proses otentikasi pada metode ini yaitu sebagai berikut. Pertama, klien melakukan request kepada server dengan menggunakan username dan password. Server akan memberikan response berupa token. Token ini akan digunakan oleh klien setiap melakukan request resource kepada server. Token bersifat acak dan tidak berelasi apapun dengan data klien yang ada sehingga lebih aman.

Pengembangan metode ini dilanjutkan oleh Huang et al [10] dengan penambahan timestamp di token pada setiap request yang terjadi. Token yang ditambahkan timestamp disebut disposable token. Dengan penambahan timestamp, maka token yang digunakan hanya berlaku dalam waktu tertentu. Sehingga mengurangi terjadinya resiko serangan. Kelemahan dari metode ini yaitu klien dan server harus membuat disposable token setiap request baru sehingga akan membebani resource dari sistem. Metode ini tidak cocok diterapkan pada sistem IoT karena sebagian besar perangkat IoT memiliki spesifikasi yang rendah.

Metode lain dikembangkan oleh Lee et al [3]. Metode ini merupakan pengembangan token yang dipadukan dengan otentikasi berdasar Identity-Based Encryption yang dicetuskan oleh Boneh dan Franklin [14]. Pada metode ini terdapat 4 tahap yaitu Setup, Extract, Encrypt dan Decrypt. Semua otentikasi dan otorisasi klien akan dilakukan melalui public dan private key yang digenerate oleh Public Key Generator (PKG). Dengan metode ini maka server tidak memerlukan session ID atau username dan password dari klien. Namun metode ini masih sebatas konsep dan belum diimplementasikan dalam aplikasi IoT.

Pengembangan lain dari otentikasi klien dalam metode komunikasi REST yaitu otorisasi klien untuk melakukan request terhadap resource yang ada. Permasalahan keamanan dan privacy data menjadi fokus utama pada hal ini. Beberapa penelitian yang membahas tentang kontrol akses klien terhadap resource URI sudah diusulkan seperti WOT Access Control[11] dan SmartOrBAC [12]. WOT Access Control menerapkan sistem yang tersebar (Decentralized Access Control) dimana setiap request yang terjadi akan memberikan response permission resource yang boleh atau tidak diakses. Sedangkan SmartOrBAC merupakan kebalikannya dengan menerapkan sistem kontrol yang terpusat. SmartOrBAC membuat rule berisi list-list klien yang memiliki permission untuk melakukan request terhadap server. Metode tersebar dan terpusat ini masih menjadi perdebatan dikalangan peneliti dalam hal penerapan ke dalam sistem IoT sehingga masih diperlukan penelitian lebih lanjut.

## 2.2 Landasan Teori

IOT

REST,

JSON

ROA

JSON WEB TOKEN

Landasan teori dituliskan berdasar tinjauan pustaka, sebagai bentuk yang lebih spesifik sesuai dengan arah penelitian Anda. Landasan teori ini didapat dari paper maupun buku, yang mendasari metodologi penelitian yang dibahas di Bab III.

### 2.2.1 Tabel dan Gambar

Contoh dan template untuk tabel terlihat diTabel 2.1.

Tabel 2.1Contoh template untuk tabel

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Huruf | Lafal | Huruf | Lafal | Huruf | Lafal | Huruf | Lafal | Huruf | Lafal |
| A | Ha | n | Na | c | Ca | R | Ra | k | Ka |
| F | Da | t | Ta | s | Sa | W | Wa | l | La |
| P | Pa | d | Dha | j | Ja | Y | Ya | v | Nya |
| M | Ma | g | Ga | b | Ba | Q | Tha | z | Nga |

Tabel 2.1 adalah contoh *template* untuk tabel, dengan memanfaatkan *Caption* dan *Cross Reference*. Tabel jangan sampai terpotong ke halaman lain. Jika terpaksa terpotong maka harus dibuat judul tabel baru pada halaman berikutnya yang merupakan lanjutan halaman sebelumnya. Contoh *template* untuk gambar terlihat di Gambar 2.1.



Gambar 2.1. Kuliner - Yimmi Tomyam

Gambar 2.1 menunjukkan contoh *template* gambar. Hati-hati dalam menggunakan *cross reference*. Jika posisinya diubah maka harus update, atau tulisan nanti akan berantakan.

Gambar grafik dimungkinkan berwarna. Semua warna akan dipertahankan pada CDROM. Grafik jangan menggunakan pola titik-titik karena ada kemungkinan tidak dapat dicetak sesuai aslinya. Gunakan SOLID FILL dan warna yang kontras untuk tampilan di layar komputer, dan gunakan warna hitam-putih untuk hardcopy.

### 2.2.2 Persamaan dan Bagian *Listing* Program

Persamaan dapat menggunakan fungsi persamaan atau ditulis dalam text biasa. Ketentuannya seperti dinyatakan dengan Persamaan (2-0).

|  |  |
| --- | --- |
|  | (2-0) |

dengan

 = nilai piksel keluaran  
 = nilai piksel masukan  
 = nilai ambang

Dalam membuat persamaan seperti Persamaan (2-0) terpaksa dibuat dalam tabel 2 kolom tanpa *border*. Hal ini dimaksudkan agar ketika dilakukan Cross Reference tidak akan muncul persamaannya. Disamping itu ketika muncul nomor persamaan harus ditambahi sendiri dengan tanda kurung tutup.

Kadang-kadang diperlukan bagian program untuk diterangkan di dalam tesis. Dengan demikian perlu penulisan bagian program tersebut, yang disebut dengan *listing* program. Penampilan*listing* program ditaruh di Lampiran.

## 2.3 Hipotesis

Hipotesis bersifat dugaan hasil tentang penelitian yang Anda lakukan.

## 2.4 Keterangan Lain-lain (tidak dimasukkan dalam format penulisan)

Dokumen ini adalah template untuk penulisan tesis. Sebuah salinan elektronik yang dapat di-download dari situs web program studi S2 jurusan Teknik Elektro UGM. Jika ada hal-hal yang belum jelas, silakan hubungi pengelola program studi atau dosen pembimbing[1].

Informasi tentang makalah akhir penyerahan tersedia dari situs web S2 jurusan Teknik Elektro UGM atau Kantor pengelola program studi[2].