Implementasi Skema Key Management Berbasis Algoritma Elliptic Curve Cryptography pada Komunikasi Sistem Smart Grid

Naura Valda Prameswari (18221173)
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
nauravalda.p@gmail.com

***Abstrak*—Saat ini, *smart grid* menjadi pilihan bagi penyedia energi listrik untuk sistem distribusi energi listrik. Sistem *smart grid* memungkinkan penyesuaian jumlah pembangkitan listrik yang tepat dengan menyediakan kemampuan untuk memantau perilaku konsumsi pelanggan. Teknologi *smart grid* menawarkan solusi efisiensi energi tetapi juga menghadapi tantangan keamanan komunikasi. Keamanan komunikasi dalam sistem *smart grid* merupakan aspek krusial yang perlu diperhatikan mengingat pengguna dari sistem ini terdiri dari berbagai kalangan mulai dari individu hingga bisnis. Makalah ini membahas skema manajemen kunci kriptografi yang dirancang untuk meningkatkan keamanan komunikasi dalam sistem *smart grid.* Manajemen kunci kriptografi mencakup pembangkitan kunci publik dan privat serta distribusi kunci publik. Sistem smart grid hanya memiliki kemampuan komputasi dan memori yang terbatas, sehingga diperlukan kunci dengan panjang bit minimal namun memiliki level keamanan yang tinggi. Algoritma ECC dipilih karena memenuhi kriteria tersebut. Implementasi skema ini diharapkan dapat meningkatkan keamanan dan keandalan komunikasi dalam sistem *smart grid*, sehingga mendukung operasi yang lebih aman dan efisien.**

Kata kunci—smart grid; elliptic curve cryptographic; key management

I. PENDAHULUAN

Sistem *smart grid* memungkinkan penyesuaian jumlah pembangkitan listrik yang tepat dengan menyediakan kemampuan untuk memantau perilaku konsumsi pelanggan. Sistem smart grid memungkinkan peningkatan efisiensi dan keandalan distribusi energi listrik dari sumber produksi listrik hingga konsumen. Sistem ini mencakup generator energi dan perangkat transmisi energi serta komponen seperti sensor, *advanced metering infrastructure* (AMI), penyimpan energi, dan juga *information gateways* yang beroperasi secara *real-time* [1]. AMI merupakan sistem yang digunakan untuk mengukur, mengumpulkan, menyimpan, menganalisis, dan memakai data pemakaian energi listrik [2]. Selain itu, smart grid menggunakan perangkat sensor yang bertanggung jawab untuk mengamati kinerja sistem serta mendeteksi setiap gangguan operasional. Komponen-komponen tersebut juga saling berkomunikasi sehingga membutuhkan distribusi informasi antar perangkatnya.

Keamanan informasi pada sistem *smart grid* sangat penting dari sisi keberlanjutan operasional, hingga kepentingan nasional. *Smart grid* mengelola distribusi listrik secara luas dan kompleks, serangan terhadap sistem *smart grid* akan menyebabkan gangguan besar dalam pasokan listrik yang dapat memengaruhi jutaan orang dan bisnis. Selain itu, sistem energi listrik merupakan infrastruktur kritis bagi sebuah negara yang dikelola oleh badan pemerintahan. Sehingga, gangguan pada sistem ini dapat berdampak pada keamanan nasional dan kestabilan ekonomi negara.

Celah keamanan informasi pada sistem *smart grid* dapat digunakan untuk manipulasi data penggunaan energi. Setelah celah keamanan ditemukan, penyusup dapat mengubah data penggunaan energi untuk mengurangi tagihan mereka sendiri atau meningkatkan tagihan pengguna lain, merugikan penyedia layanan listrik dan konsumen [3]. Penyusup juga dapat mencuri data penggunaan listrik seseorang untuk melakukan surveilans atau merencanakan tindakan kriminal seperti mengetahui kapan rumah kosong, dan sebagainya.

Karena sifat kompleks dari *smart grid* dan beragamnya kebutuhan keamanan, merancang skema autentikasi dan *key management* yang sesuai merupakan tantangan tersendiri. Untuk jaringan yang sensitif terhadap latensi seperti *smart grid*, skema autentikasi dan *key management* yang digunakan harus mampu menahan semua serangan keamanan, namun melibatkan operasi ringan dengan perhitungan yang minimal [4].

Berdasarkan analisis yang dilakukan oleh C. Varma [5] terkait perbandingan performa ECC dan RSA, didapatkan bahwa dari sisi komputasi, ECC memiliki efisiensi komputasi 10 kali lebih tinggi daripada RSA. Dari segi panjang kunci, ECC lebih pendek dengan proses pembangkitan kunci yang lebih cepat dibandingkan RSA. Kunci ECC sepanjang 160-bit menyediakan tingkat keamanan yang sama dengan 1024-bit kunci RSA. Selain itu, ECC juga menawarkan penghematan bandwidth yang dapat dipertimbangkan dibanding RSA. Dari sisi enkripsi, ECC memiliki kecepatan yang lebih cepat dari RSA, namun untuk deskripsi, RSA memiliki kecepatan yang

lebih cepat dari ECC. Dengan demikian, skema autentikasi dan key management yang diajukan pada makalah ini berbasis pada teknik kriptografi ECC.

II. TINJAUAN PUSTAKA

A. *Smart Grid*

Smart grid diartikan sebagai jaringan listrik yang secara efisien dapat mengintegrasikan perilaku dan aksi semua pengguna yang terhubung dengan sistem — generator, pelanggan, dan lain-lain — untuk memastikan sistem tenaga listrik yang efisien secara ekonomi dan berkelanjutan dengan kerugian yang rendah serta kualitas dan keamanan yang tinggi [7]. Smart grid memiliki beberapa komponen yang masing-masing memiliki fungsi dan lingkungan implementasi yang berbeda. Berikut merupakan komponen-komponen yang terdapat pada smart grid dan pemetaannya.

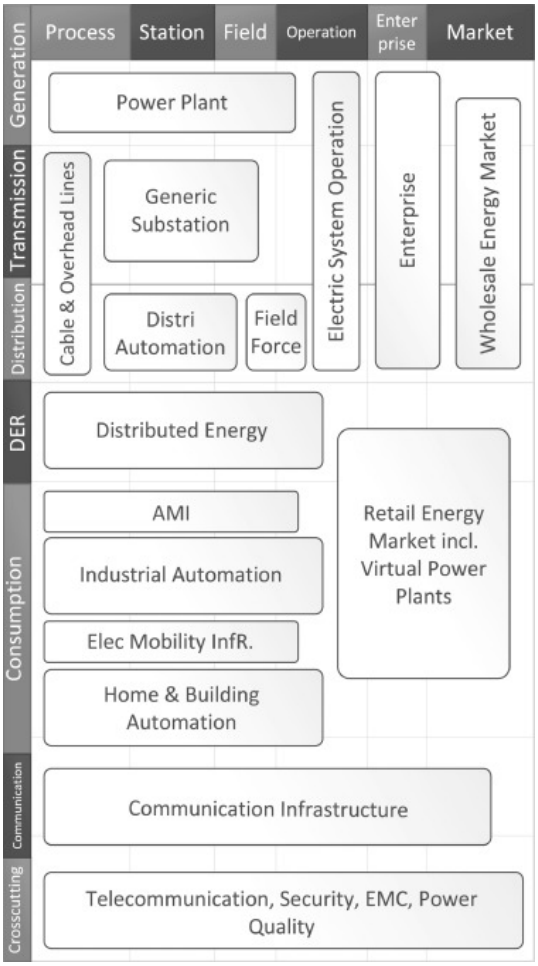

Fig. 1. Pemetaan Komponen Smart Grid Dari IEC Smart Grid Standards Map

Berdasarkan pemetaan tersebut, dapat dilihat bahwa sistem smart grid memiliki AMI (Advanced Metering Infrastructure). AMI berfungsi untuk mengumpulkan dan mengirim data penggunaan energi dari konsumen ke perusahaan penyedia energi untuk dipantau dan dianalisis.

Sistem *smart grid* minimal memerlukan skema key management dan autentikasi dalam infrastruktur komunikasi antar perangkat khususnya pada AMI, karena AMI merupakan komponen kunci dalam sistem smart grid. AMI berfungsi untuk mengumpulkan dan mengirim data penggunaan energi dari konsumen ke perusahaan utilitas. Keamanan informasi sangat penting untuk melindungi data pelanggan dan memastikan keakuratan tagihan. Berikut merupakan skema sistem keamanan yang akan diterapkan untuk smart grid.

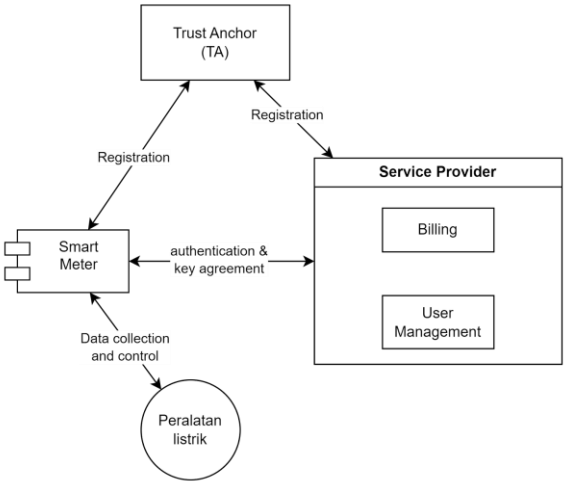

Fig. 2. Skema Kemanan Informasi *Smart Grid*

Smart meter adalah perangkat keras yang digunakan dalam mengumpulkan dan mengirim data penggunaan energi pada AMI. Dalam satu sistem smart grid, bisa terdapat banyak perangkat service provider device (misalnya dibagi berdasarkan area). Di satu service provider device, bisa terdapat banyak smart meter yang terhubung, sehingga smart meter harus memiliki data terkait service provider device tujuan pengiriman data.

B. *Elliptic Curve Cryptography*

ECC adalah kriptografi kunci-publik yang relatif lebih baru usianya dan dikembangkan oleh Neal Koblitz dan Victor S. Miller tahun 1985. Teori aljabar abstrak yang terkait dengan ECC adalah Galois Field [6]. Kurva eliptik dalam ECC biasanya didefinisikan oleh persamaan $y^2 = x^3 + ax + b$ di atas Galois Field. Koefisien a dan b menentukan bentuk kurva.

1) Pembangkitan Kunci:

a) Pemilihan kurva eliptik dan titik dasar G (generator point) pada kurva: Kurva eliptik yang dipilih harus sesuai dengan standar keamanan (NIST P-256, P-384, Curver25519, dan lain-lain).

b) Pemilihan bilangan acak: Menentukan d sebagai kunci privat, di mana $1 \leq d < n$, dengan n adalah order dari titik dasar G

c) Perhitungan kunci publik: Menghitung kunci publik Q dengan menggandakan titik dasar G sebanyak d kali, Q = d × G

2) Enkripsi

a) Merepresentasikan pesan menjadi titik pada kurva: Encode pesan menjasi titik di dalam kurva eliptik menggunakan metode Kolbitz, misalnya M = (x_M , y_M).

b) Pemilihan kunci acak k, di mana 1 ≤ k < n

c) Menghitung titik $C_1 = k \times G$ dan $C_2 = M + k \times Q$

d) Ciphertext adalah pasangan titik (C_1,C_2)

3) Dekripsi

a) Menghitung kembali pesan asli: Menghitung pesan M dengan menggunakan kunci privat d untuk menghitung k × Q = d × C_1 lalu menghitung nilai M dengan M = C_2 – d × C_1

Metode untuk encoding pesan menjadi titik di dalam kurva eliptik menggunakan metode Kolbitz. Pertama, dilakukan pemilihan sebuah kurva eliptik $y^2 = x^3 + ax + b$ mod p yang mengandung sejumlah N titik. Misalkan karakter-karakter penyusun pesan adalah angka 0, 1, 2, ..., 9 dan huruf A, B, C, ..., Z yang dikodekan menjadi 10, 11, ..., 35. Setiap karakter dalam pesan kemudian dikodekan menjadi nilai m di antara 0 dan 35. Selanjutnya, memilih sebuah bilangan bulat k sebagai parameter basis yang disepakati oleh kedua pihak. Untuk setiap nilai mk, dinyatakan x = m_k + 1, dan dilakukan substitusi x ke dalam persamaan $y^2 = x^3 + ax + b$ mod p, lalu dilakukan penentuan nilai y yang memenuhi. Jika tidak ada nilai y yang memenuhi, akan dihitung untuk x = m_k + 2, x = m_k + 3, dan seterusnya, hingga persamaan tersebut dapat dipecahkan. Pada proses decoding, untuk titik (x,y), ditentukan nilai m terbesar tetapi lebih kecil dari (x - 1) / k. Titik (x,y) dikodekan menjadi simbol m. Metode Kolbitz ini lebih aman dan memastikan setiap karakter pesan dapat diubah menjadi titik valid di dalam kurva eliptik. Algoritma ECC memiliki proses implementasi yang lebih kompleks dibanding algoritma-algoritma lainnya, Namun, algoritma ECC diklaim memiliki panjang kunci ECC yang lebih pendek daripada kunci RSA, namun memiliki tingkat keamanan yang sama dengan RSA. Sehingga dengan panjang kunci yang lebih pendek, dibutuhkan memori dan komputasi yang lebih sedikit. Algoritma ECC memiliki implementasi yang lebih kompleks dibandingkan dengan RSA. Selain itu, kurva eliptik yang digunakan harus dipilih dengan hati-hati untuk menghindari kelemahan keamanan.

C. Perancangan Skema Key Management

Berikut merupakan gambaran komponen-komponen yang akan ditinjau pada skema autentikasi dan key management sistem smart grid yang diajukan. Dalam satu sistem, bisa terdapat banyak perangkat service provider device (misalnya dibagi berdasarkan area). Di satu service provider device, bisa terdapat banyak smart meter yang terhubung, sehingga smart meter harus memiliki data terkait service provider device tujuan pengiriman data. Trust anchor merupakan komponen terpusat yang bertanggung jawab atas manajemen kunci dan verifikasi identitas. Trust anchor menyimpan parameter sistem untuk proses enkripsi dan dekripsi menggunakan ECC.

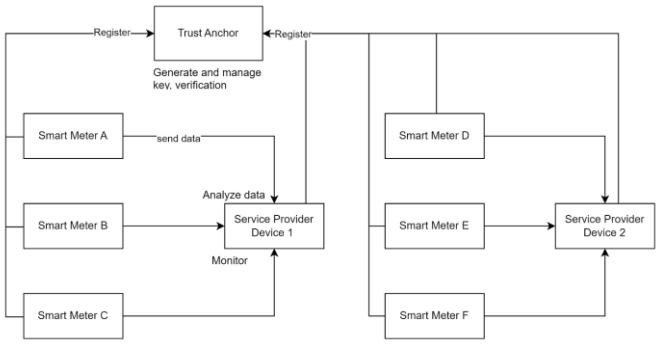

Fig. 3. Gambaran Komponen Pada Skema Manajemen Kunci

Perancangan skema autentikasi dan key management menggunakan kerangka skema yang diusulkan oleh Mahmood, K. dkk [4] dan D. Abbasinezhad-Mood. dkk [8]. Skema dibagi menjadi 3 fase, yaitu, fase inisialisasi trust anchor (tahap saat parameter kunci ECC ditentukan dan anti-collision hash function dipilih untuk mengamankan penyimpanan kunci terpusat.), fase registrasi perangkat (proses perangkat sistem smart grid mendaftarkan diri ke trust anchor untuk mendapatkan private key), dan fase autentikasi perangkat (fase saat smart meter menggunakan public key dari service provider device yang telah dipublikasikan oleh trust anchor untuk mengenkripsi pesan yang dikirimkan) yang dilakukan saat berkomunikasi.

Notasi yang digunakan pada skema yang diajukan dapat dilihat pada tabel berikut.

TABLE I. NOTASI DAN DEFINISI

Notasi	Definisi
SM	*Smart meter*
SP	*Service provider device*
(kSM, PSM)	*Private key* dan *public key* dari *smart meter*
(kSP, PSP)	*Private key* dan *public key* dari *service provider device*
P	*Generator point* pada ECC
p,q	Bilangan prima besar
(s, Ppub)	*Private key* dan *public key* pada *trust anchor*
hash()	Fungsi *hash*

1) Inisialisasi Trust Anchor

Trust Anchor adalah wadah penyimpanan kunci terpusat yang berfungsi untuk penerbitan sertifikat, verifikasi, dan manajemen kunci.

a) Trust anchor akan menentukan parameter untuk algoritma ECC berikut;

- Bilangan prima p ;

- *Elliptic Curve* E pada Galois Field Fp dengan persamaanE(Fp) : $y^2 = x^3 + ax + b$ mod p, a dan b adalah elemen dari Fp dan harus memenuhi syarat 4a3+27b2(mod p) 0;
- Generator point P pada kurva E, grup yang dihasilkan oleh P memiliki orde prima q

b) Trust anchor memilih anti-collision hash function hash();

c) Trust anchor memilih bilangan random s sebagai private key, dan menghitung public key dengan Ppub=s.P;

d) Trust anchor memublikasikan nilai {p,q,P,E(Fp),Ppub,hash()} dan menyimpan private key s secara rahasia.

2) Registrasi

Tahapan registrasi akan diterapkan untuk komponen-komponen sistem smart grid melalui *trust anchor*. Tahapan registrasi akan menghasilkan *private key* k yang disimpan pada memori perangkat, serta nilai hash *public key* + ID komponen yang dipublikasikan. *Private key* dihasilkan dari bilangan acak yang dikirimkan oleh komponen ke trust anchor yang digabungkan dengan ID komponen sebagai parameter fungsi hash, lalu mengonversinya menjadi suatu bilangan.

Pertama, registrasi *service provider device* (SP), mengirimkan ID SP dan bilangan acak r yang terenkripsi menggunakan PPub (*public key trust anchor*) ke *trust anchor*. Proses registrasi SP akan menghasilkan *private key* kSP yang akan disimpan di memori SP, dan nilai hash *public key* + ID dari SP yang dipublikasikan oleh *trust anchor*. *Service provider device* merupakan penerima pesan dari s*mart meter* (SM), sehingga smart meter dapat mengenkripsi pesan menggunakan *public key* milik *service provider device*.

Smart meter juga akan registrasi dengan parameter yang dikirim adalah bilangan acak, ID SM, dan ID SP yang terhubung. *Trust anchor* akan memberikan *private key*, dan nilai hash dari *public key* dan ID SP yang sebelumnya sudah tersimpan. Perangkat *service provider device* harus dipastikan sudah melakukan registrasi sebelum *smart meter* yang mengirimkan data ke *service provider device* tersebut melakukan registrasi.

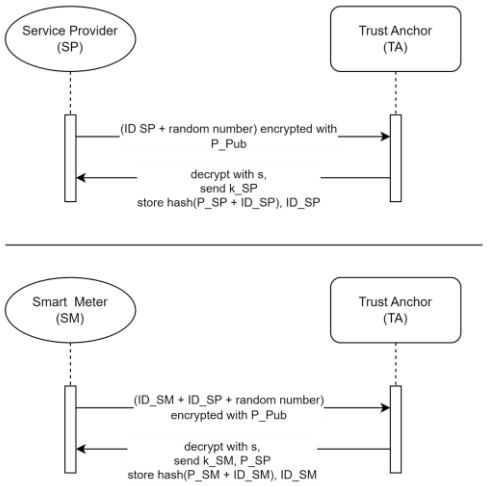

Fig. 4. Skema Registrasi Sistem

3) Autentikasi dan Pengiriman Pesan

Dalam fase autentikasi pada sistem *smart grid*, proses dimulai dengan pengiriman pesan dari S*mart Meter* (SM) ke s*ervice provider device* (SP). SM menggunakan *public key* SP yang telah dipublikasikan oleh *trust anchor* untuk mengenkripsi pesan yang akan dikirimkan. Hal ini bertujuan untuk memastikan bahwa hanya SP yang memiliki *private key* yang sesuai dapat mendekripsi dan membaca pesan tersebut. Selain enkripsi, SM juga menandatangani pesan untuk memastikan keasliannya. Pesan tersebut ditandatangani menggunakan *private key* SM, menghasilkan tanda tangan digital yang kemudian dikirim bersama pesan yang telah dienkripsi. SP, setelah menerima pesan terenkripsi, menggunakan *private key*-nya untuk mendekripsi pesan tersebut. Kemudian, SP menggunakan *public key* SM yang dipublikasikan oleh *Trust Anchor* untuk memverifikasi tanda tangan pesan. Jika verifikasi berhasil, SP dapat memastikan bahwa pesan tersebut benar-benar berasal dari SM yang sah dan tidak diubah selama transmisi. Proses ini menjamin keamanan dan integritas komunikasi antara SM dan SP dalam sistem *smart grid*.

Berikut merupakan skema autentikasi sistem.

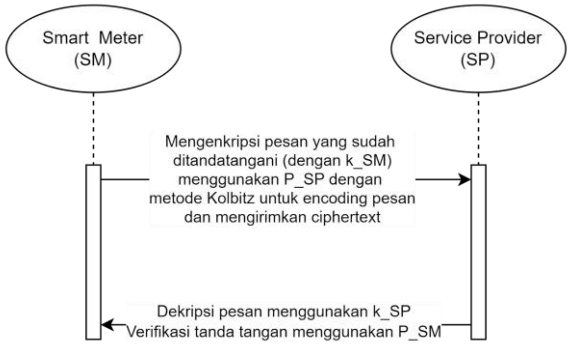

Fig. 5. Skema Autentikasi Sistem

III. PEMBAHASAN

Pada bab ini akan ditentukan kurva eliptik dengan menentukan performa tiap kurva eliptik dan tingkat keamanan dari setiap kurva eliptik berdasarkan standar keamanan yang sudah tersedia. Setelah kurva eliptik ditentukan, akan diimplementasikan skema autentikasi dan menejemen kunci dari

A. Pemilihan Kurva Eliptik

Beberapa alternatif kurva eliptik yang akan diuji adalah SECP192R1, SECP224R1, SECP256R1, 'SECP384R1', dan 'SECP521R1'. Pertama-tama, kurva akan dianalisis kecepatan komputasinya. Kecepatan komputasi kurva akan dihitung menggunakan fungsi tersebut dan dipetakan menggunakan library matplotlib.

```
def test_curve(curve, num):
```

```
    # generate key pair
    private_key = ec.generate_private_key(curve)
    public_key = private_key.public_key()
    # data smart meter
    message = {
        'meter_id': '123456',
        'timestamp': '2020-01-01T00:00:00',
        'power': 1000,
    }
    # Serialize the dictionary to a JSON string and then to
bytes
    message_bytes = json.dumps(message).encode('utf-8')

    # test signing
    start = time.time()
    for i in range(num):
        signature = private_key.sign(message_bytes,
ec.ECDSA(hashes.SHA256()))
    end = time.time()
    sign_time = (end - start) / num

    # test verification
    start = time.time()
    for i in range(num):
        public_key.verify(signature, message_bytes,
ec.ECDSA(hashes.SHA256()))
    end = time.time()
    verify_time = (end - start) / num
    return sign_time, verify_time
```

Fungsi `test_curve` dalam kode ini bertujuan untuk mengukur kinerja penandatanganan dan verifikasi tanda tangan digital menggunakan kurva eliptik tertentu dengan algoritma ECC. Fungsi ini menerima dua parameter: objek kurva eliptik dari *library* `cryptography.hazmat.primitives.asymmetric.ec` dan jumlah iterasi untuk pengujian. Pertama, fungsi ini menghasilkan pasangan kunci privat dan publik menggunakan kurva eliptik yang diberikan. Selanjutnya, sebuah pesan yang mewakili data smart meter didefinisikan sebagai kamus Python, yang kemudian diserialisasi ke dalam string JSON dan dikonversi menjadi bytes.

Pengujian penandatanganan dilakukan dengan mengukur waktu yang dibutuhkan untuk menandatangani pesan menggunakan kunci privat dalam jumlah iterasi yang ditentukan. Waktu rata-rata untuk penandatanganan dihitung dengan membagi total waktu dengan jumlah iterasi. Setelah itu, pengujian verifikasi dilakukan dengan mengukur waktu yang dibutuhkan untuk memverifikasi tanda tangan menggunakan kunci publik dalam jumlah iterasi yang sama, dan waktu rata-rata untuk verifikasi juga dihitung dengan cara yang sama. Akhirnya, fungsi ini mengembalikan waktu rata-rata untuk penandatanganan dan verifikasi dalam bentuk tuple.

Dari hasil tersebut, didapatkan kecepatan pemrosesan komputasi masing-masing kurva SECPXXXR1 sebagai berikut.

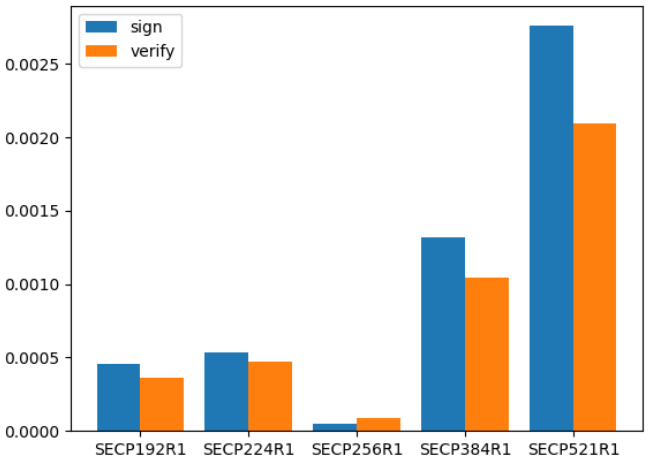

Fig. 6. Diagram Kecepatan Pemrosesan Kurva SECPXXXR1

Angka pada nama kurva menunjukkan panjang kunci yang digenerate. Misalnya, angka 192 pada SECP192R1 mengacu pada panjang kunci dalam bit yang digunakan oleh kurva elliptic tersebut. Dalam hal ini, SECP192R1 menggunakan panjang kunci 192 bit. Panjang kunci ini memengaruhi keamanan dan ukuran representasi data dalam protokol kriptografi yang menggunakannya. Semakin panjang kunci, semakin kuat keamanannya, tetapi juga membutuhkan lebih banyak komputasi untuk operasi kriptografi tertentu. Namun, hal yang menarik adalah untuk SECP256R1 memiliki kecepatan yang lebih tinggi dari pada SECP192R1 padahal memiliki kunci yang lebih panjang. Kurva SECP256R1 menggunakan kunci berukuran 256 bit, yang dianggap cukup aman untuk saat ini dan masa mendatang. Ukuran kunci yang lebih besar membuatnya lebih sulit untuk dipecahkan dengan serangan brute-force. Selain itu, kurva SECP256R1 telah diadopsi sebagai standar oleh banyak organisasi, termasuk NIST dan ANSI, yang menambah kepercayaan pada keamanannya.

Selain melakukan analisis pada kurva jenis SECPXXXR1, dilakukan juga analisis performa pada kurva jenis BrainpoolPXXXR1. Berikut adalah hasil analisis kecepatan pemrosesan komputasi pada masing-masing kurva BrainpoolPXXXR1.

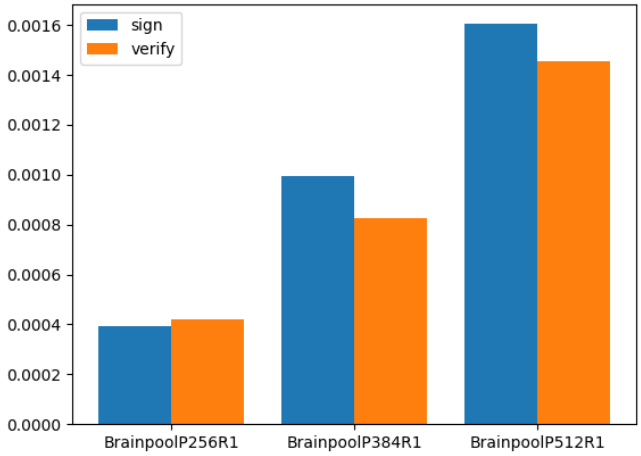

Fig. 7. Diagram Kecepatan Pemrosesan BrainpoolPXXXR1

Pada grafik tersebut, untuk kurva Brainpool, kecepatan pemrosesan komputasi sebanding dengan panjang kunci. Akan tetapi, dengan panjang kunci yang sama, yaitu BrainpoolP256R1 dan SECP256R1, didapatkan kecepatan pemrosesan yang berbeda. SECP256R1 lebih cepat dari sisi kecepatan pemrosesan komputasi daripada BrainpoolP256R1. Berdasarkan analisis tersebut, kurva yang dipilih pada algoritma ECC yang digunakan pada skema manajemen kunci pada sistem *smart grid*, karena menyediakan kecepatan komputasi yang lebih tinggi dengan panjang kunci hanya 256 bit.

B. Implementasi

Implementasi dilakukan sesuai dengan perancangan skema key management yang terdiri dari inisialisasi trust anchor, registrasi, dan autentikasi. Implementasi akan dilakukan berbasis CLI (Command Line Interface). Pada implementasi, akan dibuat satu file global sebagai Trust Anchor. Sebelum menjalankan fungsi-fungsi tersebut, trust anchor harus diinisialisasi untuk menghasilkan parameter sistem. Setelah trust anchor diinisialisasi, sistem dapat melakukan register dan autentikasi perangkat. Perangkat dibuat dalam bentuk kelas.

1) Inisialisasi Trust Anchor

```
from cryptography.hazmat.primitives.asymmetric import ec
from cryptography.hazmat.backends import default_backend

class SystemParameters:
  def __init__(self):
    self.__curve = ec.SECP256R1()
    self.__pub = None
    self.__priv = 0

  @property
  def curve(self):
    return self.__curve

  @property
  def pub(self):
    return self.__pub

  @property
  def s(self):
    return self.__priv

  def init_parameters(self):
    # generate key pair
    private_key = ec.generate_private_key(self.__curve)
    public_key = private_key.public_key()
    self.__priv = private_key
    self.__pub = public_key
```

Kelas SystemParameters digunakan untuk menginisialisasi parameter sistem, seperti kurva eliptik yang digunakan (SECP256R1 dalam kasus ini), serta menghasilkan kunci publik dan privat. Metode init_parameters menghasilkan pasangan kunci baru dan menyimpannya dalam instance kelas. SystemParameters dapat diakses secara global dengan cara instansiasi suatu variabel misalnya ‘params’ dengan objek SystemParameters(). Selanjutnya, ‘params’ dapat diimport pada file lain untuk digunakan sebagai parameter global untuk berkomunikasi dengan trust anchor dan menggunakan jenis kurva yang ditentukan. nilai p, q, P, Ppub dapat diakses pada public key. Private key tidak dapat diakses oleh entitas lain kecuali kelasnya.

```
_priv is private and cannot be accessed from outside the class
Curve and Public Key are private and cannot be updated from outside the class
```

Fig. 8. Tangkapan Layar Pembuktian Enkapsulasi Atribut

2) Registrasi

```
class SmartMeter:
  def __init__(self, sm_id, sp_id):
    self.sm_id = sm_id
    self.public_key = None
    self.sp_id = sp_id

  def generate_key_pair(self, curve):
    private_key = ec.generate_private_key(curve)
    self.public_key = private_key.public_key()
    return private_key

class ServiceProvider:
  def __init__(self, sp_id):
    self.sp_id = sp_id
    self.public_key = None

  def generate_key_pair(self, curve):
    private_key = ec.generate_private_key(curve)
    self.public_key = private_key.public_key()
    return private_key
```

Kelas SmartMeter dan ServiceProvider mewakili entitas yang akan terlibat dalam komunikasi. Mereka memiliki metode generate_key_pair untuk menghasilkan pasangan kunci baru untuk komunikasi. Private key tidak disimpan dalam kelas karena kelas ini akan menjadi objek untuk data pada trust anchor, sehingga hanya menyimpan id dan kunci publik untuk ServiceProvider, dan menyimpan id, kunci publik, id ServiceProvider, dan kunci public ServiceProvider untuk SmartMeter.

```
class DataSMSP:
  def __init__(self):
    self.SM = []
    self.SP = []
```

```
  def regist_SM(self, sm_id):
    # choose avail sp_id
    print("Available SP:")
    for sp in self.SP:
      print(sp.sp_id)
    sp_id = input("Choose sp_id: ")

    sm = SmartMeter(sm_id, sp_id)
    priv = sm.generate_key_pair(params.curve)
    self.SM.append(sm)
    pub_sp = None
    for sp in self.SP:
      if sp.sp_id == sp_id:
        pub_sp = sp.public_key
        break

    return priv, pub_sp

  def regist_SP(self, sp_id):
    sp = ServiceProvider(sp_id)
    priv = sp.generate_key_pair(params.curve)
    self.SP.append(sp)
    return priv
```

Kelas DataSMSP mengelola daftar Smart Meter dan Service Provider. Metode regist_SM memungkinkan pendaftaran Smart Meter dengan memilih Service Provider yang tersedia, sementara regist_SP digunakan untuk mendaftarkan Service Provider baru. Metode register sesuai dengan perancangan skema register pada sistem.

3) Autentikasi dan Pengiriman Pesan

Autentikasi dilakukan di sistem atau file yang terpisah dari trust anchor.

```
sm_id = input("Enter Smart Meter ID: ")
    for sm in data.SM:
      if sm.sm_id == sm_id:
        sp_id = sm.sp_id
        break
    for sp in data.SP:
      if sp.sp_id == sp_id:
        pub_sp = sp.public_key
        break
    if not pub_sp:
      print("Service Provider not found")
      continue
    print("Smart Meter public key:")
    print(sm.public_key.public_bytes())
    print("Service Provider public key:")
    print(pub_sp.public_bytes())
    print("Authenticate Smart Meter to Service Provider")
    print("Smart Meter sign message")
    message = {
      'meter_id': '123456',
      'timestamp': '2020-01-01T00:00:00',
      'power': 1000,
    }
    message_bytes = json.dumps(message).encode('utf-8')
    signature = priv.sign(message_bytes,
ec.ECDSA(hashes.SHA256()))
    print("Smart Meter signature:")
    print(signature)
    print("Service Provider verify signature")
    try:
      pub_sp.verify(signature, message_bytes,
ec.ECDSA(hashes.SHA256()))
    except:
      print("Invalid signature")
      continue
    print("Authenticate success")
```

Setelah SM dan SP terdaftar, SM perlu menghasilkan pesan yang akan dikirimkan ke SP. Pesan tersebut harus dienkripsi menggunakan public key SP. Selain itu, SM juga perlu menandatangani pesan tersebut menggunakan private key-nya. Tanda tangan digital ini akan digunakan oleh SP untuk memverifikasi keaslian pesan. Ketika SP menerima pesan terenkripsi beserta tanda tangan digital, SP perlu mendekripsi pesan tersebut menggunakan private key-nya. Setelah itu, SP menggunakan public key SM yang dipublikasikan oleh Trust Anchor untuk memverifikasi tanda tangan digital. Jika verifikasi berhasil, SP dapat memastikan bahwa pesan tersebut benar-benar berasal dari SM yang sah.

C. Simulasi

Simulasi program berisi tangkapan layar dari terminal untuk setiap keluaran dari program. Pertama, pengguna hanya disediakan 2 menu yaitu inisialisasi parameter sistem pada trust anchor atau keluar dari program.

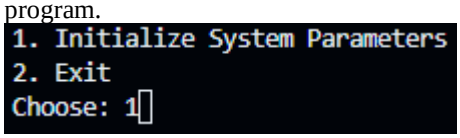

Fig. 9. Tangkapan Layar 1

Jika sistem parameter berhasil digenerate dan disimpan di trust anchor, maka output akan menjadi seperti berikut. Menu yang ditampilkan terdapat register smart meter, register service provider, autentikasi, menampilkan service provider dan smart meter yang tersedia, dan keluar daru program.

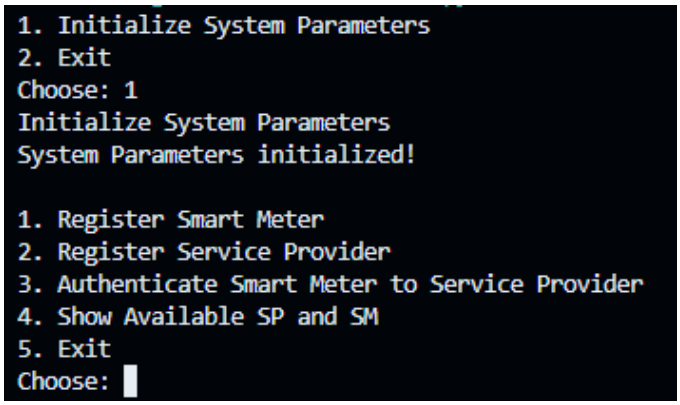

Fig. 10. Tangkapan Layar 2

Berikut adalah keluaran jika pengguna memilih melakukan register service provider.

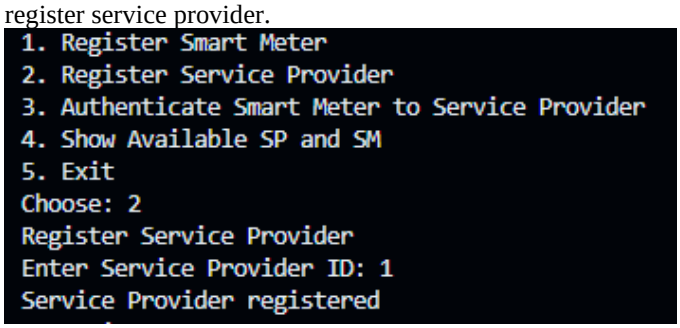

Fig. 11. Tangkapan Layar 3

Berikut adalah keluaran jika pengguna memilih menu untuk register smart meter. Pengguna akan diminta untuk memilih ide service provider yang dituju.

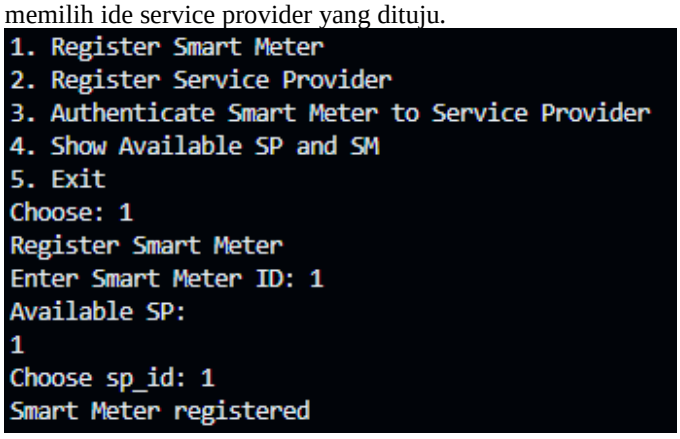

Fig. 12. Tangkapan Layar 4

Selanjutnya, ada opsi autentikasi. Berikut adalah keluaran jika proses autentikasi berhasil.

Fig. 13. Tangkapan Layar 5

Berikut adalah keluaran jika proses autentikasi gagal.

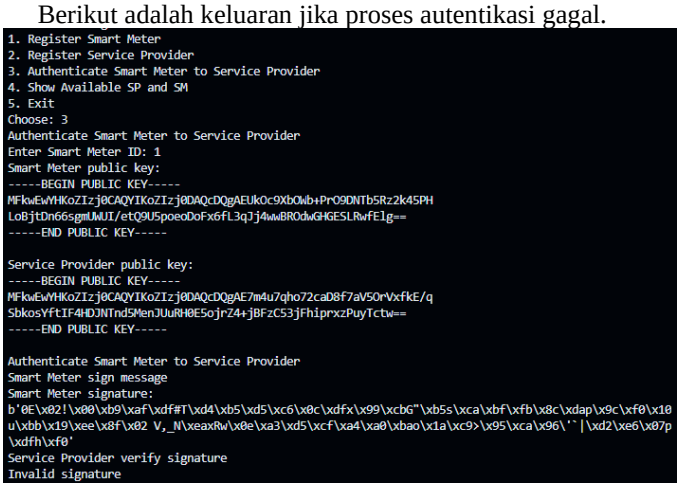

Fig. 14. Tangkapan Layar 6

IV. Analisis dan Evaluasi

Skema autentikasi dan *key management* yang diusulkan memiliki beberapa keunggulan dan kelemahan.

A. *Keunggulan Skema*

1) Centralized Key Storage: Trust Anchor berfungsi sebagai wadah penyimpanan kunci terpusat yang memfasilitasi manajemen kunci yang efisien dan aman dalam sistem. Dengan menyimpan semua kunci publik dan privat secara sentral, sistem dapat lebih mudah mengelola kunci dan mengurangi risiko kehilangan kunci.

2) Public Key Infrastructure (PKI): Penggunaan PKI dalam skema ini memberikan tingkat keamanan komunikasi yang tinggi antara Smart Meter (SM) dan Service Provider (SP). Dengan menggunakan enkripsi pesan dan tanda tangan digital, PKI memastikan integritas dan otentikasi pesan yang dikirimkan antara SM dan SP.

3) Penerbitan Sertifikat dan Verifikasi Identitas: Dengan mempublikasikan nilai hash public key + ID komponen, Trust Anchor memberikan sertifikat yang dapat diverifikasi untuk memastikan identitas yang sah. Hal tersebut akan memperkuat keamanan sistem dengan memastikan bahwa setiap komponen

yang berkomunikasi dalam sistem telah diverifikasi dan diotorisasi.

4) Skema ECC yang Efisien: Algoritma ECC yang digunakan dalam skema ini menawarkan keamanan yang sebanding dengan RSA namun dengan panjang kunci yang lebih pendek. Hal ini mengurangi beban komputasi dan memori yang diperlukan untuk implementasi, membuat skema ini lebih efisien dalam penggunaan sumber daya. Namun, implementasi yang tepat dan pemilihan parameter yang baik dalam algoritma ECC sangat penting untuk menjaga keamanan sistem.

B. Kelemahan Skema

1) Ketergantungan pada Trust Anchor: Kerentanan terbesar terletak pada Trust Anchor, karena walaupun kunci yang disimpan pada trust anchor telah di-hash, namun jika terjadi serangan pada Trust Anchor, proses registrasi dan verifikasi pada tiap komponen dalam satu sistem menjadi terganggu. Jika penyerang mendapatkan akses ke jaringan sistem, penyerang kemungkinan dapat mengetahui kunci privat yang dikirim ke komponen melalui trust anchor setiap adanya proses registrasi.

2) Penggunaan Kunci Terpusat : Penggunaan kunci terpusat pada Trust Anchor dapat menjadi titik kelemahan jika tidak diimplementasikan dengan baik. Karena semua kunci publik dan sertifikat disimpan secara sentral, jika Trust Anchor mengalami kompromi atau serangan, maka seluruh infrastruktur kriptografi dalam sistem dapat terpengaruh. Hal ini dapat menyebabkan akses tidak sah terhadap pesan yang dienkripsi, serta potensi manipulasi atau pemalsuan data. Oleh karena itu, penting untuk memastikan bahwa keamanan Trust Anchor dan infrastruktur kunci publiknya terjaga dengan baik, misalnya dengan mengimplementasikan langkah-langkah keamanan yang kuat seperti penggunaan kunci yang kuat, penggunaan protokol keamanan yang tepat, dan pemantauan serta pembaruan rutin terhadap sistem tersebut.

3) Kompleksitas Implementasi: Algoritma ECC yang digunakan dalam skema ini memiliki tingkat kompleksitas yang tinggi dalam implementasinya, memerlukan pemahaman yang mendalam dan kemungkinan kesulitan dalam pengembangan dan pemeliharaan sistem.

C. Evaluasi

Berdasarkan analisis keunggulan dan kelemahan skema, terdapat beberapa poin evaluasi untuk meningkatkan keamanan pada skema autentikasi dan *key management* yang diusulkan. Poin yang pertama adalah memastikan semua proses pertukaran data dari satu komponen ke komponen lain dilakukan dengan pesan yang terenkripsi. Kedua, diversifikasi keamanan dengan mempertimbangkan penggunaan *multiple trust anchor*s. *Multiple trust anchors* dapat meningkatkan keamanan dengan memperkenalkan redundansi dalam penyimpanan kunci publik. Dengan menggunakan beberapa trust anchors yang berbeda, risiko kerentanan tunggal pada satu trust anchor dapat dikurangi. Jika salah satu trust anchor terpengaruh oleh serangan atau kegagalan sistem, sistem masih dapat beroperasi menggunakan trust anchor lainnya.

Poin ketiga adalah mempertimbangkan skema pembaruan kunci pada 1 periode tertentu untuk mengurangi risiko serangan yang terkait dengan kunci yang telah terungkap atau usang. Pembaruan kunci secara berkala dapat membantu mengurangi risiko jika kunci tersebut terungkap secara tidak sengaja atau disengaja. Dengan memperbarui kunci, sistem dapat mencegah penyerang untuk terus menggunakan kunci yang telah diketahui. Ini penting karena penyerang yang memiliki akses ke kunci privat dapat membahayakan keamanan sistem dengan membaca pesan yang dienkripsi atau bahkan memalsukan tanda tangan digital. Proses pembaruan harus dilakukan dengan terkoordinasi agar tidak mengganggu operasi sistem secara keseluruhan. Selain itu, penting untuk memiliki kebijakan keamanan yang jelas dan terdokumentasi terkait dengan pembaruan kunci dan sertifikat, termasuk prosedur pembaruan, frekuensi pembaruan, dan langkah-langkah mitigasi risiko yang terkait dengan proses pembaruan.

Selain ketiga poin di atas, terdapat poin-poin tambahan lainnya terkait dengan memastikan keamanan lingkungan sebelum implementasi. Sebelum mengimplementasikan skema ini, penting untuk melakukan evaluasi keamanan menyeluruh terhadap lingkungan jaringan dan sistem yang akan digunakan, misalnya, memastikan bahwa jaringan memiliki *firewall* yang kuat untuk melindungi dari serangan luar, menggunakan enkripsi untuk melindungi data yang dikirimkan melalui jaringan, serta memastikan bahwa semua perangkat keras dan perangkat lunak yang digunakan dalam sistem telah diperbarui dengan patch keamanan terbaru. Selain itu, langkah-langkah pengamanan tambahan seperti segmentasi jaringan untuk membatasi akses, implementasi sistem deteksi intrusi untuk mendeteksi serangan yang tidak sah, dan pemantauan keamanan secara terus-menerus juga diperlukan. Penilaian risiko secara berkala juga harus dilakukan untuk mengidentifikasi potensi celah keamanan dan mengambil langkah-langkah yang diperlukan untuk mengurangi risiko tersebut.

Selain mengamankan infrastruktur jaringan, penting juga untuk melindungi data di tingkat aplikasi. Penggunaan protokol enkripsi seperti TLS (*Transport Layer Security*) untuk mengamankan komunikasi antara aplikasi, dan penerapan best practices dalam pengembangan perangkat lunak untuk mencegah kerentanan keamanan seperti *injection attacks* (seperti *SQL injection* dan XSS), juga harus dipertimbangkan.

Dengan mempertimbangkan dan mengimplementasikan langkah-langkah keamanan ini, skema autentikasi dan *key management* dapat ditingkatkan keamanannya dan menjadi lebih tangguh dalam menghadapi ancaman keamanan yang kompleks dan terus berkembang dalam lingkungan smart grid.

TABLE II. TABLE STYLES

Table Head	**Table Column Head**		
	Table column subhead	***Subhead***	***Subhead***
copy	More table copy[a]		

[a.] Sample of a Table footnote. *(Table footnote)*

b.

Fig. 15. Example of a figure caption. *(figure caption)*

REFERENCES

[1] S. Khasawneh and M. Kadoch, "Hybrid cryptography algorithm with precomputation for advanced metering infrastructure networks", *Mobile Netw. Appl.*, vol. 23, no. 4, pp. 982-993, 2018.

[2] N. Liu, J. Chen, L. Zhu, J. Zhang and Y. He, "A Key Management Scheme for Secure Communications of Advanced Metering Infrastructure in Smart Grid," in IEEE Transactions on Industrial Electronics, vol. 60, no. 10, pp. 4746-4756, Oct. 2013, doi: 10.1109/TIE.2012.2216237

[3] H. Nicanfar, P. Jokar, K. Beznosov and V. C. M. Leung, "Efficient Authentication and Key Management Mechanisms for Smart Grid Communications," in IEEE Systems Journal, vol. 8, no. 2, pp. 629-640, June 2014, doi: 10.1109/JSYST.2013.2260942.

[4] Mahmood, K., Chaudhry, S. A., Naqvi, H., Kumari, S., Li, X., & Sangaiah, A. K. (2018). An elliptic curve cryptography based lightweight authentication scheme for smart grid communication. *Future Generation Computer Systems*, *81*, 557-565.

[5] C. Varma, "A Study of the ECC, RSA and the Diffie-Hellman Algorithms in Network Security," 2018 International Conference on Current Trends towards Converging Technologies (ICCTCT), Coimbatore, India, 2018, pp. 1-4, doi: 10.1109/ICCTCT.2018.8551161.

[6] GeeksforGeeks. (2020, February 4). Galois Fields and Its Properties. GeeksforGeeks. https://www.geeksforgeeks.org/galois-fields-and-its-properties/

[7] Commission, E. (2011). Communication from the commission to the european parliament, the council, the european economic and social committee and the committee of the regions smart grids: From innovation to deployment COM(2011) 202. Technical Report. European Commission.

[8] D. Abbasinezhad-Mood and M. Nikooghadam, "An Anonymous ECC-Based Self-Certified Key Distribution Scheme for the Smart Grid," in IEEE Transactions on Industrial Electronics, vol. 65, no. 10, pp. 7996-8004, Oct. 2018, doi: 10.1109/TIE.2018.2807383.

[9] P. Naura, “nauravalda/smart-grid-auth-scheme,” GitHub, Jun. 2024. https://github.com/nauravalda/smart-grid-auth-scheme

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, 12 Juni 2024

Naura Valda Prameswari (18221173)