Ekstensi JWT Token dengan *Asymmetric Key* dalam Aspek *Non Repudiation* dan *Confidentiality*

Steven Nataniel - 13519002 (*Penulis*)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13519002@std.stei.itb.ac.id

***Abstrak*—JSON Web Token (JWT) adalah standar yang digunakan secara luas dalam aplikasi web untuk mentransfer dan mengamankan klaim antara pihak-pihak yang terlibat. JWT menggunakan tanda tangan digital untuk memastikan integritas data dan autentikasi pengguna. Namun, dalam implementasi JWT, penggunaan tanda tangan simetris dengan kunci rahasia bersama memiliki beberapa keterbatasan, terutama terkait dengan keamanan dan manajemen kunci.**

Kata Kunci—Json Web Token, JWT, kunci asimetrik, Non Repudiation.

I. PENDAHULUAN

JSON Web Token (JWT) adalah standar yang populer dalam keamanan aplikasi web untuk pertukaran informasi dan autentikasi pengguna. Namun, implementasi standar JWT masih menggunakan tanda tangan dengan kunci simetris atau kunci rahasia bersama (*shared secret key*) sehingga masih memiliki kekurangan terkait dengan manajemen kunci dan keamanan. Oleh karena itu, pengembangan ekstensi JWT dengan kunci asimetris (*asymmetric key*) telah menjadi perhatian utama dalam upaya meningkatkan keamanan dan fleksibilitas dalam penggunaan JWT.

Kunci asimetris, yang juga dikenal sebagai kriptografi kunci-publik, memanfaatkan pasangan kunci unik yang terdiri dari kunci pribadi (*private key*) dan kunci publik (*public key*). Kunci pribadi digunakan untuk menandatangani JWT, sedangkan kunci publik digunakan untuk memverifikasi tanda tangan. Pihak yang menerbitkan token dapat menggunakan kunci pribadi mereka untuk menandatangani JWT, sementara pihak yang menerima token dapat memverifikasi tanda tangan menggunakan kunci publik yang bersangkutan.

Dengan mengadopsi kunci asimetris dalam JWT token, manajemen kunci menjadi lebih efisien. Dalam implementasi JWT dengan tanda tangan simetris, kunci rahasia bersama harus secara aman didistribusikan di antara semua pihak yang terlibat, yang seringkali menjadi tantangan yang kompleks. Seperti contohnya, jika sebuah pihak teserang/terkena *hack*, maka pihak-pihak lain akan terkena dampak secara langsung karena kunci yang dipakai adalah kunci bersama. Namun, dengan kunci asimetris, distribusi kunci publik dapat dilakukan secara luas tanpa mengorbankan keamanan. Hal ini mempermudah pengelolaan kunci dan mengurangi kerumitan keamanan kunci bersama. Sehingga, jika terkena serangan, pihak-pihak lain dapat mengabaikan pesan dari pihak yang terkena *hack*. Karena, dengan kunci asimetris, kunci yang dipakai oleh satu pihak dengan pihak lainnya adalah berbeda.

Selain itu, kekurangan dari memakai tanda tangan simetris, yang dipakai oleh JWT standar, adalah semua pihak yang memakai kunci bersama harus tanpa terkecuali mempercayai semua yang mempunyai kunci tersebut. Akibatnya, jika ada pihak yang *malicious* berpura-pura sebagai pihak lain atau menyangkal, hal tersebut tidak bisa diverifikasi kebenarannya. Oleh karena itu, untuk mengatasi hal tersebut, diperlukan tanda tangan digital dengan kunci asimetris sehingga kasus *non repudation* bisa diminimalisir.

Payload pada JWT juga ada dalam format Base64, yang berarti tidak aman terhadap *confidentiality*. Memang, umumnya, JWT digunakan untuk mengonfirmasi *Integrity, Authentication* dan *Authorization*. Namun, jika diperlukan, bisa ditambahkan ekstensi untuk menambahkan aspek *confidentiality* juga.

Pada kasus spesifik, jika ingin merahasiakan pemberi otorisasi dan pengakses namun dapat dipastikan tindakan tersebut benar-benar dilakukan oleh orang sebenarnya dan tidak bisa disangkal, maka diperlukan kelima aspek tersebut. Contohnya dalam kasus-kasus rahasia negara seperti Badan Intelijen Negara ataupun dunia *financial*.

II. LANDASAN TEORI

A. Aspek-aspek Keamanan

Dalam konteks keamanan, terdapat beberapa aspek yang penting untuk diperhatikan. Berikut adalah beberapa aspek utama dalam keamanan:

1. Kerahasiaan (*Confidentiality*): Aspek kerahasiaan berkaitan dengan perlindungan terhadap akses yang tidak sah terhadap informasi sensitif. Hal ini dapat dicapai melalui teknik enkripsi yang mengubah informasi menjadi bentuk yang tidak dapat dibaca oleh pihak yang tidak berwenang. Enkripsi memastikan bahwa hanya pihak yang memiliki kunci dekripsi yang sesuai yang dapat mengakses dan membaca informasi yang dilindungi.
2. Integritas (*Integrity*): Aspek integritas melibatkan keabsahan dan ketidakberubahannya data selama transit atau penyimpanan. Integritas dapat diperiksa

dengan menggunakan teknik tanda tangan digital yang menggunakan algoritma kriptografi untuk menghasilkan tanda tangan unik pada data. Penerima dapat memverifikasi tanda tangan untuk memastikan bahwa data tidak diubah oleh pihak yang tidak berwenang selama perjalanan.

3. Otentikasi (*Authentication*): Aspek otentikasi berkaitan dengan memastikan identitas pengguna, perangkat, atau entitas lainnya yang terlibat dalam interaksi. Ini melibatkan penggunaan metode seperti kata sandi, kunci kriptografi, atau mekanisme otentikasi lainnya untuk memverifikasi identitas yang dinyatakan. Otentikasi yang kuat membantu mencegah akses yang tidak sah dan serangan peretasan.
4. Otorisasi (*Authorization*): Aspek otorisasi berkaitan dengan mengontrol akses ke sumber daya atau fungsionalitas tertentu berdasarkan hak atau peran pengguna. Hal ini melibatkan penggunaan kebijakan akses dan kontrol yang memastikan bahwa hanya pengguna yang sah yang memiliki hak akses yang sesuai untuk melakukan tindakan tertentu.
5. Ketersediaan (*Availability*): Aspek ketersediaan berkaitan dengan memastikan bahwa sistem atau layanan tetap tersedia dan dapat diakses oleh pengguna yang sah. Hal ini melibatkan langkah-langkah untuk mencegah dan mengatasi serangan atau gangguan yang dapat menghentikan atau mengganggu akses ke sistem atau layanan.
6. *Non-Repudiation*: Aspek *nonrepudiation* melibatkan memberikan bukti yang kuat untuk mencegah penyangkalan tindakan yang telah dilakukan. Teknik seperti tanda tangan digital digunakan untuk memberikan bukti matematis bahwa tindakan itu benar-benar dilakukan oleh pihak yang bersangkutan. Sehingga, pihak yang bersangkutan tidak bisa menyangkal bahwa aksi tersebut tidak ia lakukan.

B. JSON

JSON (JavaScript Object Notation) adalah sebuah format data yang ringan dan mudah dibaca oleh manusia maupun mesin. Format ini umumnya digunakan untuk pertukaran data antar aplikasi yang berjalan pada platform yang berbeda. JSON didasarkan pada sintak pada bahasa JavaScript, tetapi dapat digunakan dengan berbagai bahasa pemrograman. Dan sekarang JSON adalah format serialisiasi yang sangat populer pada aplikasi berbasis web.

JSON menggunakan struktur data sederhana berupa pasangan "nama-nilai" (key-value pairs) yang terdiri dari objek dan *array*. Objek dalam JSON didefinisikan dalam kurung kurawal dan terdiri dari pasangan nama-nilai yang dipisahkan oleh tanda koma. Setiap pasangan nama-nilai terdiri dari sebuah *string* yang akan menjadi *key* yang diikuti oleh tanda titik dua dan diikuti oleh sebuah nilai.

Berikut adalah contoh format data JSON.

```
  {
    "nama": "Alice Bob",
    "umur": 21,
    "alamat": {
      "jalan": "Jl. Ganesha No. 10",
      "kota": "Bandung"
    },
    "hobi": ["kriptanalisis", "tidur", "kriptografi"]
}
```

C. Digital Signature

Digital signature (tanda tangan digital) adalah metode yang digunakan untuk memverifikasi keaslian, integritas, dan tidak dapat disangkal (*non-repudiation*) dari data digital. Dalam konteks kriptografi, *digital signature* umumnya memanfaatkan teknik kriptografi asimetris. Berbeda dengan JSON Web Token, standar ini memakai tanda tangan digital dengan kunci simetris.

Proses digital signature melibatkan penggunaan pasangan kunci unik yang terdiri dari kunci pribadi (*private key*) dan kunci publik (*public key*). Pengirim data menggunakan kunci pribadi mereka untuk menghasilkan tanda tangan digital pada data yang ingin ditandatangani. Tanda tangan digital ini dapat diperiksa oleh penerima data menggunakan kunci publik yang sesuai. Jika tanda tangan dapat diverifikasi dengan menggunakan kunci publik yang benar, maka dapat dipastikan bahwa data tersebut berasal dari pengirim yang sah dan tidak mengalami perubahan selama pengiriman. Alur tanda tangan digital dapat dilihat pada Gambar 1.

Proses pembuatan tanda tangan digital melibatkan langkah-langkah berikut:

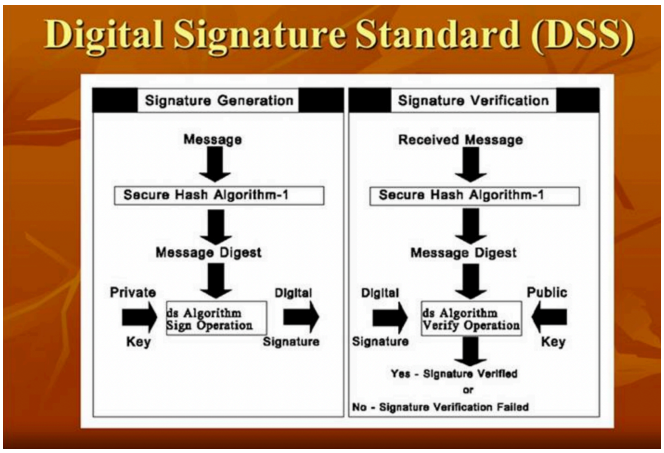
1. Pengirim mengonversi data menjadi pesan hash (*message digest*) menggunakan fungsi hash kriptografis yang aman. Pesan hash adalah representasi unik tetap dari data yang tidak dapat diubah menjadi data asli.
2. Pengirim mengenkripsi pesan hash dengan menggunakan kunci pribadi mereka. Proses ini menghasilkan tanda tangan digital yang unik.
3. Tanda tangan digital ini dikirim bersama dengan data asli ke penerima.

Penerima melakukan verifikasi tanda tangan digital dengan langkah-langkah berikut:

1. Penerima mengonversi data yang diterima menjadi pesan hash menggunakan fungsi hash yang sama yang digunakan oleh pengirim.
2. Penerima menggunakan kunci publik pengirim untuk mendekripsi tanda tangan digital yang diterima.
3. Jika hasil dekripsi tanda tangan digital sama dengan pesan hash yang dihasilkan dari data yang diterima,

maka tanda tangan digital dapat diverifikasi dan keaslian data dapat dipastikan.

Dengan menggunakan tanda tangan digital, dapat dijamin bahwa data yang dikirimkan tidak diubah atau dimanipulasi selama transit dan hanya berasal dari pengirim yang sah. Digital signature juga menyediakan mekanisme non-repudiation, yang berarti pengirim tidak dapat menyangkal tindakan mereka karena tanda tangan digital memberikan bukti matematis bahwa data tersebut berasal dari pengirim yang benar.



Gambar 1. Alur Tanda Tangan Digital

Sumber: https://signx.wondershare.com/knowledge/digital-signature-algorithm.html

D. Asymmetric Cryptography

Kriptografi asimetris juga dikenal sebagai kriptografi kunci-publik, adalah suatu metode kriptografi yang melibatkan penggunaan dua kunci yang berbeda yang disebut kunci pribadi (*private key*) dan kunci publik (*public key*). Berkebalikan dengan kriptografi simetris yang menggunakan kunci yang sama untuk enkripsi dan dekripsi, kriptografi asimetris menggunakan pasangan kunci yang berbeda.

Dalam asimetris kriptografi, kunci publik digunakan untuk mengenkripsi data, sementara kunci pribadi digunakan untuk mendekripsi data. Kunci publik dapat dibagikan secara bebas kepada semua orang, sementara kunci pribadi harus dijaga secara ketat dan hanya diketahui oleh pemiliknya. Proses enkripsi dengan kunci publik hanya dapat di-dekripsi dengan menggunakan kunci pribadi yang sesuai, dan sebaliknya. Namun, pada tanda tangan digital, fungsi kunci dibalik kegunaannya untuk mengatasi masalah *nonrepudiation*. Pada tanda tangan digital, kunci pribadi digunakan untuk mengenkripsi dan kunci publik untuk dekripsi.

Keuntungan utama dari kriptografi asimetris adalah sebagai berikut:

1. Keamanan
   Kunci pribadi adalah rahasia dan hanya diketahui oleh pemiliknya. Dengan demikian, bahkan jika kunci publik diketahui oleh semua orang, data yang dienkripsi tetap aman karena hanya dapat di-dekripsi dengan kunci pribadi yang bersangkutan. Hal ini menjadi lebih aman dibandingkan dengan kriptografi simetris.
2. Non-Repudiation: Dalam kriptografi asimetris, tanda tangan digital dapat digunakan untuk memberikan mekanisme non-repudiation. Tanda tangan digital dapat digunakan oleh penerima untuk memverifikasi keaslian pengirim dan integritas data yang dikirimkan, sementara pengirim tidak dapat menyangkal tindakan tersebut karena hanya mereka yang memiliki kunci pribadi yang sesuai untuk menghasilkan tanda tangan yang valid.

Meskipun kriptografi asimetris memiliki banyak keuntungan, kelemahannya adalah proses enkripsi dan dekripsi yang lebih lambat dan memerlukan lebih banyak sumber daya komputasi dibandingkan dengan kriptografi simetris. Oleh karena itu, biasanya asimetris kriptografi digunakan dalam skenario yang membutuhkan keamanan yang lebih kuat, sementara kriptografi simetris digunakan untuk enkripsi dan dekripsi data yang lebih cepat dan efisien.

E. JSON Web Token

JWT (JSON Web Token) adalah standar terbuka (RFC 7519) yang digunakan untuk merepresentasikan dan mentransfer informasi secara aman antara beberapa pihak. JWT sering digunakan dalam lingkungan aplikasi web dan API untuk mengautentikasi dan mengotorisasi pengguna.

Komponen utama dari JWT adalah sebagai berikut:

1. Header: Bagian header JWT berisi tipe token (typ) dan algoritma tanda tangan (alg) yang digunakan. Header umumnya berisi JSON yang dienkripsi dengan Base64 URL encoding.
2. Payload: Bagian payload JWT berisi klaim atau informasi terkait pengguna yang ingin ditransfer. Klaim dapat berupa klaim terstandarkan seperti ID pengguna (sub), waktu kadaluwarsa (exp), dan hak akses (*scope*), serta klaim khusus yang ditentukan oleh aplikasi tertentu. Payload juga dienkripsi menggunakan Base64 URL encoding.
3. Signature: Bagian signature JWT digunakan untuk memastikan integritas dan autentikasi token. Signature dihasilkan dengan menggunakan algoritma kriptografi pada header dan payload, serta kunci rahasia yang hanya diketahui oleh pihak yang mampu memverifikasi token. Signature mencegah manipulasi data pada header dan payload, serta memastikan bahwa token berasal dari sumber yang sah.

III. Rancangan, Implementasi Dan Hasil

Dalam mengimplementasikan hal ini, seharusnya tidak sulit karna standar JWT sendiri sudah menyediakan ruang untuk melakukan ekstensi.

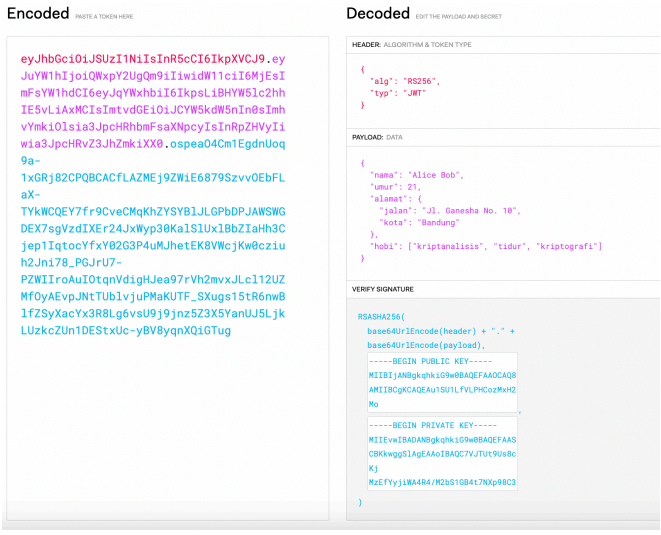
A. Non Repudiation

Dalam menerapkan *nonrepudiation*, langkahnya cukup *straightforward* yaitu dengan mengubah algoritma untuk

membuat *signature* menjadi kriptografi asimetris. Karena umumnya, JWT memakai HMAC dengan *shared key* untuk melakukan ini. Berikut adalah langkah-langkah untuk mengubah JWT menggunakan algoritma kriptografi asimetris.

1. Pembuatan Kunci Asimetris
2. Generate Pasangan Kunci: Buatlah pasangan kunci publik dan kunci pribadi menggunakan algoritma kriptografi yang mendukung kunci asimetris, seperti RSA atau ECDSA.
3. Penerbitan Token: Tentukan informasi atau klaim yang ingin dimasukkan dalam payload JWT. Ini bisa berupa ID pengguna, waktu kadaluwarsa, peran, atau klaim lain yang relevan. Kemudian, tentukan algoritma tanda tangan yang akan digunakan, misalnya RSA atau ECDSA, dan atur tipe token menjadi JWT.
4. Mengenkripsi dan Menandatangani: Gunakan kunci pribadi untuk mengenkripsi dan menandatangani payload dan header.
5. Menghasilkan Token: Gabungkan header, payload, dan tanda tangan menjadi satu token JWT yang lengkap. Biasanya penggabungan akan dipisahkan dengan tanda titik agar dapat dipecah kembali menjadi 3 bagian yaitu header, payload, dan signature.

Berikut adalah contoh pembuatan JWT Token dengan RSA256.



Gambar 2. Simulasi Pembuatan JWT Token

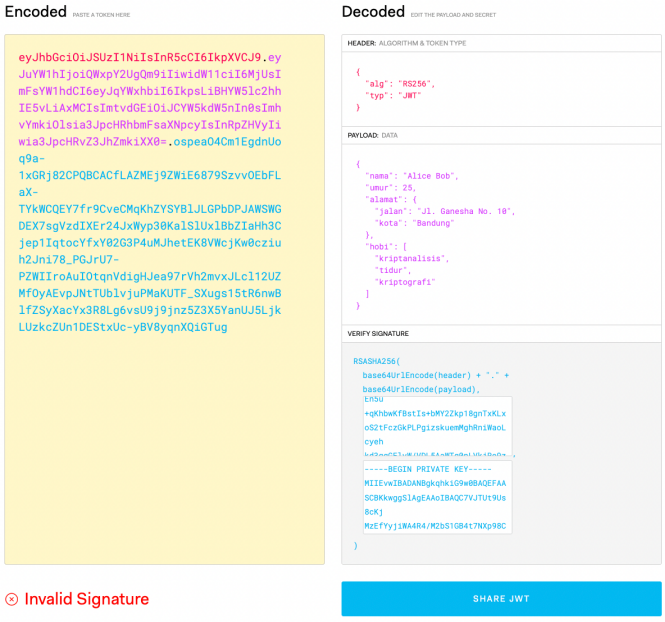
Bisa dilihat pada Gambar 2, header dienkripsi menggunakan Base64 karena header akan berisi algoritma yang dipakai. Pada kasus ini, payload juga dienkripsi menggunakan Base64. Namun, akan ada signature yang ditandai dengan warna biru. Signature ini bergantung dengan header dan payload serta kunci *private* yang dipakai untuk mengenkripsi.

Sedangkan, untuk melakukan verifikasi dan validasi token adalah sebagai berikut.

1. Verifikasi Tanda Tangan: Gunakan kunci publik yang sesuai untuk memverifikasi tanda tangan pada token. Verifikasi ini memastikan bahwa token berasal dari pihak yang sah dan integritasnya tidak dilanggar.
2. Validasi Klaim: Periksa klaim dalam payload JWT untuk memastikan bahwa token memiliki waktu kadaluwarsa yang valid, hak akses yang diperlukan, atau klaim lain yang diperlukan untuk mengotorisasi pengguna. Pemeriksaan waktu kadaluwarsa adalah opsional, ini sangat bergantung dengan pembuat token yang dapat memilih untuk memasukkan waktu kadaluwarsa atau tidak.
3. Penggunaan Token: Setelah token diverifikasi dan klaimnya valid, penerima dapat menggunakan informasi dalam token untuk mengidentifikasi pengguna, memberikan akses, atau melakukan tindakan lain yang sesuai.

Penerapan ini akan memungkinkan penggunaan kunci asimetris dalam JWT untuk mencapai keamanan yang lebih dapat dipercaya dalam pertukaran informasi dan autentikasi pengguna karena pengguna tidak dapat melakukan penyangkalan terhadap pembuatan token. Karena pihak yang menerima token tidak bisa membuat token palsu sebab pihak penerima hanya bisa melakukan validasi dengan kunci publik yang dimilikinya.

Misalkan, jika ada orang jahat mengubah data pada payload. Maka akan terverifikasi karena signaturenya tidak cocok. Terlihat pada gambar berikut, umur pada payload diubah dari 21 menjadi 25. Dan dihasilkan, akan langsung ketahuan jika payload diubah karena yang mengubah tidak mengubah *signature* karena tidak memiliki kunci private yang sesuai untuk membuat *signature*.



Gambar 3. Simulasi Pengubahan Payload pada JWT Token

B. Confidentiality

Salah satu cara untuk menambahkan kerahasiaan pada JWT dengan menggunakan kunci asimetris adalah dengan mengenkripsi payload menggunakan kunci publik penerima. Dalam skenario ini, pengirim membuat JWT seperti biasa dengan header dan payload yang ditandatangani menggunakan kunci pribadi pengirim. Namun, sebelum mengirimkan token, pengirim mengenkripsi payload menggunakan kunci publik penerima. Hanya penerima yang memiliki kunci pribadi yang sesuai yang dapat mendekripsi payload dan mengakses informasi sensitif di dalamnya. Dengan menerapkan enkripsi menggunakan kunci publik penerima, kerahasiaan payload JWT dapat dipertahankan saat pengiriman dan hanya dapat diakses oleh pihak yang memiliki kunci pribadi yang sesuai.

Bisa dilihat, payload pada teks kuning yang diberikan pada tabel berikut awalnya dienkripsi memakai Base64. Agar *confidentiality* terpenuhi, maka payload tersebut dikunci dengan RSA2048. Kunci publik dan privat yang dipakai untuk mengenkripsi adalah sebagai berikut.

AWAL:

eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJuYW1hIjoiQWxpY2UgQm9iIiwidW11ciI6MjEsImFsYW1hdCI6eyJqYWxhbiI6IkpsLiBHYW5lc2hhIE5vLiAxMCIsImtvdGEiOiJCYW5kdW5nIn0sImhvYmkiOlsia3JpcHRhbmFsaXNpcyIsInRpZHVyIiwia3JpcHRvZ3JhZmkiXX0.ospeaO4Cm1EgdnUoq9a-1xGRj82CPQBCACfLAZMEj9ZWiE6879SzvvOEbFLaX-TYkWCQEY7fr9CveCMqKhZYSYBlJLGPbDPJAWSWGDEX7sgVzdIXEr24JxWyp30KalSlUxlBbZIaHh3Cjep1IqtocYfxY02G3P4uMJhetEK8VWcjKw0cziuh2Jni78_PGJrU7-PZWIIroAuIOtqnVdigHJea97rVh2mvxJLcl12UZMfOyAEvpJNtTUblvjuPMaKUTF_SXugs15tR6nwBlfZSyXacYx3R8Lg6vsU9j9jnz5Z3X5YanUJ5LjkLUzkcZUn1DEStxUc-yBV8yqnXQiGTug

SESUDAH:

eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.T34+UolHK5BgxXBgsJNPuQCWgugHyzaFiDrLBdIz15f0TQDp123nlDWteNwlPv/xMtRyakBjjjFiItK2EmcFoO2eET8oW9PtMcmr5MUj39Ag6e8AAPJRkrfJ7FQ5Tt5wZowiH84rYC6G1QUkIwcVv9M6dpBBmLRYNz/5lghIni3gls/z/p2+6jmIOhVTXtUTiiLPDCrPatoCP/GN3CB+rW+h8M9GBtMBQAqIxf816TyZnbw7lPYMKGYeSFPrq9lH4wDV8wUsz5/X8MH+toTRBXn6XY17agR0ddmcdBRKX4ozB6AMq/Ti0qWJkO7Pm+xkigWk0kq9sIG2xBWeZn0DqQ==.ospeaO4Cm1EgdnUoq9a-1xGRj82CPQBCACfLAZMEj9ZWiE6879SzvvOEbFLaX-TYkWCQEY7fr9CveCMqKhZYSYBlJLGPbDPJAWSWGDEX7sgVzdIXEr24JxWyp30KalSlUxlBbZIaHh3Cjep1IqtocYfxY02G3P4uMJhetEK8VWcjKw0cziuh2Jni78_PGJrU7-PZWIIroAuIOtqnVdigHJea97rVh2mvxJLcl12UZMfOyAEvpJNtTUblvjuPMaKUTF_SXugs15tR6nwBlfZSyXacYx3R8Lg6vsU9j9jnz5Z3X5YanUJ5LjkLUzkcZUn1DEStxUc-yBV8yqnXQiGTug

KUNCI PUBLIK:

-----BEGIN PUBLIC KEY-----

MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAlDgTFGAYX93L1Qbbk3Yc

EBt9bFwZzkxVH4uIR/dPfrK9h7HoiW3ayDPwnbRY5g9EWoONlryuvrMm14MHhB6/

0b48BVxa0uHKGEv1olI3IueXEl+/y2QxsOdITaps4vtFN5YiBihzk0Mq1N87BQqQ

F9wQiz1i9An6J8/s/ktE4vQ409F3BYiZKIVpA+1UX9dUrDBVZ+R+yMKz2C0FQ5Y2

51PWeVt84ZFEcW083iVzfPYIWakK2vAQPu+v3NJYxpYShlTzdnSR28zam91uqna5

WJSj++cioJx0FDuRDtY6oOYxZJDX4usN5khNbXw+eBsVR6Ooo9ImLDQgF6HkQPdM

xwIDAQAB

-----END PUBLIC KEY-----

KUNCI PRIVAT:

-----BEGIN RSA PRIVATE KEY-----

MIIEogIBAAKCAQEAlDgTFGAYX93L1Qbbk3YcEBt9bFwZzkxVH4uIR/dPfrK9h7Ho

iW3ayDPwnbRY5g9EWoONlryuvrMm14MHhB6/0b48BVxa0uHKGEv1olI3IueXEl+/

y2QxsOdITaps4vtFN5YiBihzk0Mq1N87BQqQF9wQiz1i9An6J8/s/ktE4vQ409F3

BYiZKIVpA+1UX9dUrDBVZ+R+yMKz2C0FQ5Y251PWeVt84ZFEcW083iVzfPYIWakK

2vAQPu+v3NJYxpYShlTzdnSR28zam91uqna5WJSj++cioJx0FDuRDtY6oOYxZJDX

4usN5khNbXw+eBsVR6Ooo9ImLDQgF6HkQPdMxwIDAQABAoIBAHeKIhk0x3PJ9jMB

LOZpQ4jaIvXZ8hYoCh8+8dLY6wxMK//cxKbSKrl86nrGkUw60+4MW41O78BtzM3T

p6QvCJWoz8+jBL0/zTZ7TMh1dnJ4RadUAeOQE2ORpQa+hMMr1E9YjmhZB52ldk3p

HDicpEoZ+l3STJR114N0drdBDXXcNOmn3HP9vFI0qVYL9OxgGkHzf0wWvnMQkYP/

JPA/zx459RlQn85bAuI9jqtIZxuO0z4QozipTxSj1iUBdeUurn7GlYrmnjJ9YpF1

69HIrr65yQU9Dbv8+kWGq7+qf5Rj+43JToo383OAqRAdNuaQbdOLsn0hbxB5uR9+

gJipZAECgYEA4MhhBMoIkdGv0cP438B3koMC0XFe/wgsVnbSYZgie2FMacF2KIrI

ol/XGli4aYsH/5ZemiIF21DI4cK4mFu62JNd9Rytg9PRu3n5lcoVBemP95HZxxYC

ANlOJ8sO4/tGZGKMcC8G5cw+SolPiIGTcVqP8O8EYnWaN+8fwcIR10ECgYEAqM2n

XAz0Yj1Lopbqccrcwx6cXS3wWbG2nmASfQDdoWrwljNGz5jzF/MhHXrkLHMBX+tf

+jCLAdVM/vUjWMa91qdEizAks1HQoXGDJfvop0UhYta/CTMPquWuDdSVeZoZY3P8

mJWFZK9Za4rpEE1dWKkbl7x05dcHSAMKwWZ56gcCgYAYo95+t+bYlTPAusq9PEvf

M51LeDIBotO5Zq0JJbQh8jh+H0bkitITeo4JxU/xwdvOmJ0mKDScJe7fUIkLqoKA

2yahh9M9cCq9clF5qf1D+KZfbQ2CVVdC4I9ArXGkbBb7ShI6+ey3o/vwD704KXwQ

IO1Ga+z/HgUZvIyuoKd0AQKBgHOWPt9dT4plhoES7f51dq11LITxi/8fHXldN8u4

6Ao7+/odqoNErFDd/vL8z3lQVsSjm9jpnIXo1QxQN5/2ls+BQgI9OSRBBadG3xMH

Z8wCHlZbzfwypqvTElgOFTxrNK9RIMoquGi1WXm2aNjs4NpxEZ/ZQAKSNz4sNxUv

YpPJAoGASfxDt9AoLin6ojT7l9dwSc7mA5h6MQvfmK9KWkRZPd4RKSHpCLFLMlJs

0eAtyx8ookT6C2Vkwh0eyWd6z+jZvtcM1yqFCp9I2oE4uQGG5ZjwFqZGQ20fkPyw

eJ105Y9vnFNTJ4bsrgeusQLlFFuA9HifrXGi/WkXGgX7LGCtSBI=

-----END RSA PRIVATE KEY-----

Perlu diketahui, apabila payload melewati batas RSA, maka diperlukan untuk memecah *string* Base64 payload tersebut menjadi beberapa bagian lalu digabungkan dengan delimiter tertentu. Dan untuk melakukan dekripsi, maka harus dipecah ke beberapa bagian lalu didekripsi dan digabungkan kembali.

Metode lain untuk menambahkan kerahasiaan pada JWT dengan kunci asimetris adalah dengan menggunakan enkripsi end-to-end. Dalam skenario ini, pengirim mengenkripsi seluruh JWT menggunakan kunci publik penerima sebelum mengirimkannya. Penerima menggunakan kunci pribadi yang sesuai untuk mendekripsi dan mendapatkan akses ke informasi yang terkandung dalam token. Dengan menerapkan enkripsi end-to-end, baik header maupun payload token dilindungi secara rapi, dan informasi sensitif dalam token hanya dapat dibaca oleh penerima yang memiliki kunci pribadi yang sesuai. Pendekatan ini memastikan bahwa kerahasiaan token dipertahankan selama pengiriman dan hanya pihak yang berwenang yang dapat mengakses isinya. Namun, masalahnya adalah *end-to-end encryption* pasti adalah *string* panjang yang melewati batas RSA. Sehingga, kurang mangkus dan efisien apabila diterapkan kecuali memang diperlukan untuk mengunci semua bagian (header, payload, signature) dalam JWT.

IV. Kesimpulan

Dapat disimpulkan bahwa penggunaan kriptografi asimetris pada JSON Web Token (JWT) memberikan manfaat yang signifikan dalam hal *nonrepudiation* dan *confidentiality*. Dengan menggunakan kunci asimetris, tanda tangan digital pada JWT memastikan bahwa token tidak dapat disangkal oleh pengirim, sehingga mencapai aspek *nonrepudiation*. Tanda tangan digital ini memverifikasi integritas dan otentikasi token, sehingga penerima dapat memastikan bahwa token berasal dari pengirim yang sah dan tidak mengalami perubahan selama pengiriman. Selain itu, dengan menerapkan enkripsi pada payload menggunakan kunci asimetris, kerahasiaan informasi sensitif dalam token terjaga dengan baik. Hal ini memastikan hanya pihak yang memiliki kunci pribadi yang sesuai yang dapat membaca dan mendapatkan akses ke informasi sensitif di dalam payload. Dengan kombinasi kriptografi asimetris, tanda tangan digital, dan enkripsi payload, JWT dapat mencapai tingkat keamanan lebih diperkuat daripada JWT standar, sehingga yang sebelumnya hanya dapat memastikan *integrity, authentication,* dan *authorization* dapat ditambah *confidentiality* dan *nonrepudiation*. Hal ini akan sangat berguna apabila diterapkan dalam layanan-layanan yang sangat sensitif seperti dalam dunia *financial technology* atau rahasia negara seperti Badan Intelijen Negara. Contohnya, jika ingin merahasiakan identitas pengakses/pemberi otorisasi dengan tindakan yang tidak bisa disangkal.

References

[1] Munir, Rinaldi. (2021). Slide Perkuliahan IF4020 Kriptografi. Di akses melalui https://informatika.stei.itb.ac.id/.

[2] Auth0. (2023). Website JWT. Di akses melalui https://jwt.io/.

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 22 Mei 2023

Steven Nataniel