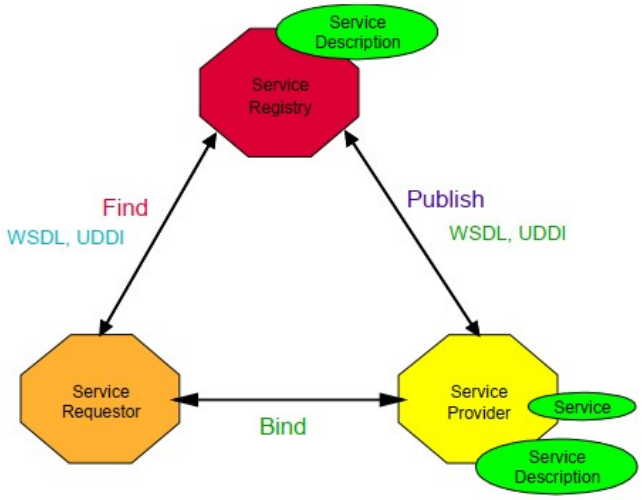
BAB II

LANDASAN TEORI

2.1 *Web Service*

Web service merupakan sebuah sistem perangkat lunak yang dibuat untuk mendukung interoperabilitas dan komunikasi antar sistem pada sebuah jaringan. *Web service* digunakan sebagai suatu fasilitas yang disediakan oleh suatu *website* untuk menyediakan layanan-layanan (dalam bentuk informasi) kepada sistem lain, sehingga sistem lain dapat berinteraksi dengan sistem tersebut melalui layanan-layanan (*service*) yang disediakan oleh sistem yang menggunakan *web service*. *Web service* juga dapat didefinisikan sebagai suatu antar muka yang menggambarkan sekumpulan operasi yang bisa diakses melalui sebuah jaringan dalam bentuk *XML* (Kreger, 2001). Contoh implementasi dari *web service* antara lain adalah *SOAP* dan *REST*. Arsitektur *web service* dapat dilihat pada gambar 2.1.



Gambar 2.1 Arsitektur *Web Service* (Kreger, 2001).

2.2 *RESTful*

Konsep *REST* pertama kali diperkenalkan oleh Roy Fielding pada tahun 2000 (Fielding, 2000). Gagasan utama dari *REST* adalah konsep *resource* sebagai komponen dari aplikasi yang perlu dialamatkan. *REST WS* membangun integrasi dengan cara yang lebih ringan dan sederhana, dan berfokus pada sumberdaya. *REST* dapat dijelaskan dalam lima batasan, diantaranya:

1. *Resource Identification*: *Web* bergantung pada *Uniform Resource Identifier* (*URI*) untuk mengidentifikasi sumber daya, sehingga *link* kesumber daya dapat dibentuk menggunakan skema identifikasi yang mudah untuk dikenali.
2. *Connectedness*: artinya klien dari *RESTFul Service* seharusnya mengikuti *link* untuk menemukan sumber daya agar dapat berinteraksi dengan *service*.
3. *Uniform Interface*: artinya sumber daya harus tersedia melalui antar muka yang seragam dengan semantik yang mendefinisikan interaksi, seperti *Hypertext Transfer Protocol* (*HTTP*). *HTTP* mencakup metode *POST*, *GET*, *PUT* dan *DELETE*.
4. *Self-Describing Messages*: artinya mengekspos *resource* yang ada, *RESTFul* menggunakan lebih dari satu format data (*XML*, *JSON*, *RDF*, dll) dibandingkan dengan *SOAP* (*XML*), namun hal ini tergantung *developer*.
5. *Stateless Interactions*: mengharuskan setiap *request* dari klien lengkap, dalam arti bahwa semua informasi untuk melayani *request* ke *server* harus berisi setiap informasi yang dibutuhkan agar *request*

dapat dipahami , dan tidak ada ketergantungan dengan *state* atau penanda dari *client*.

Terdapat dua bagian pesan yang digunakan untuk membangun komunikasi dengan *server* yaitu pesan *Header* dan pesan *Body*. *HTTP Header*, yang umum meliputi *header request*, *header response*, dan terdapat bidang entitas-*header*. *Request* sumberdaya dari masing-masing *client* dapat dikendalikan dengan memanfaatkan *HTTP Header*. Kolom *Header* terdiri dari nama diikuti dengan titik dua (":") atau *white space* dan konten *field*. Nama *field* bersifat *case-sensitive*. *Header* berisikan semua informasi yang diperlukan untuk mengumpulkan metode *request* dan respon (www.w3.org, 2018).

HTTP body mencakup pesan *HTTP* yang digunakan untuk memuat entitas *body* melalui protokol *HTTP* yang berhubungan dengan *request*. *Client* biasanya melakukan *request* menggunakan *HTTP body* yang berisikan informasi setiap parameter yang untuk *request*. Respons yang muncul berisikan informasi yang didapatkan dari hasil *request*.

REST menentukan sekumpulan prinsip arsitektur yang mana dapat digunakan untuk merancang *WS* yang berfokus pada sumber daya sistem, termasuk bagaimana sumber daya yang dialamatkan dan ditransfer melalui *HTTP* oleh berbagai klien yang ditulis dalam bahasa pemrograman yang berbeda (Rodriquez, 2008). *REST* dapat mengoperasikan operasi *CRUD* (*create, read, update dan delete*) yang dapat dilakuan dengan memanfaatkan metode *HTTP* antara lain *POST, GET*, *PUT* dan *DELETE*.

Format application/x-w ww-form-URLencoded yang digunakan oleh masing-masing metode *HTTP* diantaranya *GET* dan *DELETE* berbeda dengan *POST* dan *PUT* adalah berbeda, hal ini dikarenakan cara *parsing data* yang berbeda. *Parsing*

data pada metode *GET* dan *DELETE* dimuai melalui *URL*, sedangkan *POST* dan *PUT* melakukan *parsing data* melalui *payload HTTP* dengan memanfaatkan *media type 'application/x-www-form-UR Lencoded'*.

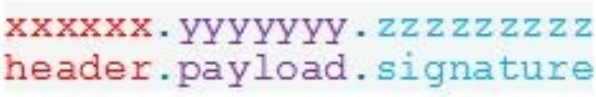
Kode status *HTTP* respon server terhadap aksi yang dilakukan oleh *client:*

- Kode status 201: *request* telah terpenuhi dan menghasilkan sumber daya yang baru *Create*.
- Kode status 200: respon standar untuk request *HTTP* dari *client* yang dinyatakan sukses oleh *server*. Respons sebenarnya akan tergantung pada metode *request* yang digunakan.

2.3 *JWT (JSON Web Token)*

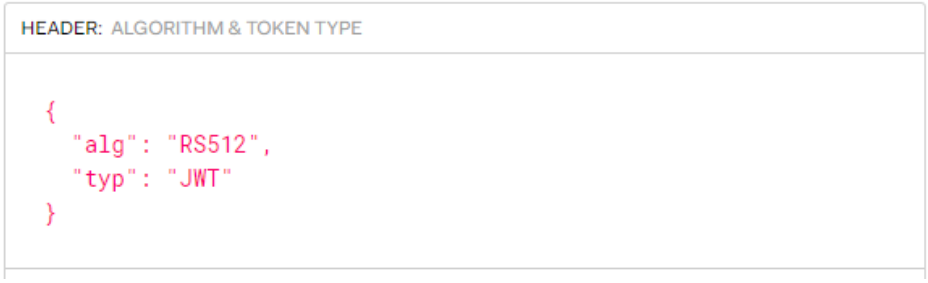
JWT ini adalah sebuah *token* berbentuk string *JSON* yang sangat padat (ukurannya), informasi mandiri yang gunanya sendiri untuk melakukan sistem autentikasi dan pertukaran informasi. Bentuk token *JWT* kecil sehingga dapat dikirim melalui *URL*, parameter *HTTP POST* atau di dalam *Header HTTP* dan juga karena ukurannya yang kecil maka dapat ditransmisikan dengan lebih cepat. *JWT* disebut sebagai informasi mandiri karena isi dari *token*nya mengandung informasi dari user yang dibutuhkan, sehingga tidak perlu *query* ke *database* lebih dari 1 kali. *Token* tersebut dapat diverifikasi dan dipercaya karena sudah di-*sign* secara digital. *Token JWT* bisa di-*sign* dengan menggunakan *secret* (algoritma *HMAC*) atau pasangan *public* / *private key* (algoritma *RSA*). Umumnya untuk melakukan login tidak seperti pada aplikasi website biasa dimana *user* menggunakan *session* untuk mengingat siapa yang sedang *login*. *API* yang *user* bangun biasanya menggunakan konsep *JWT* atau dibacanya sebagai "jot" (Jwt.io, 2018). *JWT* tidak tergantung dengan bahasa program tertentu.

Struktur *JWT* terdiri dari 3 bagian yang dipisahkan oleh titik (".") yaitu *header, payload dan signature*.



Gambar 2.2 Struktur *JSON Web Token* (Jwt.io, 2018).

Header biasanya terdiri dari 2 bagian : tipe *token* yaitu *JWT* dan algoritma hashing yang akan digunakan, seperti *RSA-512* atau lainnya. Contoh *header* dapat dilihat pada Gambar 2.3.



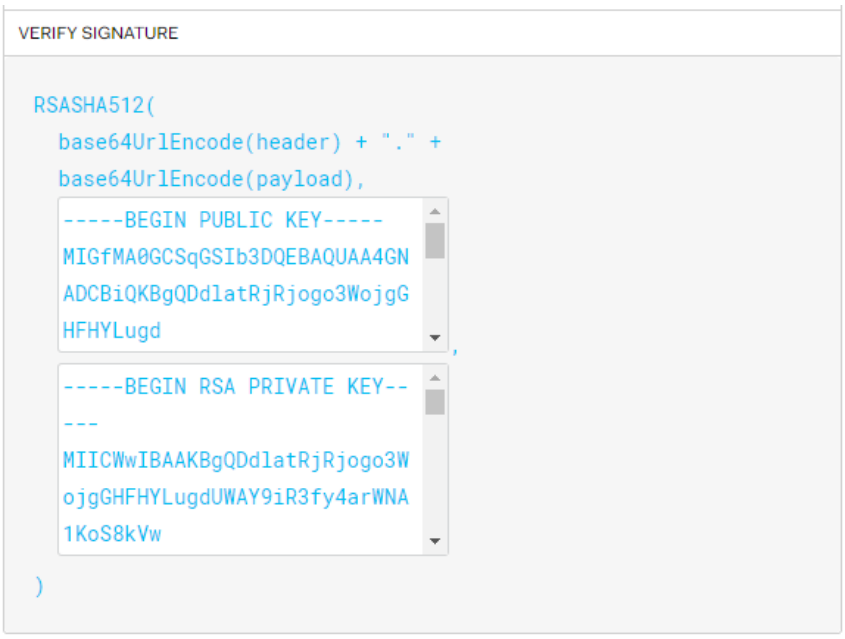
Gambar 2.3 *JWT Header* (Jwt.io, 2018).

Payload, bagian kedua berisi klaim. *Klaim* adalah pernyataan tentang suatu entitas (biasanya, pengguna) dan metadata tambahan. Ada 3 jenis *klaim*: *reserved*, *public* dan *private claims*. Bagian kedua (*payload*) dapat dilihat pada Gambar 2.4.

Gambar 2.4 *JWT Payload* (Jwt.io, 2018).

Bagian ketiga dari *JWT* adalah *signature* berisi *hash* dari komponen-komponen *header*, *payload* dan kunci rahasia. Contoh *JWT Signature* ini

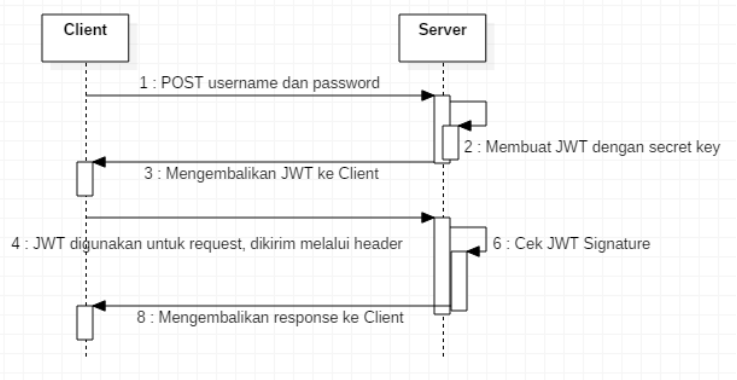
menggunakan algoritma *RSA-512*, *signature* dapat dibuat dengan cara seperti pada Gambar 2.5.



Gambar 2.5 *JWT Signature* (Jwt.io, 2018).

Signature dibentuk dengan menggunakan *header* dan *payload* sehingga *JWT* mampu memberikan kemudahan bagi *client* untuk mengakses sumberdaya tanpa harus login berulang memasukan *username password*. *Token* dapat dipanggil melalui *AJAX* ke *server* karena panggilan dapat menggunakan *header HTTP* untuk mengirimkan informasi penggunanya.

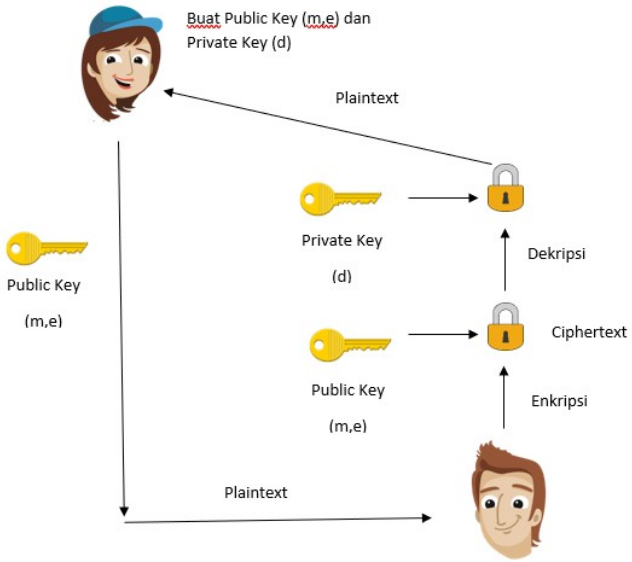
Mekanisme kerja dari *JWT* secara umum dapat dilihat pada Gambar 2.6.



Gambar 2.6 Mekanisme *JWT*.

2.4 *RSA (Secure Hash Algoritma)*

Algoritma *RSA* adalah sebuah algoritma berdasarkan skema kriptografi kunci *public*. Nama *RSA* sebagai diambil dari inisial nama para penemunya yaitu Ron Rivest, Adi Shamir, dan Leonard Adleman. *RSA* dibuat di *MIT* pada tahun 1977 dan dipatenkan oleh *MIT* pada tahun 1983. Paten tersebut berakhir tahun 2000, sehingga saat ini semua orang dapat menggunakannya dengan bebas. Algoritma *RSA* adalah algoritma yang mudah untuk diimplementasikan dan dimengerti (Rivest,1997).



Gambar 2.7 Diagram Algoritma RSA (Rivest,1997)

2.5.1 Tahap – Tahap Algoritma *RSA*

A. Pembangkitan Kunci pada Algoritma *RSA*

Misalkan Arifin ingin mengirim sebuah pesan melalui jalur yang aman kepada Sofia. Sofia akan memberikan kunci *public*nya kepada Arifin sedangkan kunci *private* akan disimpan untuk dirinya sendiri.

a. Pilih 2 bilangan prima seperti p, q dimana p tidak sama dengan q.

b. Hitung m = p * q

c. Hitung n = (p-1) * (q-1)

d. Pilih **e** yang relatif prima terhadap n.

e. Hitung nilai **d** sehingga memenuhi (e*d) mod n = 1.

f. (**e**, **m)** adalah kunci *public* yang diberikan untuk pihak lain yang ingin berkomunikasi (pihak yang akan mengirim pesan) untuk keperluan enkripsi.

g. (**d, m)** adalah kunci *private* yang harus dipegang sendiri oleh pihak yang akan menerima pesan untuk keperluan dekripsi.

B. Enkripsi

Misalkan Arifin ingin mengirim sebuah pesan X kepada Sofia.

a. Arifin akan menerima atau mendapatkan kunci *public* dari Sofia yaitu (e, m). Arifin menggunakan kunci *public* (e, m) untuk mengenkripsi pesan X.

C = X^e (mod m)

C adalah ciphertext hasil enkripsi plaintext X

b. Arifin akan mengirimkan C tersebut kepada Sofia

C. Dekripsi

Ciphertext yang diterima Sofia kemudian didekripsi, Sofia akan menggunakan kunci *private* (d, m) untuk mendekripsi

ciphertext tersebut agar dapat diketahui *plaintext*-nya. Tahapan-tahapan yang digunakan adalah sebagai berikut:

a. N = C^d (mod m)

b. N adalah plaintext yang ditemukan setelah mendekripsi C dengan kunci *private* d dan kunci *public* m.

2.5 *Postman*

Postman adalah *software* yang berfungsi untuk melakukan uji coba *API* (*Application Programing Interface*). *Postman* memudahkan pengujian terhadap *API*, karena *postman* merupakan aplikasi berbasi *GUI* (*Graphical User Interface*) atau memeliki tampian antarmuka pada tampilan postman terdapat beberapa kolom yang digunakan untuk melakukan *request* URL dari *API*, metode HTTP lengkap, *header*, *body*, dan lain-lain. *Request* cukup hanya mengisi parameter-parameter yang dibutuhkan dan dikirim ke *API* yang akan diuji. Kelebihan postman adalah dapat menyimpan konfigurasi yang telah dibuat sebelumnya ke direktori koleksi sehingga dapat digugunakan kembali. *Postman* memiliki banyak fitur dan dapat membantu produktivitas dalam pekerjaan dan membuat pengembangan *API* menjadi cepat dan mudah.

2.6 Penelitian Terkait

Penelitian terkait yang relevan dengan permasalahan pada penelitian ini adalah :

a. Vibha Kumari dalam jurnal yang diterbitkan pada tahun 2015 mengenai *Web Service Protocol: SOAP vs REST*. Penelitian tersebut menyimpulkan bahwa *SOAP* dan *RESTful* menempati posisi teratas dalam penggunaaan

Web Service Protocol. Penelitian ini melakukan perbandingan antara kedua *Web Service* tersebut. Berdasarkan perbandingan yang dilakukan, *RESTful* dinilai memiliki performa yang baik, namun *REST* memiliki kekurangan dari sisi keamanan. Penelitian kali ini akan diterapkan *JWT* dengan algoritma *RSA-512* pada arsitektur *RESTful WS* tersebut.

b. Mukhammad Agus Arianto, Sirojul Munir dan Khusnul Khotimah dari STT Terpadu Nurul Fikri dalam jurnal yang diterbitkan pada tahun 2016 mengenai Analisis dan Perancangan *Representational State Transfer* (*REST*) *Web Service* Sistem Informasi Akademik STT Terpadu Nurul Fikri Menggunakan Yii *Framework*. Penelitian tersebut menghasilkan sebuah sistem akademik STT Terpadu Nurul Fikri berbasis teknologi *REST*, dan *REST Web Service* ini dapat berjalan dengan baik. Penerapan *web service* ini belum menerapkan pengamanan pertukaran data. Penelitian kali ini akan diterapkan pengamanan data menggunakan *JSON Web Token* dengan algoritma *RSA*-512 pada *REST WS*.

c. Penidas Fiodinggo Tanaem, Danny Manongga, dan Ade Iriani dari Universitas Kristen Satya Wacana Salatiga dalam jurnal yang diterbitkan pada tahun 2016 mengenai *RESTFul Web Service* untuk Sistem Pencatatan Transaksi Studi Kasus PT. XYZ. Penelitian tersebut menghasilkan sebuah arsitektur *RESTful WS* yang aman bagi PT.XYZ. *RESTful WS* yang dibangun, menggunakan *JSON Web Token HMAC* dalam mengamankan komunikasi yang terjadi. Penelitian kali ini akan diimplementasikan *JWT* dengan algoritma *RSA*-512.

d. Iwan Iskandar dari UIN Sultan Syarif Kasim Riau dalam jurnal yang diterbitkan pada tahun 2015 mengenai Penerapan Teknologi *REST Service*

untuk Integrasi Data Mustahiq Berbasis *Service Oriented Architecture* (*SOA*). Penelitian tersebut menyimpulkan bahwa penerapan *SOA* dapat membantu dalam hal integrasi data mustahiq. Penerapan *SOA* ini belum disertai pengamanan pertukaran data. Penelitian kali ini akan diterapkan sistem pengamanan data menggunakan *JWT* dengan algoritma *RSA*-512.

e. Andi Resta Pradika dari Universitas Dian Nuswantoro dalam jurnal yang diterbitkan pada tahun 2016 mengenai Implementasi *Web Service* untuk Sinkronisasi Data Transkrip Nilai Mahasiswa di Unit Tata Usaha Fakultas Universitas Dian Nuswantoro. Penelitian tersebut menghasilkan sinkronasi data transkrip yang dapat dilakukan secara *realtime*. Sistem ini belum menerapkan keamanan pada sistemnya. Penelitian kali ini akan diterapkan mekanisme autentikasi menggunakan *JWT* dengan *RSA*-512.

f. Kevin Fidelis, Adi Wibowo, Justinus Andjarwirawan dari Universitas Kristen Petra dalam jurnalnya yang diterbitkan pada tahun 2016 mengenai Pembuatan *Web Application* untuk Mendukung Pelayanan Asisten Tutor di Universitas Kristen Petra. Penelitian tersebut menghasilkan aplikasi *web* yang sangat responsif yang dapat digunakan baik pada *computer, tablet*, dan *smartphone*. Penelitian ini juga telah menerapkan pengamanan pertukaran data dengan *JWT SHA-256*. Penelitian kali ini akan diterapkan pengamanan *JWT* dengan algoritma *RSA*-512.

g. Andri Warda Pratama, Adhitya Bhawiyuga, dan Mahendra Data dari Universitas Brawijaya dalam jurnalnya pada tahun 2018 mengenai Implementasi Autentikasi *JSON Web Token* (*JWT*) sebagai Mekanisme Autentikasi Protokol *MQTT* pada Perangkat *NodeMCU*. Penelitian ini menghasilkan bahwa *JWT* dengan algoritma *SHA*-256 dapat digunakan

dalam mekanisme autentikasi. Penelitian kali ini akan diimplementasikan *JWT* dengan algoritma *asymmetric RSA*-512.

h. Alam Rahmatulloh, Heni Sulastri dan Rizal Nugroho dari Universitas Siliwangi dalam jurnalnya pada tahun 2018 mengenai Keamanan *RESTful Web Service* Menggunakan *JSON Web Token (JWT) HMAC SHA*-512. Penelitian tersebut menghasilkan bahwa *JWT* dengan algoritma *HMAC SHA*-512 memiliki performa yang lebih baik dari *HMAC SHA*-256. Penelitian kali ini akan diimplementasikan *JWT* dengan menggunakan algoritma *asymmetric RSA*-512.

i. Daisuke Oku, Masao Yanagisawa, dan Nozomu Togawa dalam jurnalnya pada tahun 2018 mengenai *Scan-based Side-channel Attacks againts HMAC-SHA-256 Circuit Based on Isolating Bit-transition Group Using Scan Signatures*. Penelitian ini menghasilkan serangan *Scan-based Side-channel* yang telah berhasil mengambil *secret key* pada *HMAC-SHA*-256 dalam waktu yang relatif singkat. Serangan ini tentunya menjadi ancaman tersendiri pada *HMAC-SHA*-512 yang menggunakan *symmetric key* (*secret key*). Penelitian kali ini akan dilakukan implementasi mekanisme autentikasi menggunakan *JWT* dengan algoritma *asymmetric RSA*-512.

j. Ritu Tripathi, dan Sanjay Agrawal dalam jurnalnya pada tahun 2014 mengenai *Comparative Study of Symmetric and Asymmetric Cryptography Techniques*. Penelitian ini menghasilkan bahwa secara keseluruhan metode enkripsi *asymmetric* memiliki tingkat keamanan yang lebih baik, algoritma *RSA* lebih unggul dibanding algoritma enkripsi *asymmetric* lain. Penelitian kali ini akan mengimplementasikan algoritma *asymmetric RSA*-512 pada *JWT* sebagai alternatif terhadap penggunaan *symmetric key* pada *JWT*.

k. Yogesh Kumar, Rajiv Munjal, dan Harsh Sharma dalam jurnalnya pada tahun 2011 mengenai *Comparison of Symmetric and Asymmetric Cryptography with Existing Vulnerabilities and Countermeasures.* Penelitian ini membandingkan antara metode enkripsi *symmetric* dan *asymmetric*. Algoritma yang dipakai pada enkripsi *symmetric* yaitu *DES*, dan algoritma pada enkripsi *asymmetric* memakai *RSA*. Hasil Penelitian menunjukan bahwa algoritma *RSA* yang menggunakan *asymmetric key* memiliki tingkat keamanan yang lebih tinggi dibandingkan *DES* dengan *symmetric key*. Penelitian kali ini akan mengimplementasikan algoritma *RSA*-512 pada *JWT* sebagai alternatif sistem keamanan pada *RESTful*.

l. S. Nithya, E. George Dharma Prakash Raj dalam jurnalnya pada tahun 2014 mengenai *Survey on Asymmetric Key Cryptography Algorithms*. Penelitian ini menghasilkan bahwa kombinasi *private key* dan *public key* pada *asymmetric key* dapat meningkatkan keamanan pada sebuah data yang dienkripsi. *Asymmetric key* merupakan sebuah pondasi dalam keamanan pertukaran data dalam sebuah jaringan. Penelitian kali ini akan diimplementasikan *JWT* dengan algoritma *asymmetric RSA*-512 pada *RESTful*.

m. Priteshkumar Prajapati, Nehal Patel, Robinson Macwan, Nisarg Kachhiya, Parth Shah dalam jurnalnya yang terbit pada tahun 2014 mengenai *Comparative Analysis of DES, AES, RSA Encryption Algorithms*. Hasil penelitian ini menunjukan bahwa *RSA* unggul pada penggunaan memori pada saat proses enkripsi dan dekripsi. Penelitian kali ini akan dilakukan implementasi algoritma *asymmetric RSA*-512 pada *JWT*.

n. Sourabh Chandra, Smita Paira, Sk Safikul Alam, dan Goutam Sanyal dalam jurnalnya pada tahun 2014 mengenai *A Comparative Survey of Symmetric and Asymmetric Key Cryptography*. Penelitian ini menghasilkan bahwa *asymmetric key* lebih unggul jika dilihat dari tingkat keamanan, sedangkan *symmetric* memiliki keunggulan dalam kemudahan implementasi. *Asymmetric key* lebih optimal ketika diimplementasikan bersama *digital signature*. *Digital Signature* yang diberikan meningkatkan kerahasiaan dan keabsahan data sehingga salah satu pihak tidak dapat menyangkal data yang dienkripsi. Penelitian kali ini akan mengimplementasikan algoritma *asymmetric RSA*-512 dengan penggunaan *digital signature* pada *JWT*.

o. Andysah Putera Utama Siahaan, Elviwani, dan Boni Oktaviana dalam jurnalnya pada tahun 2018 mengenai *Comparative Analysis of RSA and ElGamal Cryptographic Public-key Algorithms*. Hasil dari penelitian ini menunjukan bahwa dari kedua algoritma *asymmetric key* yang dibandingkan, algoritma *RSA* memiliki proses waktu enkripsi dan dekripsi yang lebih cepat dari algoritma *ElGamal*. *RSA* memiliki tingkat keamanan yang tinggi, sehingga algoritma ini disarankan dalam penggunaan keamanan data. Penelitian kali ini akan diimplementasikan algoritma *RSA*-512 pada *JWT* sebagai alternatif sistem keamanan pada *RESTful*.

Tabel 2.1 Penelitian Terdahulu

No.	Penulis	Judul	State of The Art
1	Vibha Kumari 2015	*Web Service Protocol*: *SOAP* vs *REST*	1. *REST* dinilai lebih baik dibanding SOAP

2	Mukhammad Agus Arianto, Sirojul Munir dan Khusnul Khotimah 2016	Analisis dan Perancangan Representational *State Transfer* (*REST*) *Web Service* Sistem Informasi Akademik STT Terpadu Nurul Fikri Menggunakan *Yii Framework*	1.Implementasi *REST* memenuhi kebutuhan sistem informasi yang dibangun
3	Iwan Iskandar 2015	Penerapan Teknologi *REST Service* untuk Integrasi Data Mustahiq Berbasis *Service Oriented Architecture*	1.Implementasi *REST* mendukung integrasi data antar platform yang berbeda
4	Andi Resta Pradika 2016	Implementasi *Web Service* untuk Sinkronisasi Data Transkrip Nilai Mahasiswa di Unit Tata Usaha Fakultas Universitas Dian Nuswantoro	1.Implementasi *REST* mendukung sikronisasi data pada sistem informasi

Lanjutan Tabel 2.1 Penelitian Terdahulu

5	Kevin Fidelis, Adi Wibowo, Justinus Andjarwirawan 2016	Pembuatan *Web Application* untuk Mendukung Pelayanan Asisten Tutor di Universitas Kristen Petra	1.Implementasi *JWT* sebagai keamanan web
6	Penidas Fiodinggo Tanaem, Danny Manongga, dan Ade Iriani 2016	*RESTFul Web Service* Untuk Sistem Pencatatan Transaksi Studi Kasus PT. XYZ	1.*JWT* dapat digunakan sebagai salah satu sistem keamanan pada *REST* 2.Implementasi *SHA*-256 pada *JWT*
7	Andri Warda Pratama, Adhitya Bhawiyuga, dan Mahendra Data 2018	Implementasi Autentikasi *JSON Web Token* (*JWT*) sebagai Mekanisme Autentikasi Protokol *MQTT* pada Perangkat *NodeMCU*	1.Implementasi *JWT* dengan algoritma *symmetric SHA-256* sebagai mekanisme autentikasi
8	Alam Rahmatulloh, Heni Sulastri dan Rizal Nugroho 2018	Keamanan *RESTful Web Service* Menggunakan *JSON Web Token (JWT) HMAC SHA*-512	1.Algoritma *SHA*-512 lebih baik dari *SHA*-256
9	Daisuke Oku, Masao Yanagisawa, dan Nozomu Togawa 2018	*Scan-based Side-channel Attacks againts HMAC-SHA-256 Circuit Based on Isolating Bit-transition Group Using Scan Signatures*	1. Serangan *scan-based side-channel* dapat menyerang *secret key* pada algoritma *symmetric SHA*
10	Ritu Tripathi, dan Sanjay Agrawal 2014	*Comparative Study of Symmetric and Asymmetric Cryptography Techniques*	1.Enkripsi *asymmetric* lebih baik dari enkripsi *symmetric* 2.*RSA* unggul dibanding enkripsi *asymmmetric* lain

Lanjutan Tabel 2.1 Penelitian Terdahulu

11	Yogesh Kumar, Rajiv	*Comparison of Symmetric and*	1.Enkripsi *asymmetric RSA*

	Munjal, dan Harsh Sharma 2011	*Asymmetric Cryptography with Existing Vulnerabilities and Countermeasures*	lebih unggul dibanding enkripsi *symmetric DES*
12	S. Nithya, E. George Dharma Prakash Raj 2014	*Survey on Asymmetric Key Cryptography Algorithms*	1.Penggunaan *private key* dan *public key* meningkatkan keamanan enkripsi
13	Priteshkumar Prajapati, Nehal Patel, Robinson Macwan, Nisarg Kachhiya, Parth Shah 2014	*Comparative Analysis of DES, AES, RSA Encryption Algorithms*	1.*RSA* lebih unggul pada ukuran memori saat enkripsi dan dekripsi
14	Sourabh Chandra, Smita Paira, Sk Safikul Alam, dan Goutam Sanyal 2014	*A Comparative Survey of Symmetric and Asymmetric Key Cryptography*	1.*Symmetric key* lebih mudah digunakan 2.*Asymmetric key* lebih unggul dalam keamanan 3.*Digital signature* menambah kerahasiaan dan keabsahan data
15	Andysah Putera Utama Siahaan, Elviwani, dan Boni Oktaviana 2018	*Comparative Analysis of RSA and ElGamal Cryptographic Public-key Algorithms*	1.*RSA* lebih cepat dibanding *ElGamal* 2.*RSA* memilliki tingkat keamanan yang tinggi

Tabel 2.2 Penelitian Terdekat

No.	Penulis	Judul	State of The Art
1	Alam Rahmatulloh,	Keamanan *RESTful Web Service*	1.Implementasi *JWT*

	Heni Sulastri dan Rizal Nugroho 2018	Menggunakan *JSON Web Token (JWT) HMAC SHA*-512	dengan Algoritma *symmetric SHA*-512 2.Algoritma *SHA*-512 lebih baik dari *SHA*-256
2	Muhammad Nur Arifin 2018	Implementasi Mekanisme Autentikasi *RESTful WS* Menggunakan *JSON Web Token* dengan Algoritma *Asymmetric RSA*-512	1.Implementasi *JWT* dengan Algoritma *asymmetric RSA*-512

Tabel 2.3 Matriks Ruang Lingkup Penelitian

No.	Jurnal	Lingkup Penelitian				Penulis
1	*Web Service Protocol*: *SOAP* vs *REST*	√	-	-	-	Vibha Kumari 2015
2	Analisis dan Perancangan Representational *State Transfer* (*REST*) *Web Service* Sistem Informasi Akademik STT Terpadu Nurul Fikri Menggunakan *Yii Framework*	√	-	-	-	Mukhammad Agus Arianto, Sirojul Munir dan Khusnul Khotimah 2016

Lanjutan Tabel 2.3 Matriks Ruang Lingkup Penelitian

3	Penerapan Teknologi *REST Service* untuk Integrasi Data Mustahiq Berbasis *Service Oriented Architecture*	√	-	-	-	Iwan Iskandar 2015
4	Implementasi *Web Service* untuk Sinkronisasi Data Transkrip Nilai	√	-	-	-	Andi Resta

	Mahasiswa di Unit Tata Usaha Fakultas Universitas Dian Nuswantoro					Pradika 2016
5	Pembuatan *Web Application* untuk Mendukung Pelayanan Asisten Tutor di Universitas Kristen Petra	√	√	-	-	Kevin Fidelis, Adi Wibowo, Justinus Andjarwirawan 2016
6	*RESTFul Web Service* Untuk Sistem Pencatatan Transaksi Studi Kasus PT. XYZ	√	√	√	-	Penidas Fiodinggo Tanaem, Danny Manongga, dan Ade Iriani 2016
7	Implementasi Autentikasi *JSON Web Token* (*JWT*) sebagai Mekanisme Autentikasi Protokol *MQTT* pada Perangkat *NodeMCU*	√	√	√	-	Andri Warda Pratama, Adhitya Bhawiyuga, dan Mahendra Data 2018
8	Keamanan *RESTful Web Service* Menggunakan *JSON Web Token (JWT) HMAC SHA*-512	√	√	√	-	Alam Rahmatulloh, Heni Sulastri dan Rizal Nugroho 2018

Lanjutan Tabel 2.3 Matriks Ruang Lingkup Penelitian

9	*Scan-based Side-channel Attacks againts HMAC-SHA-256 Circuit Based on Isolating Bit-transition Group Using Scan Signatures*	-	√	√	-	Daisuke Oku, Masao Yanagisawa, dan Nozomu Togawa 2018
10	*Comparative Study of Symmetric and Asymmetric Cryptography Techniques*	-	-	√	√	Ritu Tripathi, dan Sanjay Agrawal

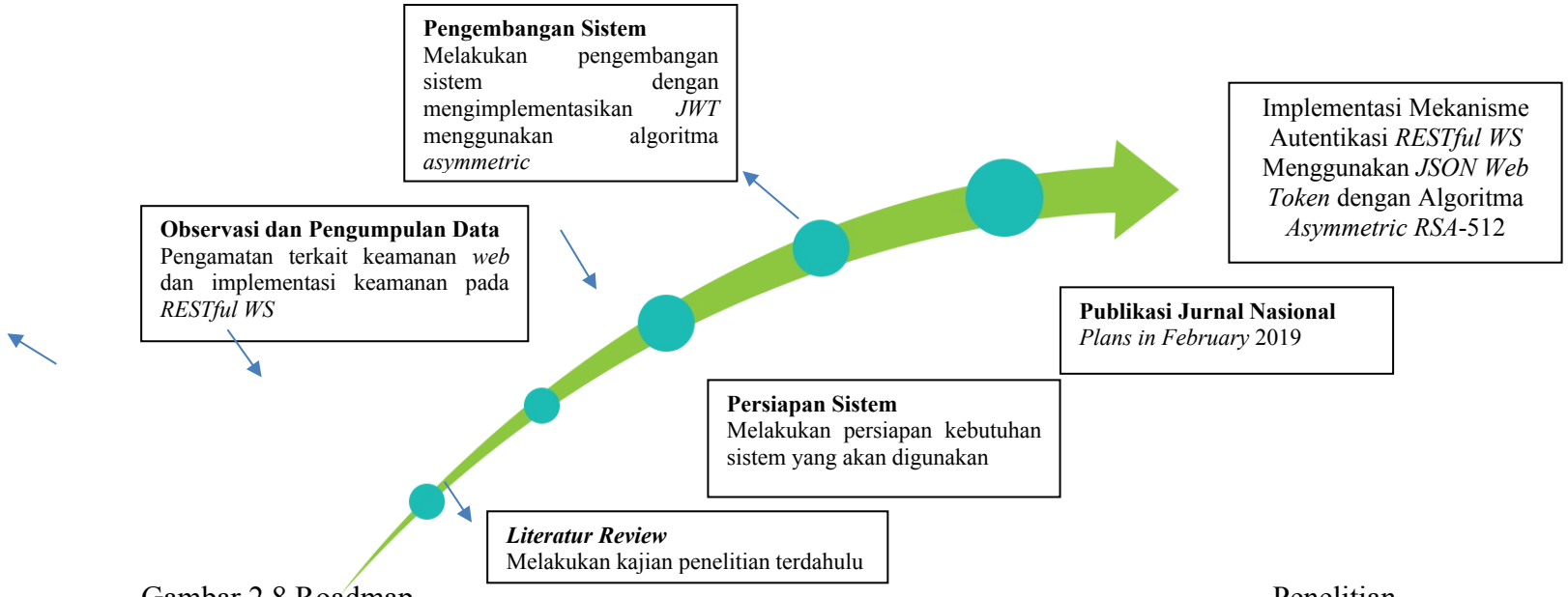
						2014
11	*Comparison of Symmetric and Asymmetric Cryptography with Existing Vulnerabilities and Countermeasures*	-	-	√	√	Yogesh Kumar, Rajiv Munjal, dan Harsh Sharma 2011
12	*Survey on Asymmetric Key Cryptography Algorithms*	-	-	-	√	S. Nithya, E. George Dharma Prakash Raj 2014
13	*Comparative Analysis of DES, AES, RSA Encryption Algorithms*	-	-	√	√	Priteshkumar Prajapati, Nehal Patel, Robinson Macwan, Nisarg Kachhiya, Parth Shah 2014
14	*A Comparative Survey of Symmetric and Asymmetric Key Cryptography*	-	-	√	√	Sourabh Chandra, Smita Paira, Sk Safikul Alam, dan Goutam Sanyal 2014

Lanjutan Tabel 2.3 Matriks Ruang Lingkup Penelitian

15	*Comparative Analysis of RSA and ElGamal Cryptographic Public-key Algorithms*	-	-	-	√	Andysah Putera Utama Siahaan, Elviwani, dan Boni Oktaviana 2018
16	Implementasi Mekanisme Autentikasi *RESTful WS* Menggunakan *JSON Web*	√	√	-	√	Muhammad Nur Arifin

	Token dengan Algoritma *Asymmetric* *RSA*-512					2018

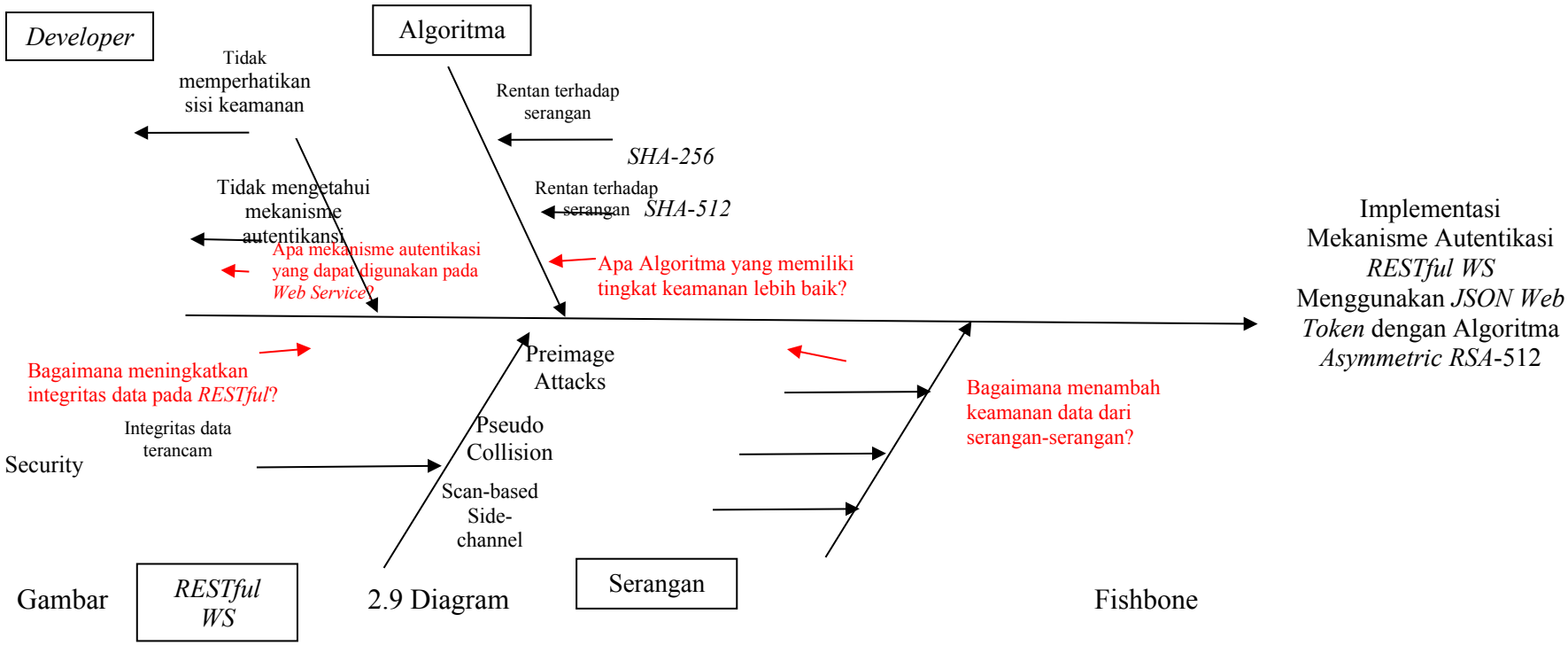
Roadmap Penelitian



Gambar 2.8 Roadmap Penelitian

Gambar 2.8 merupakan *roadmap* penelitian yang akan dilakukan. Langkah pertama yang dilakukan yaitu *literatur review*, kemudian observasi dan pengumpulan data dilanjutkan dengan persiapan sistem, pengembangan sistem, hingga publikasi jurnal nasional yang direncanakan akan dilaksanakan pada Februari 2019.

Diagram *Fishbone*



Gambar 2.9 Diagram Fishbone

Gambar 2.9 merupakan Diagram *Fishbone* pada penelitian kali ini. Masalah-masalah yang ditemukan yaitu mengenai mekanisme autentikasi dan algoritma yang dapat digunakan untuk menambah keamanan dan integritas data pada *RESTful WS*.