Autentikasi Kepemilikan *File* dengan *Elliptic Curve Cryptography*

Kevin Erdiza Yogatama, 13515016[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13515016@std.stei.itb.ac.id

***Abstract*—Masalah kepemilikan data dalam Internet merupakan masalah yang memiliki banyak kepentingan. Di makalah ini, diusulkan sebuah kerangka kerja yang mengatasi masalah tersebut menggunakan Elliptic Curve Cryptography. Kerangka ini memungkinkan sebuah entitas untuk membuat tanda kepemilikan yang keabsahannya bisa dibuktikan oleh pihak manapun hanya dengan menggunakan *public key*.**

***Keywords*—Tanda Kepemilikan *File*, *Digital Signature*, *Elliptic Curve Cryptography*.**

I. Pendahuluan

Perkembangan Internet saat ini telah membuat perpindahan dan penyalinan data antar perangkat menjadi sangat mudah dan hampir tidak terbatas. Aspek tersebutlah yang membuat setiap informasi jadi lebih mudah diakses dan komunikasi jadi tidak terbatas jarak dan waktu. Namun hal tersebut membuat tidak adanya data yang unik dan menjadi kaburnya kepemilikan sebuah data pada internet. Kepemilikan sebuah data pada internet menjadi salah satu riset keamanan *file* yang masih berkembang sampai saat ini.

Pada tulisan ini, Diusulkan sebuah kerangka kerja yang memungkinkan autentikasi kepemilikan sebuah *file* untuk suatu individu tertentu. Kerangka kerja ini memanfaatkan dan mengembangkan teknik *Digital Signature* yang menggunakan *Elliptic Curve Cryptography*.

II. Dasar Teori

A. Elliptic Curve Cryptography

Elliptic Curve Cryptography merupakan sebuah teknik kriptografi yang memanfaatkan sifat asimetri dari suatu pasangan operasi pada matematika kurva eliptik. Sifat asimetri yang dimaksud adalah terdapatnya suatu operasi mudah namun untuk operasi sebalikya merupakan operasi yang sulit komputasinya. Dengan memanfaatkan sifat tersebut, proses kriptografi bisa mudah menghasilkan nilai tertentu namun nilai tersebut akan sulit dihitung mundur untuk mendapatkan nilai sebelumnya.

Pada matematika kurva eliptik, sifat asimetri ditemukan pada kurva eliptik dengan domain nilai yang *finite*. Kurva eliptik merupakan kurva yang dibentuk dari persamaan berikut:

$$\{ y^2 = x^3 + ax + b,\ 4a^3 + 27b^2 \neq 0 \mid (x, y) \in R^2 \} \cup \{ 0 \}$$

Terdapat sejumlah operasi yang bisa dilakukan terhadap titik yang dihasilkan oleh sebuah kurva eliptik. Dua dari operasi tersebut adalah pasangan operasi *scalar multiplication* dan operasi *logarithm*.

1. *Scalar Multiplication*
   Operasi *scalar multiplication* pada matematika kurva eliptik adalah berlakunya persamaan berikut:

   $$nP = P + P + ... + P (n\ kali)$$

   Yang berlaku untuk setiap titik P pada kurva eliptik dan untuk nilai integer *n* positif apapun.

2. *Logarithm*
   Sedangkan operasi *logarithm* adalah operasi kebalikan dari operasi *scalar multiplication*, dimana untuk persamaan $Q = nP$ dimana Q dan P diketahui, nilai n yang dicari.

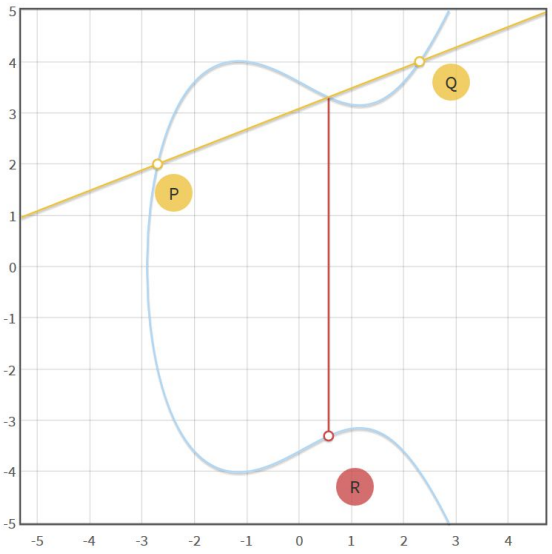

Diagram 1: Kurva eliptik dengan a = -4 dan b = 13

Dalam *Elliptic Curve Cryptography,* sepasang operasi tersebut akan memiliki sifat asimetri jika domain nilai untuk kurva eliptik dibatasi untuk nilai yang *finite*. Titik yang dihasilka kurva eliptik tersebut dapat dinotasikan sebagai berikut.

$$\{ y^2 = x^3 + ax + b \,(mod\, p),\ 4a^3 + 27b^2 \neq 0 \,(mod\, p)\ ,$$
$$|\ p \in bilangan\ prima,\ (x, y) \in F^2 \} \cup \{ 0 \}$$

Titik dari kurva eliptik pada domain yang *finite* tersebut masih dapat dioperasikan dengan operasi yang sama dengan operasi yang bisa dilakukan pada kurva eliptik domain nilai *real*. Dan pada kurva tersebutlah, operasi *scalar multiplication* adalah operasi yang mudah, sedangkan operasi *logarithm* adalah operasi yang sulit; sifat asimetri yang dimanfaatkan oleh *Elliptic Curve Cryptography*.

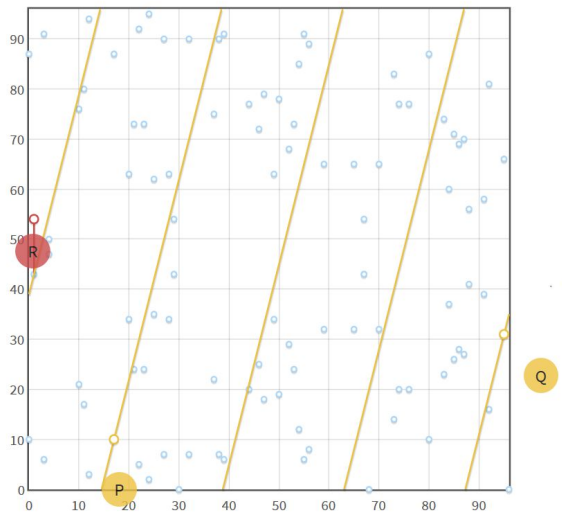

Diagram 2 : Titik titik yang dibentuk kurva eliptik domain nilai *finite* dengan a = 2, b = 3, p = 97

B. Parameter dari Kurva Elliptik Domain Nilai Finite

Setiap sistem kriptografi menggunakan kurva eliptik yang unik untuk membedakan satu sama lain. Kurva eliptik diidentifikasin dengan 6 parameter dalam bentuk sextuplet (p, a, b, G, n, h). keenam parameter tersebut adalah nilai prima *p,* nilai koefisien *a* dan *b*. Titik dasar subgrup kurva eliptik *G*, nilai *order n* dari subgrup, dan nilai *cofactor* h dari subgrup.

Nilai *p*, *a*, *b* adalah nilai variabel yang terdapat pada persamaan kurva eliptik pada domain nilai *finite* yang dijelaskan di bagian sebelumnya. Ketiga nilai ini membentuk sekumpulan titik-titik kurva eliptik yang unik.

Titik *G* merupakan titik yang terdapat pada kurva eliptik. Titik ini dipilih setelah kurva eliptik (nilai *p, a, b)* ditentukan. Titik G yang dipilih akan membentuk sebuah subgrup titik dengan karakteristik nilai *order n* dan nilai *cofactor h*. Titik *G* yang ideal dipilih adalah titik *G* yang menghasilkan subgrup dengan nilai *order* yang tinggi.

Untuk kriptografi, *private key d* pada kurva eliptik daomain nilai *finite* adlah bilanga *random* dari rentang *{ 1, ..., n - 1 }* sedangkan *public key* didapatkan dari perhitungan *H = dG*. Hal ini masuk akal dalam matematika kurva eliptik daomain nilai *finite*, karena operasi *scalar multiplication* untuk mendapat nilai *H* adalah mudah, sedangkan masalah *logarithm* untuk mendapatkan nilai *d* adalah sulit.

C. Elliptic Curve Digital Signature Algorithm (ECDSA)

Salah satu teknik yang memakai kriptografi kurva eliptik

adalah *Elliptic Curve Digital Signature Algorithm (ECDSA)*. Teknik ini memungkinkan siapapun untuk mengecek keabsahan *signature* yang dipasangkan kepada sebuah data. Ada 2 proses yang dilakukan pada ECDSA: proses pembuatan *signature* dan verifikasi *signature*

Pada proses pembuatan *signature*. Dengan menggunakan seluruh parameter kurva, nilai *hash* data (z) yang ingin di *sign,* dan *private* (*d)* milik pihak yang ingin men-*sign*, Langkah yang dilakukan adalah sebagai berikut:

1. Pilih nilai bilangan bulat *random k* dari rentang *{ 1, …, n - 1 }*
2. Hitung titik P = kG
3. Dengan P = (x_p, y_p) , hitung $r = x_p \bmod n$
4. Jika r = 0, kembali ke langkah 1
5. Hitung nilai $s = k^{-1}(z + rd) \bmod n$
6. Jika s = 0, kembali ke langkah 1

Langkah perhitungan tersebut akan menghasilkan sepasang nilai *(r , s)* yang akan digunakan sebagai *signature* data yang di-*sign*.

Untuk verifikasi, dengan menggunakan *public key (H)* pihak yang men-*sign*, nilai *hash* data (z) yang di-*sign* , dan nilai *(r , s)*, langkah yang dilakukan adalah berikut:

1. Menghitung nilai $u_1 = s^{-1}z \bmod n$
2. Menghitung nilai $u_2 = s^{-1}r \bmod n$
3. Menghitung nilai $P = u_1G + u_2H$

Dimana, dengan nilai P = (x_p, y_p), *signature* diterima jika $r = x_p \bmod n$.

Operasi ECDSA memungkinkan siapapun untuk melakukan pengecekan keabsahan *signature* karena nilai yang dibutuhkan untuk kalkulasi tidak ada yang tersembunyi. Sifat inilah yang difokuskan pada kerangka kerja yang diusulkan makalah ini dengan mengembangkan algoritma ECDSA ini.

III. Rancangan Solusi

A. Kerangka Kerja

Kerangka kerja yang diusulkan memiliki struktur yang digambar dengan diagram berikut:

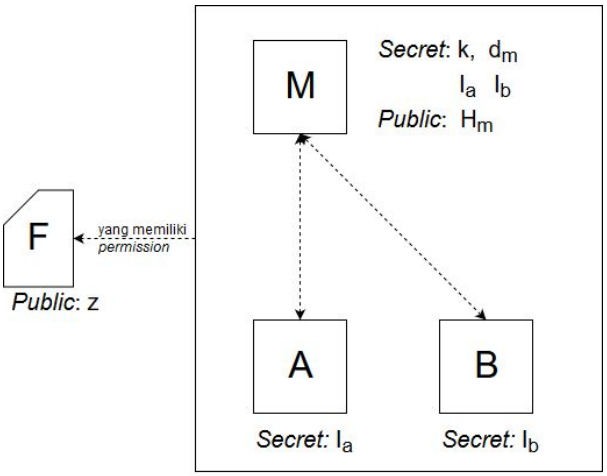

Diagram 3 : Kerangka umum

Diagram tersebut menggambarkan sebuah lingkungan yang terdiri sebuah entitas *master* M yang memberikan tanda kepemilikan untuk akses dan entitas *user* A dan B (dan bisa lebih) yang memiliki hak untuk memiliki sebuah *file* yang tersedia.

Entitas *master* memiliki nilai *k* dan d_m yang rahasia hanya untuk entitas *master* dan nilai H_m yang publik diakses siapapun di lingkungan tersebut. Nilai d_m dan H_m merupakan pasangan *private-public key* untuk *Elliptic Curve Cryptography*. Entitas *master* memiliki akses penuh ke *file* yang tersedia dan dapat memberikan tanda kepemilikan *file* ke entitas yang ada di lingkungan tersebut dengan cara yang akan dijelaskan di bagian selanjutnya. Entitas *master* juga mengetahui nilai identitas yang dimiliki setiap entitas *user*.

Entitas *user* memiliki nilai identitas *I* yang rahasia hanya untuk entitas itu sendiri dan entitas *master*. Entitas *user* secara *default* tidak bisa mengakses *file* yang tersedia namun bisa meminta tanda kepemilikan dari entitas *master* mendapatkan akses.

Entitas terakhir adalah entitas *file* yang tersedia, yang memiliki nilai z yang merupakan nilai *hash* dari *file* tersebut. Sebuah *file* hanya bisa diakses oleh yang memiliki tanda kepemilikan *file* namun nilai z dari *file* tersebut publik, dimana nilai z digunakan untuk prosedur permintaan tanda kepemilikan.

Di dalam kerangka kerja ini terdapat 2 proses: pembuatan tanda kepemilikan dan Autentikasi/Verifikasi Tanda Kepemilikan.

B. Pembuatan Tanda Kepemilikan

Pembuatan Tanda Kepemilikan akan dilakukan jika sebuah entitas *user* meminta tanda kepemilikan untuk suatu *file* kepada entitas *master*. Langkah yang dilakukan dalam proses ini adalah berikut:

1. Entitas *user* memberitahukan *file* mana yang ingin diakses kepada entitas *master*
2. Entitas *master* akan menghitung nilai hash dari gabungan dua nilai: nilai z dari *file* yang diminta dan nilai I dari user tersebut.
3. Entitas *master* melakukan proses *signing* yang langkahnya serupa dengan *signing* dengan ECDSA, namun bekerja terhadap nilai z yand dihitung di langkah 2. Proses ini akan menghasilkan nilai (r , s)
4. Nilai (r , s) akan dikirim kembali ke entitas *user* yang meminta tanda kepemilikan, dimana nilai tersebut akan digunakan untuk pembuktian tanda kepemilikan yang dijelaskan di bagian berikutnya.

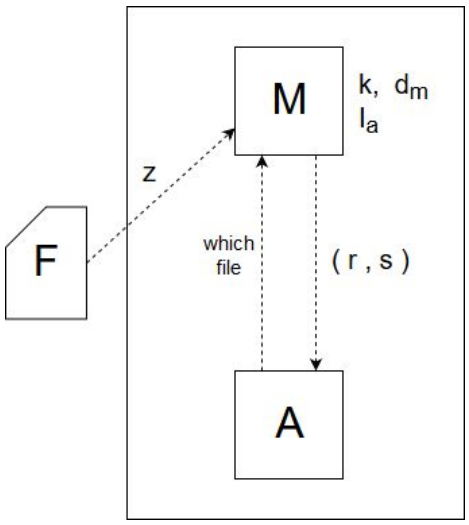

diagram 4 : diagram perpindahan data dalam proses pembuatan tanda kepemilikan

C. Autentikasi/Verifikasi Tanda Kepemilikan

Verifikasi Tanda Kepemilikan dilakukan untuk mengecek apakah sebuah entitas *user* benar-benar memiliki tanda kepemilikan untuk melakukan akses. Hal ini yang dilakukan adalah berikut:

1. Pihak yang ingin mengverifikasi meminta entitas *user* yang ingin dicek untuk menghitung nilai hash dari gabungan nilai z *file* dimiliki dan nilai I dari *user* yang dicek.
2. Pihak yang ingin mengverifikasi juga akan meminta nilai H_m dari entitas *master*.
3. Nilai *hash* z yang dihasilkan, nilai (r, s) yang dimiliki entitas *user* yang dicek, dan nilai H_m akan dipakai untuk verifikasi dengan langkah yang serupa dengan verifikasi ECDSA. Langkah verifikasi akan menghasilkan titik P
4. Sebuat entitas *user* terbukti memiliki tanda kepemilikan *file* jika nilai x dari point P yang dihasilkan membenarkan persamaan $r = x_p \bmod n$.

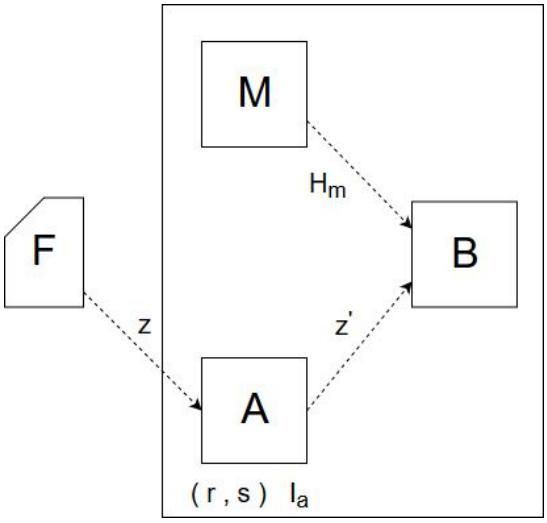

diagram 5 : diagram perpindahan data dalam proses verifikasi tanda kepemilikan, dengan contoh entitas B sebagai pihak yang ingin mengecek keabsahan tanda kepemilikan.

IV. KESIMPULAN

Di makalah ini, telah diusulkan sebuah kerangka kerja yang menyediakan solusi untuk pembuatan tanda kepemilikan file yang dibuat oleh satu entitas *master*. Kerangka tersebut juga memungkinkan pihak manapun untuk melakukan pengecekan keabsahan.

REFERENCES

[1] https://andrea.corbellini.name/2015/05/30/elliptic-curve-cryptography-ecdh-and-ecdsa/ diakses pada tanggal 27 Mei 2019.

[2] https://andrea.corbellini.name/ecc/interactive/modk-add.html diakses pada tanggal 26 Juni 2019.