Implementasi Enkripsi dan Dekripsi Data Sensitif pada SharedPreferences Aplikasi Android menggunakan ECC dan AES

Edwin Rachman (NIM 13515042)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13515042@std.stei.itb.ac.id

***Abstrak*— Terkadang dalam proses pengembangan aplikasi mobile di platform Android, dibutuhkan sebuah media penyimpanan data sensitif seperti password, nomor kartu kredit, dll. Salah satu media penyimpaman lokal utama yang dapat digunakan oleh aplikasi Android adalah SharedPreferences. SharedPreferences untuk penggunaan lazim relatif aman karena hanya dapat dibaca oleh aplikasi dengan UID tertentu. Tetapi untuk kasus data yang sensitif SharedPreferences tidak aman sama sekali karena datanya dapat dengan mudah diakses oleh penyerang dengan root access. Oleh karena itu, dibutuhkan sebuah metode enkripsi dan dekripsi data yang aman untuk SharedPreferences, salah satunya adalah dengan menggunakan Advanced Encryption Standard (AES). Kunci privat dari AES kemudian dienkripsi menggunakan Elliptic-Curve Cryptography (ECC) sehingga membentuk sebuah sistem kriptografi hybrid. Kunci privat ECC yang digunakan oleh implementasi disimpan dalam Android Keystore System, yang merupakan sebuah sistem container kunci kriptografik yang sangat aman yang disediakan oleh API Android sejak versi 23. Metode enkripsi dan dekripsi ini diimplementasikan menggunakan sebuah kelas turunan khusus untuk SharedPreferences dimana setiap method set mengenkripsi value yang akan dimasukkan terlebih dahulu sebelum memasukkannya ke SharedPreferences, dan setiap method get mendekripsi value yang didapatkan terlebih dahulu sebelum mengembalikan valuenya ke pengguna.**

Kata kunci—Android; SharedPreferences; ECC; AES; cryptography

I. PENDAHULUAN

Zaman sekarang, data merupakan hal yang sangat bernilai, terutama data-data yang bersifat sensitif seperti password, nomor kartu kredit, dll. Data-data sensitif tersebut penting untuk diamankan karena jika jatuh pada tangan yang tidak berwewenang maka mereka dapat dengan mudah mengeksploitasi data sensitif tersebut dan menyebabkan kerugian-kerugian yang sangat besar (baik secara materil seperti uang, atau imateril seperti pencurian identitas) bagi pengguna perangkat lunak. Akan tetapi, sering keamanan dari data-data tersebut tidak terlalu diperhatikan oleh pengembang-pengembang aplikasi perangkat lunak karena kemalasan, kelalaian, atau ketidaktahuannya akan pentingnya keamanan data.

Oleh karena itu, keamanan data merupakan hal yang sangat perlu untuk diperhatikan oleh pengembang-pengembang perangkat lunak. Hal tersebut lebih perlu diperhatikan oleh pengembang-pengembang aplikasi *mobile*, yang karena sifatnya yang dapat dibawa kemana-mana membuatnya lebih rentan lagi untuk dicuri dan dibobol oleh *hacker* untuk mendapatkan informasi pengguna aplikasi *mobile* tersebut.

Salah satu metode penyimpanan data yang paling sering digunakan pada sistem operasi mobile Android adalah SharedPreferences, yang merupakan sebuah penyimpanan key-value yang sering digunakan oleh pengembang aplikasi Android untuk menyimpan data-data *preference* dari pengguna secara offline. Salah satu data *preference* yang sering disimpan adalah nomor kartu kredit, yang merupakan contoh dari data sensitif. Akan tetapi, SharedPreferences disimpan secara plainteks sebagai sebuah file XML. Untuk kasus-kasus lazim, penyimpanan seperti itu sudah cukup aman karena file tersebut hanya diakses oleh aplikasi dengan UID yang terasosiasi dengan SharedPreferences tersebut. Tetapi, jika seseorang berhasil mencuri *device* pengguna dan membobolnya dengan root access, maka ia dapat dengan mudah mengakses file data tersebut.

Untuk menyelesaikan masalah tersebut, dibutuhkan cara untuk mengamankan data yang terdapat pada SharedPreferences, yaitu dengan membuat sebuah kelas turunan khusus dari SharedPreferences yang lebih aman. Hal tersebut dapat dilakukan dengan cara melakukan override terhadap method-method set agar melakukan enkripsi value terlebih dahulu sebelum memasukkannya ke SharedPreferences sebenarnya, dan juga pada method-method get agar melakukan dekripsi value terlebih dahulu sebelum dikembalikan ke pengguna. Sistem kriptografi yang akan digunakan adalah sistem kriptografi hybrid, dimana kunci enkripsi yang digunakan adalah sebuah Advanced Encryption Standard (AES), yang kemudian dienkripsi menggunakan Elliptic-Curve Cryptography (ECC). Kunci ECC yang digunakan oleh implementasi akan disimpan dalam Android Keystore System, yang merupakan sebuah sistem container kunci kriptografik yang sangat aman yang disediakan oleh API Android sejak versi 23.

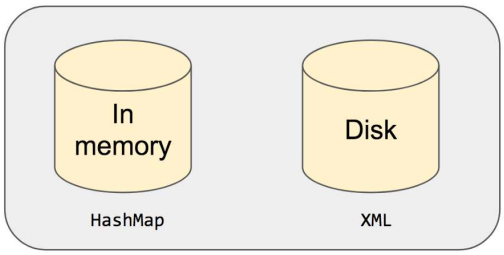
II. DASAR TEORI

A. *SharedPreferences*

SharedPreferences adalah sebuah tempat penyimpanan key-value yang tersedia pada ekosistem pengembangan aplikasi pada Android. *SharedPreferences* digunakan untuk mengakses dan memodifikasi data untuk sebuah aplikasi tertentu. [1]

SharedPreferences secara internal menggunakan *in-memory storage* di atas *disk storage*. Setiap operasi dilakukan pada *in-memory storage* terlebih dahulu, baru kemudian pada *disk storage* jika diperlukan. *In-memory storage* pada dasarnya adalah sebuah HashMap, sehingga waktu setiap operasi hanya memiliki kompleksitas waktu O(1). *Disk storage* disimpan dalam bentuk file xml yang terstruktur sebagai berikut.

```
<?xml version=1.0 encoding='utf-8' standalone='yes'?>
<map>
    <string name="KEY">value</string>
</map>
```



Gambar 1. Struktur internal SharedPreferences

File xml tersebut disimpan pada data/data/<APP ID APLIKASI>/shared_prefs/<NAMA SHAREDPREFERENCES>.xml. File tersebut dapat dilihat di Device File Explorer pada Android Studio atau dengan program utilitas adb. [2]

B. *Elliptic Curve Cryptography* (ECC)

Elliptic Curve Cryptography (ECC) adalah algoritma kriptografi kunci publik yang didasarkan pada struktur aljabar dari kurva eliptik pada *finite field*. ECC dikembangkan oleh Neal Koblitz dan Victor S. Miller pada tahun 1985.

Alasan mengapa kurva eliptik digunakan untuk kriptografi kunci public adalah karena menemukan logaritma diskrit dari sebuah kurva eliptik yang acak sangatlah sulit, yang disebut sebagai Elliptic Curve Discrete Logarithm Problem (ECDLP). Keamanan dari ECC tergantung pada kemampuan menghitung perkalian titik pada kurva, dan ketidakmampuan untuk menghitung pengalinya jika mengetahui titik awal dan hasil produknya.

ECC membutuhkan beberapa parameter domain, yaitu:

- p = field dimana kurva didefinisikan
- a, b = konstanta yang mendefinisikan kurvanya
- G = titik generator
- n = prime order dari G
- h = cofactor

Kelebihan dari ECC adalah panjang kuncinya yang lebih pendek daripada kunci RSA, tetapi memiliki tingkat keamanan yang sama dengan RSA. Misal, kunci ECC sepanjang 160-bit menyediakan keamanan yang sama dengan kunci RSA 1024-bit. Oleh karena itu, ECC baik digunakan untuk piranti nirkabel, dimana prosesor, memori, dan umur baterai terbatas. [3]

C. *Advanced Encryption Standard* (AES)

Advanced Encryption Standard (AES) adalah subset dari cipher Rijndael yang dikembangkan oleh Vincent Rijmen dan Joan Daemen, yang mengajukan proposal dari cipher tersebut ke NIST untuk seleksi standar enkripsi yang baru setelah DES. [4]

AES merupakan sebuah jaringan substitution-permutation, tidak seperti DES yang menggunakan jaringan Feistel. AES memiliki block size yang fixed berukuran 128-bit, dan key size berukuran 128, 192, dan 256-bit.

Algoritma AES adalah sebagai berikut:

- KeyExpansions menggunakan key schedule Rijndael.
- Round pertama: AddRoundKey
- Round selanjutnya: SubBytes, ShiftRows, MixColumns, AddRoundKey
- Round terakhir: SubBytes, ShiftRows, AddRoundKey

III. IMPLEMENTASI DAN HASIL

A. Implementasi

Untuk melakukan pengamanan dari data sensitif yang akan disimpan menggunakan SharedPreferences, maka akan diimplementasikan sebuah kelas turunan dari SharedPreferences dengan nama SecureSharedPreferences.

Sebelum proses enkripsi/dekripsi dapat dilakukan, pengguna kelas SecureSharedPreferences harus terlebih dahulu memanggil fungsi generateKey(), yang melakukan pembangkitan kunci ECC yang disimpan pada Android KeyStore.

Kelas SecureSharedPreferences melakukan override atas method-method get berikut:

- getAll(): get semua entry
- getBoolean(): get sebuah entry bertipe Boolean
- getFloat(): get sebuah entry bertipe Float
- getInt(): get sebuah entry bertipe Int
- getLong(): get sebuah entry bertipe Long
- getString(): get sebuah entry bertipe String
- getStringSet(): get sebuah entry bertipe Set<String>

Hal yang dilakukan oleh override metode-metode tersebut adalah melakukan dekripsi nilai valuenya terlebih dahulu sebelum dikembalikan ke pengguna. Kunci ECC yang terdapat pada Android KeyStore digunakan untuk melakukan dekripsi pada segment enkapsulasi kunci AES pada nilai valuenya. Setelah kunci AES didapatkan, maka kunci tersebut digunakan untuk dekripsi segment enkapsulasi datanya. Hasil dari dekripsi tersebut adalah yang dikembalikan ke pengguna.

Kelas SecureSharedPreferences melakukan override atas method edit() sehingga mengembalikan kelas SecureSharedPreferences.Editor, yang merupakan kelas turunan dari Editor SharedPreferences biasa.

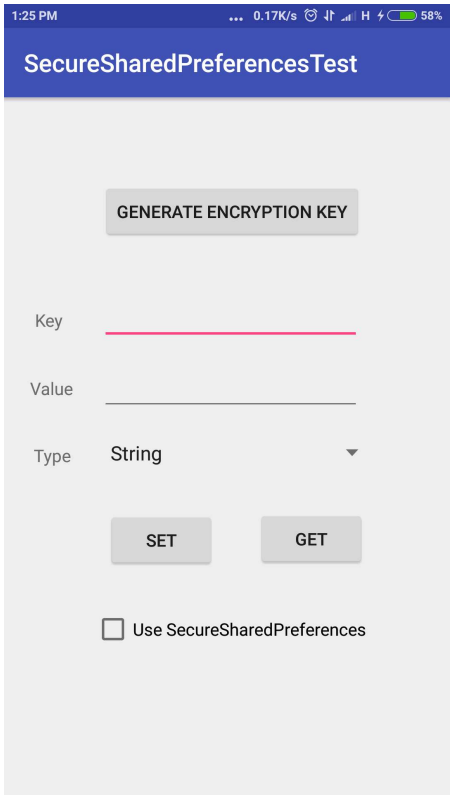
Kelas SecureSharedPreferences.Editor melakukan override atas method-method put berikut:

- putBoolean(key, value): put sebuah entry bertipe Boolean
- putFloat(key, value): put sebuah entry bertipe Float
- putInt(key, value): put sebuah entry bertipe Int
- putLong(key, value): put sebuah entry bertipe Long
- putString(key, value): put sebuah entry bertipe String
- putStringSet(key, value): put sebuah entry bertipe Set<String>

Hal yang dilakukan oleh override metode-metode tersebut adalah melakukan enkripsi nilai valuenya terlebih dahulu sebelum dimasukkan ke SharedPreferences. Pertama dilakukan generasi kunci simetrik AES untuk dimasukkan pada segment enkapsulasi kuncinya. Kemudian, nilai value sebenarnya dienkripsi dengan kunci simetrik AES tersebut. Setelah itu, segment enkapsulasi kuncinya dienkripsi menggunakan kunci ECC pada Android KeyStore. Gabungan dari segment enkapsulasi kunci dan data tersebut adalah yang disimpan pada SharedPreferences (dalam base64).

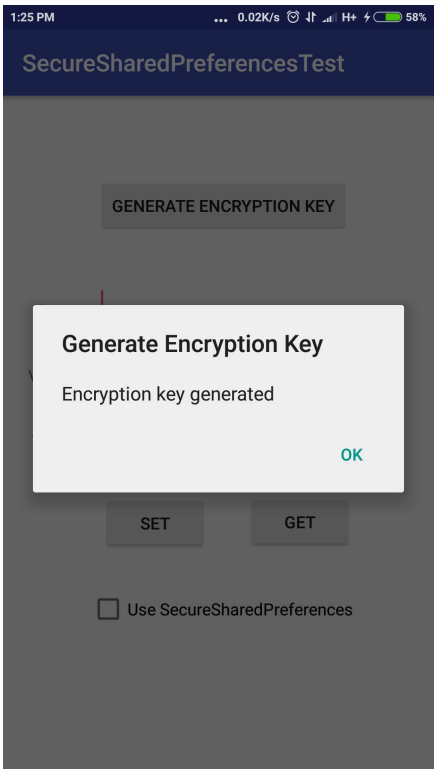
B. Hasil Implementasi

Untuk melakukan pengujian atas kelas SecureSharedPreferences, telah dibuat sebuah aplikasi pengujian sederhana. Aplikasi tersebut memiliki tampilan utama sebagai berikut:



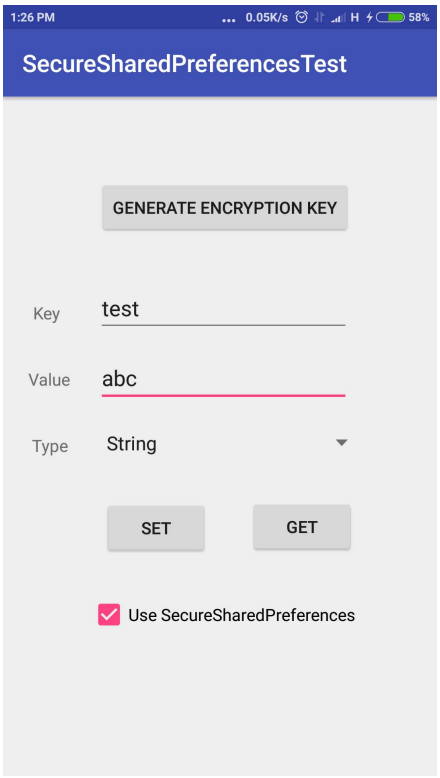
Gambar 2. Tampilan antarmuka aplikasi awal

Sebelum SecureSharedPreferences dapat digunakan, pengguna terlebih dahulu membangkitkan sebuah key enkripsi dari sistem kriptografi hybrid ECC-AES yang akan digunakan dengan klik tombol “Generate Encryption Key”. Berikut adalah tampilan setelah pengguna klik tombol tersebut:



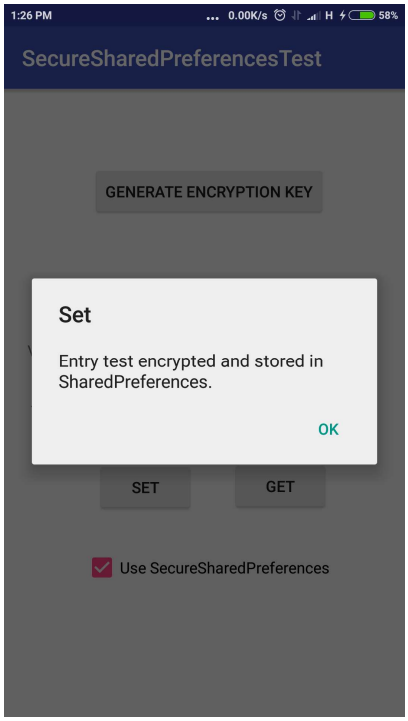
Gambar 3. Tampilan antarmuka aplikasi setelah tombol "Generate Encryption Key" diklik

Setelah itu, pengguna dapat mengisi pasangan key-value dan tipe data value yang akan disimpan. Checkbox "Use SecureSharedPreferences" dicentang supaya yang digunakan adalah SecureSharedPreferenes, bukan SharedPreferences biasa.



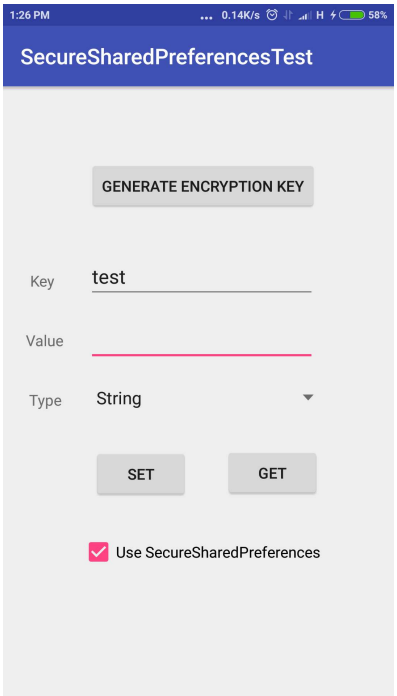
Gambar 4. Tampilan antarmuka aplikasi setelah pengguna mengisi pasangan key-value dan tipe.

Pengguna kemudian klik tombol "Set" untuk menyimpan pasangan key-value tersebut.



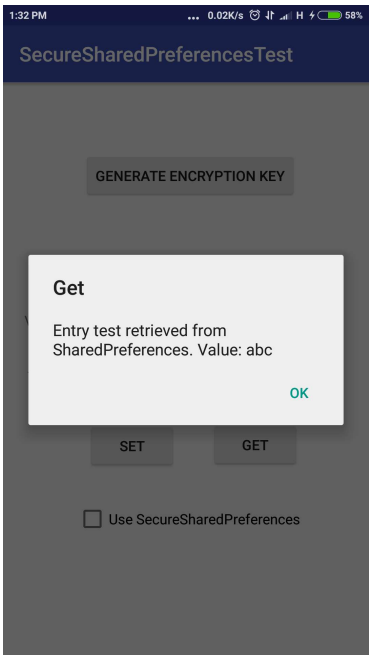
Gambar 5. Tampilan antarmuka aplikasi setelah pengguna klik tombol “Set”.

Untuk melakukan operasi get, pengguna cukup mengisi Key dan Type-nya saja.



Gambar 6. Tampilan antarmuka aplikasi setelah pengguna mengisi key dan tipe.

Pengguna kemudian klik “Get” untuk melakukan operasi get.



Gambar 8. Tampilan antarmuka aplikasi setelah pengguna klik tombol “Get”.

IV. Analisis

A. Analisis Enkripsi

Berikut adalah file XML SharedPreferences yang tidak menggunakan enkripsi.

```
<?xml version=1.0 encoding='utf-8'
standalone='yes'?>
<map>
    <string name="test">abc</string>
</map>
```

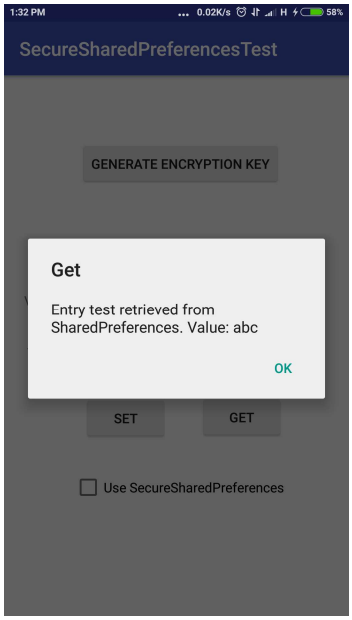
Berikut adalah file XML SharedPreferences yang menggunakan enkripsi via SecureSharedPreferences.

```
<?xml version=1.0 encoding='utf-8'
standalone='yes'?>
<map>
    <string name="test">
afxIMDUDrawmCbvpomoBcxjZpTxpFpKpBPqoooq
jmD0=</string>
</map>
```

Dapat dilihat bahwa nilai plainteks “abc” telah dienkripsi menjadi nilai base-64 yang terlihat di atas untuk kasus yang menggunakan SecureSharedPreferences.

B. Analisis Dekripsi

File yang sebelumnya, dapat didekripsi dengan baik dan menghasilkan tampilan.



Gambar 9. Tampilan aplikasi hasil dekripsi

Jika misalnya file SharedPreferences diubah sedikit bagian valuenya menjadi sebagai berikut:

```
<?xml version=1.0 encoding='utf-8'
standalone='yes'?>
<map>
    <string name="test">
bfxIMDUDrawmCbvpomoBcxjZpTxpFpKpBPqoooq
jmD0=</string>
</map>
```

Maka aplikasi akan menampilkan pesan error yang menyatakan bahwa operasi dekripsi gagal.

C. Analisis *Cryptosystem*

Penggunaan ECC untuk *hybrid cryptosystem* dapat dilakukan, tetapi tidak terlalu memberikan keuntungan untuk enkripsi pada sistem operasi Android. Hal ini disebabkan oleh *space* Android KeyStore yang cukup besar dan tidak terlalu membutuhkan keuntungan dari ECC yaitu keysize yang kecil.

V. Kesimpulan dan Saran

A. Kesimpulan

Kesimpulan yang dapat diperoleh dari makalah ini berdasarkan uraian yang telah dijelaskan pada keempat bab sebelumnya adalah sebagai berikut:

- Implementasi SecureSharedPreferences yang merupakan SharedPreferences yang terenkripsi berhasil dilakukan.
- Proses enkripsi entry SharedPreferences dengan SecureSharedPreferences berhasil dilakukan.
- Proses dekripsi entry SharedPreferences dengan SecureSharedPreferences berhasil dilakukan.
- Penggunaan ECC untuk hybrid cryptosystem ini sebenarnya tidak terlalu cocok karena keuntungan dari ECC tidak dipergunakan dengan baik.

B. Saran

Dalam penulisan makalah ini, terdapat beberapa saran-saran yang diperoleh untuk perkembangan yang lebih lanjut mengenai bidang kriptografi:

- Disarankan kepada para pengembang aplikasi Android yang ingin melalukan hal yang serupa, yaitu mengamankan SecurePreferences, untuk coba menggunakan RSA dalam implementasinya.
- Disarankan kepada para expert dan pengajar untuk mendidik pengembang perangkat lunak yang diajarnya agar lebih memperhatikan pentingya dari keamanan data sensitif pada perangkat lunak.

Ucapan Terima Kasih

Saya mengucapkan terima kasih kepada dosen pengampu mata kuliah IF4020 – Kriptografi, bapak Rinaldi Munir, karena telah mengajar dan membimbing kami selama satu semester ini. Saya juga mengucapkan terima kasih kepada teman-teman seangkatan yang secara langsung dan tidak langsung membantu saya dalam proses pembelajaran saya dalam mata kuliah ini.

Daftar Pustaka

[1] Android Developers. “SharedPreferences”. [Online]. https://developer.android.com/reference/android/content/SharedPreferences

[2] Obut, Orhan. 2018. “Android 101 : Shared Preferences” [Online]. https://proandroiddev.com/shared-preferences-101-aee26c13e4

[3] Standards for Efficient Cryptography Group (SECG). 2000. “SEC 1: Elliptic Curve Cryptography”. [Online]. http://www.secg.org/sec1-v2.pdf

[4] US National Institute of Standards and Technology. 2001. [Online]. http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

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, 18 Mei 2018

Edwin Rachman
NIM 13515042