UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE INFORMÁTICA

GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MARCOS VINÍCIOS DA SILVA MACHADO

UM ESTUDO SOBRE CRIPTOGRAFIA BASEADA EM ATRIBUTOS

Recife

2013

MARCOS VINÍCIOS DA SILVA MACHADO

UM ESTUDO SOBRE CRIPTOGRAFIA BASEADA EM ATRIBUTOS

Trabalho de conclusão de curso de CIÊNCIA DA COMPUTAÇÃO do CENTRO DE INFORMÁTICA da UNIVERSIDADE FEDERAL DE PERNAMBUCO, desenvolvido como requisito obrigatório parcial para obtenção do grau de bacharel em ciência da computação.

Orientação: Djamel Fawzi Hadj Sadok

Recife

2013

Resumo

A garantia de que a informação vai ser acessada por apenas aqueles usuários que tem acesso, ou seja, a confidencialidade da informação, é uma das principais metas dos sistemas criptográficos, e no esquema de criptografia baseada em atributos não poderia ser diferente. Mas a criptografia baseada em atributos, um esquema de chaves assimétricas, aborda esse problema tentando garantir uma alta granularidade com controle da informação, permitindo assim vários tipos de controles de acesso. Essas qualidades são produzidas utilizando conceitos atuais da criptografia moderna e de complexidade computacional, aliando abstrações no controle de políticas, permitindo assim que a confidencialidade seja atingida. Nesse trabalho será apresentada a criptografia baseada em atributos e seus principais esquemas, expondo uma análise de cada um deles observando critérios essenciais em sistemas criptográficos seguros e aplicáveis, e finalmente exibindo resultados obtidos a partir da implementação deles utilizando a framework CHARM e comparando-os.

Palavras-chave: Criptografia baseada em atributos, políticas, controle de acesso, CHARM.

Abstract

The assurance that the information will be accessed only by authorized users, in other words, confidentiality, is an essential goal in many cryptography schemas, and in attribute-based cryptography schemas it is not different. However, the attribute-based cryptography, an asymmetric key scheme, addresses this problem by trying to achieve a fined-grained access control, allowing several types of access control. Those qualities are generated by modern cryptography concepts, and computational complexity theory, allied with some policy control abstraction, thus providing confidentiality. In this work, a survey on attribute-based cryptography and its main schemes are presented along with an analysis of them, observing essential criteria in secure and applicable cryptographic systems, and, finally, bringing up and comparing results produced by the implementation of the schemes on CHARM framework.

Key-words: Attribute-based encryption, policy, access control, CHARM.

Agradecimentos

Primeiramente a Deus, à minha família, ao meu pai, à minha mãe e meus irmãos, pois sem eles eu não estaria aqui hoje.

À minha tia, ao meu primo e prima que têm sido minha segunda família, me acolhendo e me apoiando nessa difícil etapa.

Aos meus amigos que sempre estiveram dispostos a me ajudar, os quais muitos deles hoje são novos irmãos.

Ao Prof. Djamel Sadok por me orientar no desenvolvimento desse trabalho.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

Historicamente, criptografia é conhecida como a arte ou processo de escrita de cifras ou códigos, e assim é significada pelo dicionário MICHAELIS, 2004. Diversas civilizações antigas e modernas utilizaram esquemas criptográficos para manter o sigilo de informações vitais, e com isso ter um canal de comunicação seguro.

> Foram as ameaças de interceptação de informações por inimigos que motivaram o desenvolvimento de técnicas códigos e cifras para mascarar uma mensagem e, com isso, fazendo com que ela fosse lida apenas pela pessoa a quem se destina a mensagem. (Simon Singh, 2000, p. 13, tradução nossa).

No entanto, a criptografia moderna difere drasticamente da clássica, tanto pela forma de concepção dos novos esquemas, quanto pelas diversas formas de aplicação e os seus mais variados fins. A mudança é tão grande, que hoje, diversos métodos são usados transparentemente e sem o próprio conhecimento dos usuários.

“Em resumo, criptografia se transformou de uma forma de arte que tratava de manter a comunicação militar sigilosa, para uma ciência que ajuda a proteger sistemas para pessoas comuns ao redor do globo.”. (KATZ e LINDELL, 2007, p. 4, tradução nossa).

Mas, mesmo com todos os avanços, ainda há problemas na segurança da informação, como a manutenção e armazenamento de informações sensíveis dos usuários por terceiros na internet. Empresas como Google ou Yahoo, que têm e-mails, dados, preferências pessoais armazenadas, sempre se preocupam com o comprometimento dessas informações.

Um método para diminuir alguns desses problemas é armazenar os dados criptografados, logo, se alguma informação for comprometida, as perdas serão limitadas. Uma das desvantagens desse método é que o compartilhamento das informações criptografadas por um usuário se torna bem limitada, visto que para compartilhar o acesso a alguma das informações ou o usuário teria que servir de “tradutor”, ou ele teria que compartilhar a sua chave privada dando acesso a todas as suas informações.

Como nenhuma das opções citadas são atrativas a nenhuma das partes SAHAI e WATERS [3] idealizaram uma solução para esse problema introduzindo o conceito de criptografia baseada em atributos.

Esse novo conceito visa resolver de maneira flexível e escalável o problema de controle de acesso às informações criptografadas, de maneira a evitar o uso de chaves assimétricas em excesso, e a redundância de informações criptografadas iguais, que seria necessário armazenar para permitir o acesso a uma mesma informação por vários usuários. Percebendo a falta deste tipo de análise na literatura, este trabalho tem como objetivo avaliar criticamente pontos necessários em esquemas criptográficos após a apresentação de conceitos úteis ao seu entendimento.

A seção 2 irá apresentar os principais sistemas desenvolvidos por esse novo conceito junto do conhecimento e definições necessárias para o entendimento deles, a seção 3 mostrará os critérios que serão avaliados nos esquemas, já a seção 4 fará a análise nos sistemas dos critérios levantados na seção anterior, e finalmente será apresentada na seção 5 a conclusão e trabalhos futuros.

2 Criptografia Baseada em Atributos

O primeiro esquema de criptografia baseada em atributos foi introduzido por SAHAI e WATERS [3] em 2005, com a *Fuzzy Identity-Based Encryption* (IBE), que é um esquema para decriptação baseado em identidade, o qual o uso poderia ser es[1]tendido para a criptografia baseada em atributo.

No mesmo ano NAIL at al. [4] propuseram um esquema criptográfico baseado em tributos também com *threshold*, mas que tinha uma abordagem mais prática que o esquema anterior.

Em 2006, GOYAL et al. [5] propuseram um esquema de criptografia baseado em atributos com as políticas de acesso na chave privada dos usuários, *key-policy atribute based encryption* (KP-ABE). Além da política na chave, esse esquema traz uma estrutura de funções monótonas (*monotonic functions*) para acesso.

OSTROVSKY et al. [6] propuseram em 2007 um novo esquema com as políticas de acesso na chave privada com uma estrutura de funções não monótonas (*non-monotonic functions*) para acesso.

No mesmo ano BETHENCOURT et al. [7] propuseram um esquema em que a política de acesso estão no dado criptografado, *ciphertext-policy atribute-based encryption* (CP-ABE), em vez de na chave privada. Posteriormente, um grande número de esquemas criptográficos seguindo a mesma ideia foram propostos [8, 9, 10, 11, 12, 13].

Em 2010 e 2011 foi proposto por WANG et al. [14] um esquema hierárquico de criptografia baseada em atributos, *hierarchical attribute-based encryption* (HABE), o qual usa as políticas na forma normal disjuntiva para gerar as chaves hierarquicamente e assume que todos os atributos em uma clausula conjuntiva são administrados pela mesma autoridade, gerando a ideia de domínio. Esse esquema combina o esquema de encriptação hierárquica baseada em identidade, *hierarchical identity-based encryption* (HIBE) com o de encriptação baseada em atributos com a política no cifrotexto[1].

[1] Cifrotexto é um neologismo da palavra em inglês *ciphertext*, introduzido pelo Prof. Dr. Ruy de Queiroz, vinculado ao Centro de Informática da Universidade Federal de Pernambuco.

As próximas subseções irão esmiunçar três dos esquemas contextualizados, sendo eles o IBE, o KP-ABE e o CP-ABE, que são considerados os principais na literatura. Mas para um bom entendimento, antes, será apresentado conceitos necessários para o entendimento dos mesmos.

2.1 Conceitos

Alguns conceitos são necessários para o entendimento dos esquemas criptográficos supracitados, mesmo que de uma maneira superficial, por isso eles são dispostos abaixo.

2.1.1 Mapeamento Bilinear

Seja G_1 e G_2 dois grupos cíclicos multiplicativos de primos de ordem *p*. Seja g um gerador de G_1 e e um mapeamento bilinear, $e : G_1 \times G_1 \rightarrow G_2$. O mapeamento bilinear e apresenta as seguintes propriedades:

1. Bilinearidade: para todo $u, v \in G_1$ e $a, b \in \mathbb{Z}_p$, nós temos $e(u^a, v^b) = e(u, v)^{ab}$.

2. Não degeneração: $e(g, g) \neq 1$.

Nós dizemos que G_1 é um grupo bilinear se a operação de grupo em G_1 e o mapeamento bilinear $e : G_1 \times G_1 \rightarrow G_2$ são os dois eficientemente computáveis. Note que o mapeamento e é simétrico, porque $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

2.1.2 Coeficiente de Lagrange

É usado para interpolação polinomial, levando os valores de um determinado polinómio em outro. Dado um conjunto de pontos x_j e um conjunto de números y_j, o coeficiente de Lagrange resulta em um polinómio que todos os x_j pontos assumem seus y_j valores correspondentes.

Ele é definido como: $\Delta_{i,S}$ para todo $i \in \mathbb{Z}p^*$ e um conjunto *S* de elementos em $\mathbb{Z}p^*$: $\Delta_{i,S}(x) = \prod_{j \in S,\ j \neq i} \frac{x-j}{i-j}$.

2.1.3 Árvore de acesso

Seja τ uma árvore representando uma estrutura de acesso. Todo nó que não é folha é um *threshold gate*, descrito pelos seus filhos e um valor de limiar (*threshold*). Se num_x é o número de filhos de um nó x e k_x o seu valor de limiar, então $0 < k_x \leq num_x$. Quando $k_x = 1$, o *threshold gate* funciona como um "OU", quando $k_x = num_x$ funciona como um "E" lógico. Toda folha é descrita por um atributo e um limiar de valor $k_x = 1$.

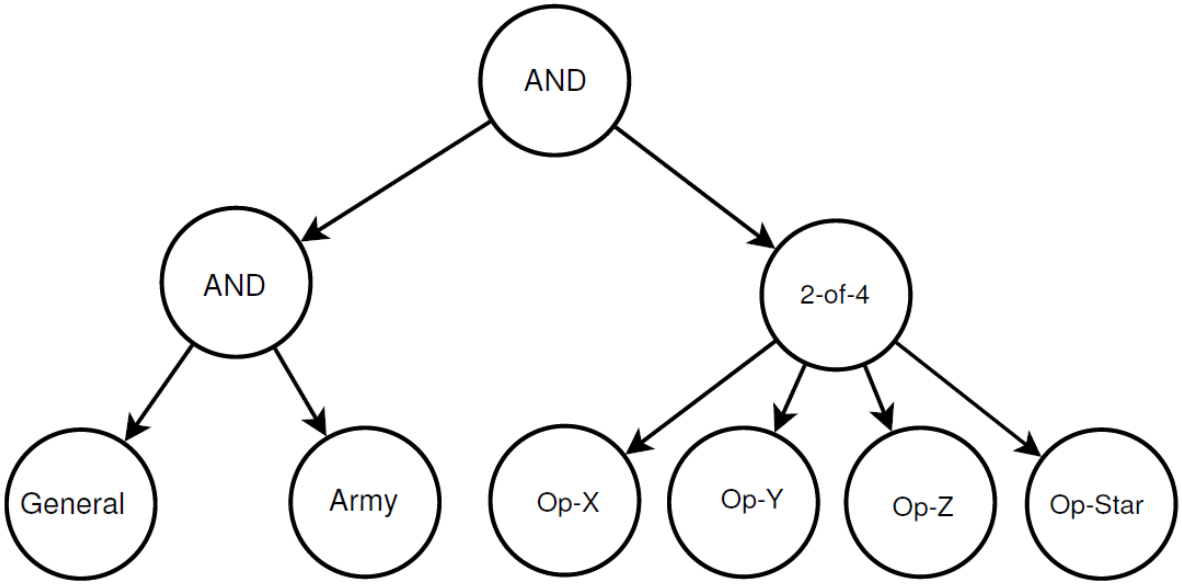

Figura 2.1: Exemplo de Árvore de acesso em que o nó raiz e o seu filho à esquerda tem valor de "E" lógico, seu filho à direita permite o acesso se tiver 2 atributos de qualquer um dos quatro.
Fonte: https://www.cs.utexas.edu/~vsub/pdf/ABE-scheme.pdf

Para facilitar o uso da árvore de acesso, algumas funções foram criadas. Definimos pai(x) como a função que retorna o pai do nó x. A função att(x) é definida apenas para os nós folhas e ela retorna o atributo associado a x. A árvore τ também tem definida uma função de ordenação entre os nós de cada folha, ou seja, os filhos de um nó é ordenado de 1 até *num*. Com isso, a função índice(x) retorna o número associado ao nó, os quais os valores dos índices são associados para os nós na árvore de acesso para uma dada chave gerada de uma modo arbitrário.

2.1.4 Compartilhamento de segredo

São métodos de compartilhar um segredo entre um grupo de participantes, em que cada um recebe uma parte do segredo. Esse segredo só pode ser

reconstruído quando um número suficiente de possíveis diferentes partes são combinadas. Cada parte é inútil sozinha.

Em específico é utilizado o esquema de Adi Shamir de compartilhamento de segredo. Em seu esquema cada participante recebe uma parte do segredo e é necessário uma quantidade *k* de partes para reconstruir o segredo, na qual *k* pode ser algumas ou todas as partes.

A ideia do compartilhamento de segredos de Shamir é dividir o segredo *D* em *n* pedaços D_1, D_2, ..., D_n de maneira que:

a) O conhecimento de *k* partes transforma o cálculo de *D* trivial.

b) O conhecimento de *k* – 1 ou menos pedaços tornam *D* completamente não determinado.

Esse tipo de esquema é chamado de *threshold* (*k*, *n*), e o modelo de Shamir utiliza o fato de que é necessário *k* pontos para definir um polinômio de grau *k* – 1, para satisfazer essas necessidades.

Partindo de que se quer usar um esquema de *threshold* (*k*, *n*) para compartilhar o segredo *S*, sem perda de generalidade assumindo que *S* é um elemento de um corpo finito *G* (que no nosso caso é um grupo multiplicativo de primos de ordem *P*) de tamanho $0 < k \leq n < P$, *P* sendo um primo.

Escolhe-se *k* – 1 coeficientes a_1, ..., a_{k-1} de *G*, e atribuímos a a_0 *S*. Constrói-se o polinômio $f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots + a_{k-1}x^{k-1}$, e com isso é possível obter *n* pontos desse polinômio, (*i*, *f(i)*), e dar cada um deles para um participante. Com isso, dado um conjunto *k* de pontos qualquer obtido do polinômio, utilizando interpolação linear (coeficiente de Lagrange), obtêm-se todos os coeficientes do polinômio construído e assim o segredo *S*.

2.2 Esquema baseado em atributos

O esquema baseado em atributos proposto por SAHAI e WATERS [3] foi idealizado para uma situação em que uma determinada pessoa gostaria de criptografar um documento para apenas os usuários que tivessem um certo conjunto de atributos. Por exemplo em um departamento de uma indústria

farmacêutica um documento contendo a fórmula de um novo remédio seria criptografada para apenas os participantes do projeto, que ainda trabalham no departamento poder ler. Nesse caso, qualquer pessoa que tivesse na sua identidade os atributos {“departamento-x”, “projeto-y”} poderia descriptografar o cifrotexto.

Nesse esquema existe uma autoridade, que é quem está com os dados, e existe os usuários, que é quem irá receber os dados. A autoridade é responsável por gerar chaves para os usuários criptografar ou descriptografar os dados, de acordo com os atributos que estão na chave pública e na chave mestra da autoridade. A chave privada é criada usando a identidade do usuário, que são uma lista de atributos que definem a permissão de acesso do usuário. A autoridade vai criptografar os dados com uma chave pública e um conjunto de atributos que definem quem tem acesso para descriptografar os dados. Os atributos usados na chave privada do usuário tem que casar com os usados na chave pública, atingindo assim um valor *d* (*threshold*) mínimo que permite o acesso.

Se algum atributo novo for incluído a lista de atributos para uma nova política, as chaves precisam ser geradas novamente. No esquema, existe quatro algoritmos para ser executados: *Setup*, *KeyGen*, *Encrypt* e *Decrypt*.

Seja G_1um grupo bilinear de primos de ordem *p,* seja *g* um gerador de G_1. Além disso, $e : G_1 \times G_1 \rightarrow G_2$ descreve um mapeamento bilinear e *d* um valor de limite (*threshold*).

As identidades serão subconjuntos de algum universo *U*, de tamanho |*U*|. Cada elemento será associado com um único inteiro do conjunto $\mathbb{Z}p^*$.

Setup: primeiro define-se um universo *U* de elementos, depois são escolhidos $t_1, \ldots, t_{|U|}$ uniformemente e aleatoriamente de $\mathbb{Z}p^*$. Finalmente escolhe-se *y* uniformemente e aleatoriamente de $\mathbb{Z}p$. As chaves públicas são PK = ($T_1 = g^t_1, \ldots, T_{|U|} = g^t_{|U|}$ $Y = e(g, g)^y$. A chave mestra MK = $t_1, \ldots, t_{|U|}$, *y*.

KeyGen: para gerar uma chave privada para a identidade $I \subseteq U$, um polinômio *q* de grau *d* – 1, que *q*(0) = *y* é escolhido aleatoriamente. A chave privada *D* é { $D_i = g^{\frac{q(i)}{t_i}}$ } $\forall_i \in I$.

Encrypt: a mensagem $M \in G_2$ e um conjunto de atributos *I'* é criptografada seguindo os seguintes passos: um número aleatório *s* é escolhido de $\mathbb{Z}p$ então o cifrotexto é publicado como: $C = (I', E = MY^s = e(g, g)^{ys}, \{E_j = T_j^S\}_{j \in I'})$.

Decrypt: o cifrotexto *E* criptografado usando os atributos *I'* é descriptografado utilizando a chave *D* gerada para a identidade *I*, da seguinte maneira: selecionam-se *d* atributos do conjunto $I \cap I'$ arbitrariamente para computar $e(E_j, D_j) = e(g, g)^{q(i)s}$ se $|I \cap I'| > d$. Com o coeficiente de Lagrange computamos $Y^s = e(g, g)^{q(0)s} = e(g, g)^{ys}$, com isso a mensagem $M = E/Y^s$.

É possível observar que a eficiência do algoritmo de encriptação é determinada pela quantidade de exponenciações, que, por sua vez, é determinado pela quantidade de atributos na identidade. O tamanho da chave pública cresce linearmente de acordo com o número de atributos no esquema. A eficiência do algoritmo de decriptação é determinando por *d* computações bilineares.

2.3 Esquema baseado em atributos com as políticas na chave privada

O primeiro esquema foi proposto em 2006 por GOYAL et al. [5], e além de ter seus atributos na chave privada, ele os usa na forma de árvore de acesso para construir a política de acesso. Se os atributos do texto criptografado satisfizerem a árvore de acesso, o texto é descriptografado.

Esse esquema permite políticas mais complexas e ricas, pelo uso da árvore de acesso. O “E” e o “OU” lógico podem ser usados em nós intermediários da árvore de acesso, permitindo assim mais de uma identidade, ou uma identidade mais descritiva na chave privada, diferentemente do primeiro esquema.

Essa construção é definida, também, com quatro algoritmos para serem executados, *Setup*, *KeyGen*, *Encrypt* e *Decrypt*, em sua descrição para um universo mais simples. Na descrição do esquema para um universo largo existe ainda o algoritmo de delegação de chaves, *Delegate*, que não será apresentado porque não é necessário para o funcionamento do sistema. É importante saber

apenas, que ele permite a um detentor de uma chave privada delegar chaves com um subconjunto dos seus atributos.

Seja G_1um grupo bilinear de primos de ordem *p,* seja *g* um gerador de G_1. Além disso, $e : G_1 \times G_1 \rightarrow G_2$ descreve um mapeamento bilinear. Um parâmetro de segurança *k* irá determinar o tamanho dos grupos e lembrando da definição do coeficiente de Lagrange, temos a seguinte definição.

Setup: primeiro define-se um universo *U* de elementos, depois se escolhe $t_1, \ldots, t_{|U|}$ uniformemente e aleatoriamente de $\mathbb{Z}p$. Finalmente,*y* é escolhido uniformemente e aleatoriamente de $\mathbb{Z}p$. As chaves públicas são PK = ($T_1 = g^{t}_1, \ldots, T_{|U|} = g^{t}_{|U|}$ $Y = e(g, g)^{y}$. A chave mestra MK = $t_1, \ldots, t_{|U|}, y$.

KeyGen: o algoritmo retorna como saída a chave privada do usuário, a qual o permite descriptografar os cifrotextos cifrados com o conjunto de atributos se, e somente se, a árvore de acesso retornar verdadeiro para o conjunto. O algoritmo começa escolhendo um q_x polinomial para todos os nós *x* da árvore de acesso τ. Esses polinômios são escolhidos de cima para baixo (top-down), ou seja, da raiz para as folhas, seguindo as seguintes regras: para cada nó *x* da árvore, escolha o grau d_x do polinómio q_x sendo um a menos que o valor de limiar (*threshold*) k_x daquele nó, ou seja, $d_x = k_x - 1$. Para a raiz *r*, escolha o valor $q_r(0) = y$ e os outros d_r pontos vão ser escolhidos aleatoriamente para completar q_r . Para qualquer outro nó *x*, escolha $q_x(0) = q_{pai(x)}$(índice(*x*)) e escolha os outros d_x pontos aleatoriamente para definir d_x completamente.

Para cada nó folha *x* de τ o seguinte segredo é dado: $D_x = g^{\frac{q_x(0)}{t_i}}$ onde i = attr(*x)*.

Encrypt: para criptografar uma mensagem *m* pertencente a G_2, para o conjunto de atributos *I*, escolhe-se um valor aleatório *s* pertencente a $\mathbb{Z}p$. O cifrotexto será:

$E = (I, E' = MY^{s}, \{E_i = T_i^{s}\}_{i \in I})$.

Decrypt: o algoritmo é definido de forma recursiva. Primeiramente é definida a função DecryptNode, que recebe como entrada o cifrotexto *E* = (*I*, *E'*,

$\{E_i\}_{i \in I}$), a chave privada *D* (assumindo que a árvore de acesso está contida na chave) e o nó *x* da árvore. Ela retorna um elemento de G_2 ou $\perp$.

Seja i = att(x). Se o nó x é uma folha então:

$$\text{DecryptNode}(E,D,x) = \begin{cases} e(D_x, E_i) = e\left(g^{\frac{q_x(0)}{t_i}}, g^{s \cdot t_i}\right) = e(g,g)^{s \cdot q_x(0)} \; se \; i \in I \\ \perp \text{ caso contrário} \end{cases}$$

Agora considerando o caso recursivo de *x* não ser uma folha, DecryptNode funciona da seguinte maneira. Para todos os nós *z* que são filhos de *x*, chamamos DecryptNode e guardamos o valor como F_z. Seja S_x um conjunto de filhos de *z* de k_x-tamanho, que $F_z \neq \perp$. Se não existe esse conjunto, então o nó não foi satisfeito e é retornado $\perp$.

Caso exista, então a seguinte computação é feita:

$F_x = \prod_{z \in S_x} F_z^{\Delta_{i,S'_x}(0)} = e(g,g)^{s \cdot q_x(0)}$, de acordo com a construção e utilizando interpolação polinomial permitida pelo uso do coeficiente de Lagrange.

Com a função DecryptNode definida, o algoritmo de descriptografar se resume a chamar DecryptNode passando a raiz da árvore, que nos dará como resultado $M = E/Y^s$.

Note que o controle do acesso aos dados está dentro da chave privada do usuário, ou seja, o usuário controla o acesso aos dados, mas os dados não controla quem pode os acessar. Isso é importante porque quer dizer que quem controla o uso dos dados é a entidade geradora de chave e não o dono dos dados. A performance desse esquema irá diferenciar no algoritmo Decrypt e KeyGen, porque agora passam a depender a árvore de acesso.

2.4 Esquema baseado em atributo com as políticas no cifrotexto

GOYAL et al. [5] quando publicou o esquema de criptografia baseada em atributos com a política de acesso na chave sugeriu no seu trabalho um esquema que a política estaria no cifrotexto, mas não apresentou nenhuma construção do mesmo. Mas, no mesmo ano, J. BETHENCOURT et al [7] apresentaram uma

construção de um esquema de criptografia baseada em atributos com a política de acesso no cifro texto.

Nesse modelo, a chave privada será identificada com um conjunto de atributos descritivos. O usuário que deseja criptografar os dados irá especificar por meio de uma árvore de acesso a política de acesso, a qual deve ser satisfeita pelo conjunto de atributos da chave privada para poder ser descriptografado.

O esquema é definido com quatro algoritmos básicos para o funcionamento e um adicional, que são: *Setup*, *KeyGen*, *Encrypt*, *Decrypt* e o *Delegate* respectivamente, que não será apresentado pelos mesmos motivos que os do esquema anterior.

Seja G_1um grupo bilinear de primos de ordem *p,* seja *g* um gerador de G_1. Além disso, $e : G_1 \times G_1 \rightarrow G_2$ descreve um mapeamento bilinear. Um parâmetro de segurança *k* irá determinar o tamanho dos grupos e temos definido o coeficiente de Lagrange como nos outros esquemas. Adicionalmente, temos uma função hash $H : \{0,1\}^* \rightarrow G_1$ que será modelada como um oráculo aleatório. Essa função irá mapear qualquer atributo descrito em uma cadeia binária para um elemento aleatório do grupo.

Setup: será escolhido um grupo bilinear G_1 de ordem prima *p* com o gerador *g*. Em seguida será escolhido dois expoentes aleatórios α e β, ambos pertencentes a ℤp. A chave pública será $PK = G_1, g, h = g^{\beta}, f = g^{\frac{1}{\beta}}, e(g,g)^{\alpha}$ e a chave mestra será $MK = (\beta, g^{\alpha})$.

KeyGen: o algoritmo de geração de chave privada recebe como entrada um conjunto *S* de atributos e retorna uma chave que identifica esse conjunto. Primeiramente é escolhido um $r \in \mathbb{Z}p$. aleatoriamente e depois será escolhido um $r_j \in \mathbb{Z}p$ para cada atributo $j \in S$. A chave é computada como:

$$SK = (D = g^{(\alpha+r)/\beta}, \forall j \in S : D_j = g^r \cdot H(j)^{r_j}, D'_j = g^{r_j})$$

Encrypt: como foi explicado antes, uma mensagem *M* é criptografada utilizando uma árvore de acesso τ. O algoritmo primeiro escolhe um polinómio q_x para cada nó *x* da árvore. Esses polinómios são escolhidos de cima pra baixo

(top-down), ou seja, da raiz *R* até as folhas. Para cada nó *x* da árvore, atribua o grau $d_x = k_x - 1$para o polinómio q_x, onde k_x é o valor de limiar do nó *x*.

Começando pela raiz *R* o algoritmo escolhe um $s \in \mathbb{Z}p$ aleatório e atribui $q_R(0) = s$. Depois, é escolhido d_R outros pontos aleatoriamente para o polinómio q_R para defini-lo completamente. Para qualquer outro nó *x*, ele define $q_x(0) = q_{pai(x)}$(índice(*x*)) e escolhe os outros d_x pontos aleatoriamente para definir d_x completamente.

Seja *Y* o conjunto de nós folhas em τ. O cifrotexto é computado da seguinte maneira:

$$CT = (\tau, C' = Me(g,g)^{\alpha s}, C = h^s, \forall y \in Y: C_y = g^{q_y(0)}, C'_y H(att(y))^{q_y(0)})$$

Decrypt: o algoritmo de descriptografar será descrito de maneira recursiva. Novamente é definido primeiramente a função DecryptNode, que recebe como entrada o cifrotexto $CT = (\tau, C', C, \forall y \in Y : C_y, C'_y)$, a chave privada *SK*, que está associada com o conjunto de atributos *S* e um nó *x* de τ. Se o nó *x* for uma folha, então *i* = att(*x*) e DecryptNode é definido assim:

$DecryptNode(CT, SK, x) = \frac{e(D_i, C_x)}{e(D'_i, C'_x)} = e(g,g)^{rq_x(0)}$. Se $i \notin S$, então DecryptNode retorna $\perp$.

Agora considerando o caso de *x* não ser uma folha, DecryptNode funciona da seguinte maneira. Para todos os nós *z* que são filhos de *x*, chamamos DecryptNode(*CT, SK, z*) e guardamos a saída como F_z. *Seja* S_x um conjunto arbitrário de tamanho k_z tal que $F_z \neq \perp$, se tal conjunto não existe a função retorna $\perp$.

De outro modo, será computado $F_x = \prod_{z \in S_x} F_z^{\Delta_{i, S'_x}(0)} = e(g,g)^{r \cdot q_x(0)}$, de acordo com a construção e utilizando interpolação polinomial permitida pelo uso do coeficiente de Lagrange.

Com a função DecryptNode definida, encrypt será igual a chamada de DecryptNode passando a raiz da árvore como nó, se a árvore for satisfeita pode *S*, então definimos $A = DecryptNode(CT, SK, r) = e(g,g)^{rq_R(0)} = e(g,g)^{rs}$. Com isso obtemos:

$$\frac{C'}{\frac{e(C,D)}{A}} = \frac{C'}{(e(h^s, g^{\frac{(\alpha+r)}{\beta}})/e(g,g)^{rs})} = M.$$

É notável que esse esquema melhora uma das desvantagens mais evidentes do esquema de políticas na chave privada, permitindo que os dados “escolham” quem pode descriptografá-lo. Quanto à performance, o algoritmo de geração de chaves e o algoritmo de encriptação são bem diretos, com o Encrypt requerendo apenas duas exponenciações para cada folha da árvore de acesso e o KeyGen requerendo duas exponenciações para cada atributo dado. A complexidade do Decrypt na sua forma mais simples requer no máximo uma exponenciação para cada nó ao longo do caminho que satisfaz a árvore de acesso, logo a eficiência também será definida pela maneira que se determina esse caminho.

3 Metodologia e Critérios

Um estudo comparativo entre os esquemas apresentados na seção anterior foi realizado e será discorrido na próxima seção, mas para isso, se fez necessária a escolha de critérios para serem avaliados e da implementação em código dos três esquemas.

Foram escolhidos para análise os seguintes critérios: confidencialidade da informação, alta granularidade de controle de acesso, resistência a ataques de combinação e escalabilidade, os quais foram extraídos observando o que os esquemas tentam alcançar, que é o armazenamento seguro de informações sensíveis permitindo controle de acesso e compartilhamento com alta granularidade.

Todos os pontos escolhidos são essenciais em um sistema criptográfico, afim de que seja possível utilizá-lo em aplicações de maneira segura e praticável, por isso será destacada a importância de cada um antes do comparativo.

A codificação dos sistemas criptográficos se fez necessária para uma comparação e avaliação de performance de vários aspectos dos sistemas. Todos eles foram implementados utilizando CHARM [16], uma framework para rápida prototipagem de sistemas criptográficos, logo ela também será detalhada a seguir.

3.1 Confidencialidade da informação

É um requisito necessário em todo sistema criptográfico que qualquer dado criptografado seja descriptografado se o usuário tem a permissão necessária. Essa característica é desejada porque os dados geralmente são armazenados em ambientes hostis como na nuvem, além disso, esses dados estarão acessíveis em qualquer lugar, e a internet é um meio hostil também.

O comportamento esperado é que todos os dados criptografados por qualquer um desses esquemas só sejam descriptografados pelas pessoas que tenham autorização, independente se a forma de controle de acesso seja na chave privada ou seja ele no cifrotexto.

3.2 Alta granularidade de controle de acesso

Faz-se necessário em um sistema de controle de acesso que seja possível atribuir permissões diferentes para usuários diferentes, mesmo que eles estejam em um mesmo grupo. Esse requisito é indispensável porque em um ambiente real nem todas as pessoas tem o mesmo tipo de acesso a mesma informação, ou ainda, porque as vezes é preciso dar a alguém o direito de acesso a algum tipo de informação que previamente não se tinha.

Esse nível de granularidade é um comportamento desejado pelos esquemas, pois alta granularidade no controle de acesso é uma das diretrizes motivadoras na construção dos mesmos, e além disso, também é muito desejado em um ambiente real.

3.3 Resistência a ataques de combinação

Ser resistente a ataques de combinação é uma característica muito importante nos esquemas e no controle de acesso da informação, visto que se dois ou mais usuários combinam seus atributos e conseguem descriptografar os dados, teríamos ai uma falha grave de segurança.

Ser resistente é o comportamento esperado nos esquemas, porque cada atributo está relacionado com um polinômio ou com um número aleatório, então diferentes usuários não conseguem combinar seus atributos uns com os outros.

3.4 Escalabilidade

É esperado em um esquema criptográfico que, mesmo que a quantidade de usuários ou fluxo de dados manipulado aumente, ele consiga se ajustar bem ou que exista algum mecanismo para comportar esse crescimento.

O que se espera dos sistemas baseados em atributos é que o desempenho não seja afetado mesmo com os problemas supracitados, e que ele possa funcionar eficientemente na presença de algum deles.

3.5 CHARM

CHARM [16] é um arcabouço criado para rápida prototipação de sistemas criptográficos em Python[1], e foi criado para diminuir o tempo de desenvolvimento e a complexidade de código, enquanto promove reuso. Ele tem um *design* híbrido, em que as operações matemáticas de alto custo são realizadas por módulos em C, enquanto o esquema criptográfico é feito utilizando Python, uma linguagem de alto nível que facilita o desenvolvimento uma vez que permite uma melhor abstração.

Dentre outros recursos, CHARM apresenta código de fácil extensão, uma *engine* e um compilador de protocolos, usa o padrão de projeto *adapter* em sua arquitetura, ferramentas para *benchmark* e *profiling* e é facilmente embutida em outras aplicações.[2]

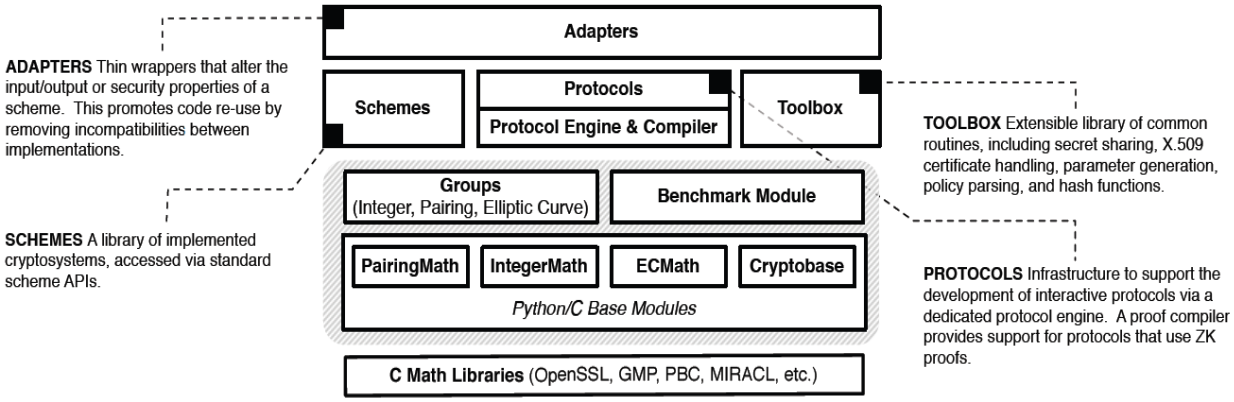

Figura 3.1: Visão geral da arquitetura da framework CHARM.

Fonte: J. A. Akinyele et al [16], p. 2.

3.6 Implementação

Como dito antes, para a implementação dos três esquemas foi usado o *framework* CHARM. Cada um deles está implementado em uma classe diferente e seguiram a descrição de suas propostas, tendo os quatro algoritmos principais, *setup*, *keygen*, *decrypt* e *encrypt*, definidos.

Para permitir uma maior praticidade uso, a distribuição do CHARM foi modificada para os três esquemas estarem no pacote de implementações

[2] Python é uma linguagem de programação de alto nível gerenciada pela Python Software Foundation.

prontas “charm.schemes”, permitindo que elas sejam incluídas e utilizadas de maneira rápida.

O primeiro esquema foi representado pela classe Fibe_SW05, o segundo pela classe KPabe_GSB08 e o terceiro pela classe CPabe_BSW07, e nelas já estão inclusas todas as estruturas necessárias para o funcionamento. Como CHARM é um framework de prototipação, os grupos oferecidos para mapeamento bilinear não são de texto, mas sim de números, logo não é possível criptografar texto sem algum mecanismo extra.

O que se espera dessas implementações é poder comparar o desempenho dos três esquemas de uma maneira mais prática, apresentando resultados visuais em detrimento de apenas explorar a complexidade algorítmica dos esquemas.

4 Comparações e análises

Na seção anterior foram mostrados todos os pontos que são considerados importantes em um esquema baseado em atributos, e porque eles são. Agora cada um deles será analisado e comparado nos esquemas.

Todos os três esquemas apresentam o requisito de confidencialidade da informação, e a dificuldade de quebrar o esquema é reduzida a dificuldade do problema bilinear de Diffie-Hellman (BDH), ao problema bilinear de Diffie-Hellman modificado (MBDH), ou ainda é provada a segurança dele sem redução a outro problema de complexidade computacional.

O primeiro esquema tem a dificuldade de ser quebrada reduzida à suposição do problema bilinear de Diffie-Hellman, e com a definição feita na seção 2.2 ele é *chosen-plaintext* seguro. Mas aplicando técnicas como as transformações de Fujisaki-Okamoto [17] ou *simulation-sound NIZK proofs* [18] é possível torna-lo seguro contra ataques do tipo *chosen-ciphertext*.

O segundo esquema apresenta a dificuldade reduzida ao mesmo problema do primeiro, o BDH, e do mesmo jeito que o primeiro, a sua prova de segurança foi feita para o modelo de ataque *chosen-plaintext*. Mas aplicando as mesmas técnicas que podem ser aplicadas no primeiro, é possível torna-lo seguro contra ataques *chosen-ciphertext*.

O último não apresenta redução a um problema complexo mais estudado, em vez disso, é apresentada a prova de segurança dele que um usuário não tem uma chance maior que desprezível de quebrar o esquema. Do mesmo modo que os anteriores ele é seguro contra ataques *chosen-plaintext*, mas pode ter sua segurança melhorada aplicando as mesmas técnicas dos anteriores.

Todos eles são seguros contra ataques de combinação, porque todos eles utilizam compartilhamento de segredo de Shamir na criação da chave privada ou na encriptação dos dados, e o compartilhamento de Shamir é comprovadamente seguro. A prova de segurança dessa técnica é feita utilizando os polinômios de Lagrange, e nela é provado que todo polinômio gerado por essa técnica é único e apenas utilizando os componentes certos o segredo pode ser recuperado.

Alta granularidade de controle é um requisito não presente apenas no primeiro esquema, e isso se deve a falta de expressividade do sistema. Essa falta de expressividade é decorrente do uso da tolerância a erros do esquema, ou seja, é preciso que pelo menos *k* atributos coincidam entre o cifrotexto e a chave privada.

No esquema de atributos na chave privada não foi apresentado o algoritmo de delegação, mas é possível ter um na construção do esquema para um universo largo, e isso, mais o fato dele ser bem mais expressivo que o anterior, caracterizam-no como um esquema com alta granularidade de controle.

O esquema de atributos no cifrotexto apresenta as mesmas características do anterior, e mais outra que é essencial em uma aplicação prática, o controle de quem acessa os dados é feito pelo próprio cifrotexto, em vez de ser feito pela chave privada, como seus dois anteriores. Essa é uma propriedade importante, já que o dono dos dados vai ter o controle de acesso dependendo minimamente da autoridade, evitando que todas as chaves privadas sejam refeitas quando um atributo for incluso na permissão de leitura, por exemplo.

Escalabilidade também é um item importante do ponto de vista prático, e não foi bem incorporado em nenhum dos esquemas apresentados. Embora dois deles apresentem um algoritmo para delegação de chaves privadas, só isso não ajuda muito quando se pensa em uma grande escala. Imaginando o cenário com vários centros, o qual cada um precisaria distribuir chaves, fazer um controle de atributos sem resultar em um mesmo atributo sendo usado em mais de um lugar é no mínimo complicado.

Além da falta de mecanismos de expansão, o primeiro esquema apresenta ainda mais outro problema para ser escalado, ele não é expressivo o suficiente para ser usado em um universo grande de atributos e políticas, como já foi explicado antes, e isso é uma falta grave quando se pensa em uma aplicação prática.

De forma resumida, a tabela abaixo relaciona esquema e os pontos avaliados.

Tabela 4.1: Relação dos esquemas criptográficos e os critérios analisados.

Item	IBE	KP-ABE	CP-ABE
Confidencialidade da informação	Tem	Tem	Tem
Alta granularidade de controle	Não	Tem	Tem
Resistencia a ataques de combinação	Tem	Tem	Tem
Escalabilidade	Não	Não	Não

Foram realizados testes nos seguintes algoritmos dos esquemas, *Keygen*, *Encrypt* e *Decrypt*, relacionando a quantidade de atributos ou sobreposição de atributos com o tempo de execução de cada um deles, como é mostrado abaixo.

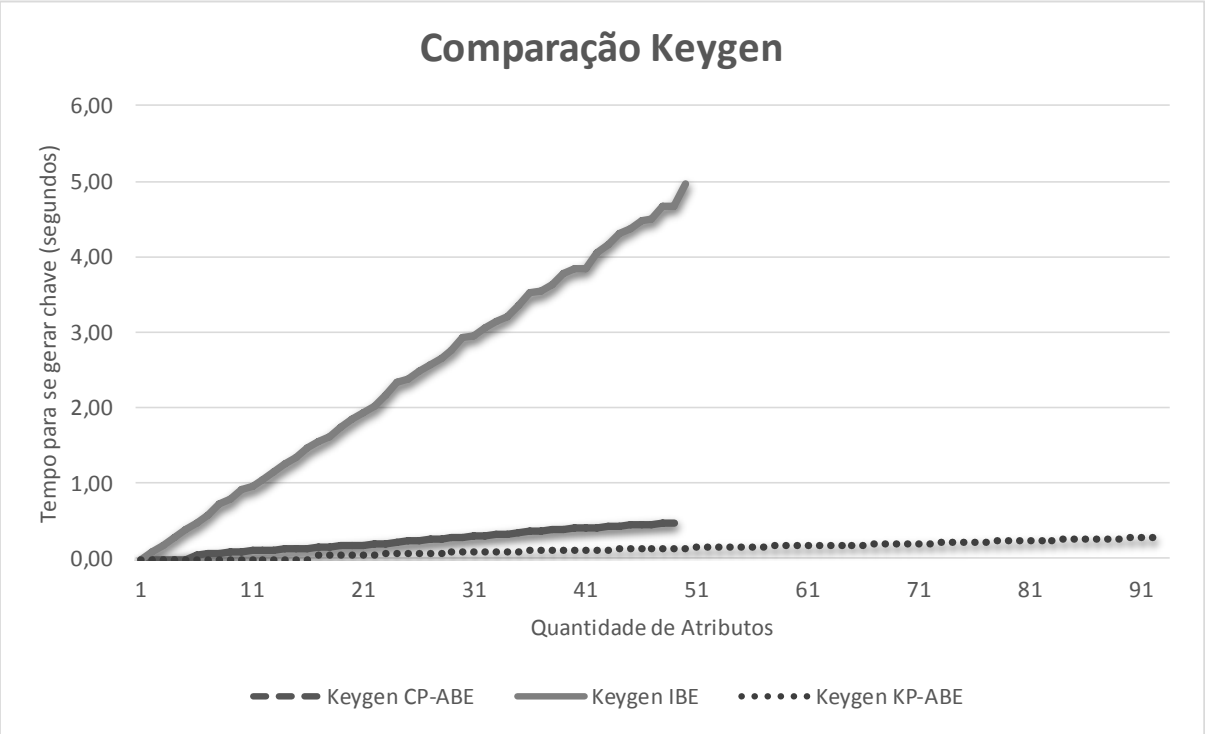

Figura 4.1: Comparação de performance do algoritmo keygen.

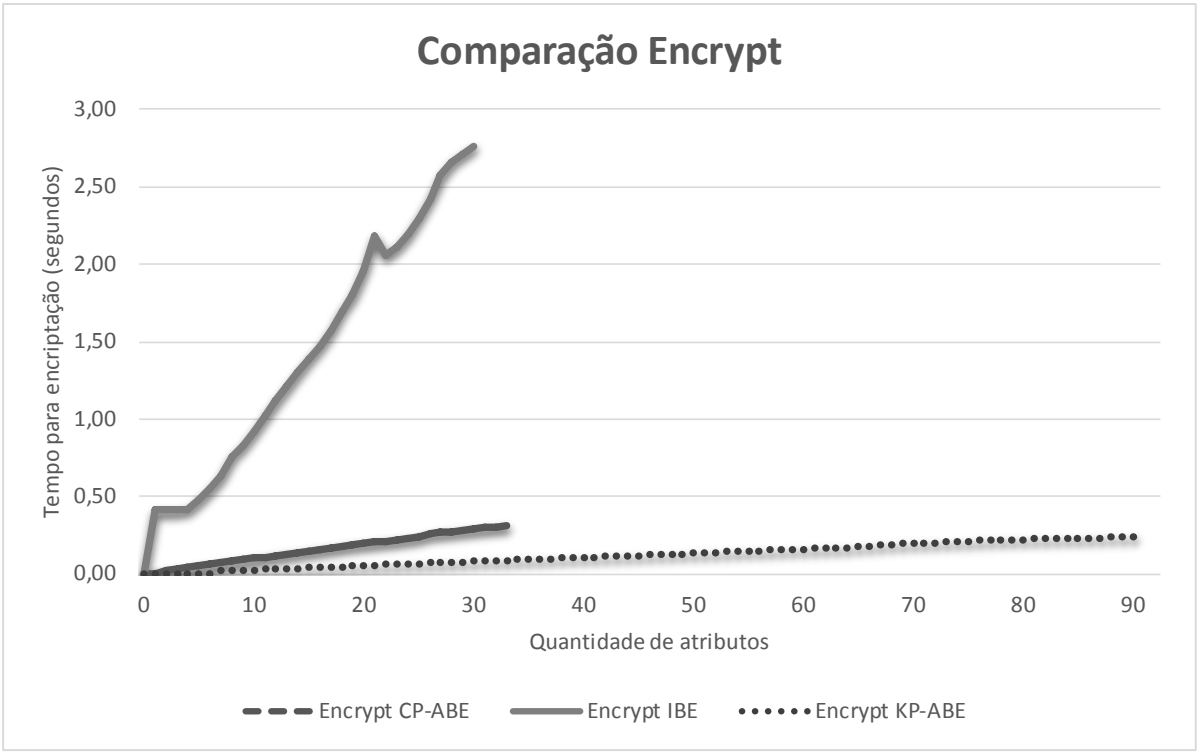

Figura 4.2: Comparação de performance do algoritmo Encrypt.

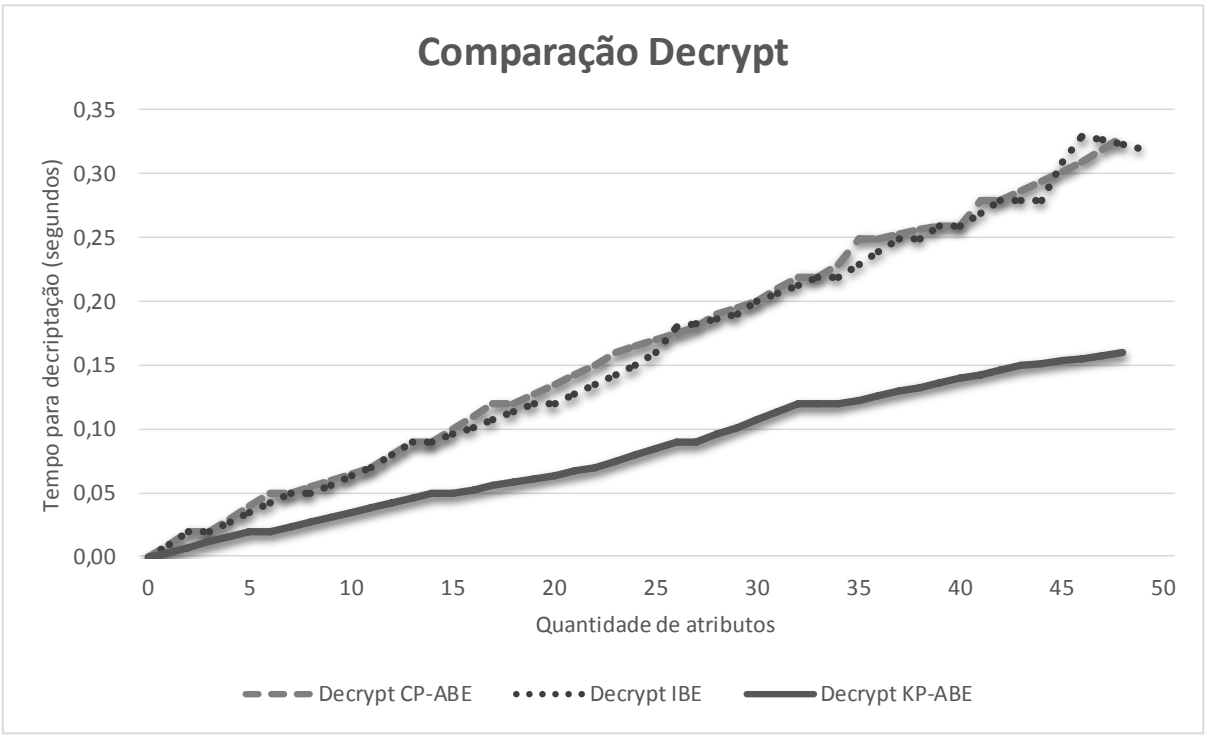

Figura 4.3: Comparação de performance do algoritmo Decrypt.

Observando todos os gráficos é possível perceber que em termos de eficiência o primeiro esquema está muito aquém dos seus sucessores nos algoritmos de geração de chave e de encriptação. O algoritmo de decriptação, por outro lado, apresenta um desempenho aceitável em comparação aos outros.

O esquema de atributos na chave privada foi o que apresentou o melhor desempenho em todos os algoritmos, apesar de o de atributos no cifrotexto não se diferenciar tanto.

Apesar de existirem diferenças entre todos os três sistemas, quando se compara performances, do ponto de vista prático, todos podem ser usados em aplicações dependendo das necessidades a que se destinam. Essas diferenças podem ser justificadas por uma comparação da complexidade dos algoritmos, como mostrada na seguinte tabela.

Tabela 4.2: Nomenclatura e descrição dos símbolos usados.

Item	Descrição
I	Conjunto de atributos de uma identidade IBE
CP	Chave publica
CM	Chave mestra
G_x	Grupo bilinear de primos de ordem p, x = 1,2
Q	Polinômio gerado pelo compartilhamento de segredo.
L	Coeficiente de Lagrange
S	Segredo a ser compartilhado
D	Limiar mínimo de casamento de atributos no IBE
t	Número aleatório de $\mathbb{Z}_p$
A	Árvore de acesso
A_F	Conjunto de nós folha da árvore de acesso
H	Função hash

Tabela 4.3: Relação dos algoritmos dos esquemas criptográficos e sua análise de complexidade.

Item	IBE	KP-ABE	CP-ABE
KeyGen	$\|I\|{G_2}^{q}{G_2}^{t^{\|I\|}}[(\|I\|+1){G_1}^{L}]$	$\|CM\|\ {G_1}^{\frac{Q}{t}}$	$\|I\|{G_1}^{t}H^{r}$
Encrypt	$\|CP\|\{{G_2}^{t^{\|I\|}}[(\|I\|+1){G_1}^{L}]\}^{S}$	$\|I\|{G_1}^{S}$	$\|A_F\|H^{Q}$
Decrypt	$D{G_2}^{q(i)\,S\,L}$	$\|A\|{G_2}^{S}$	$\|A\|H{G_2}^{S}$

Fazendo o paralelo entre os tempos gerados pelos esquemas implementados e a complexidade algorítmica de cada um deles, os resultados obtidos são definitivamente validados, se comportando de maneira semelhante ao crescimento algorítmico.

5 Conclusões

Nesse trabalho é possível entender como surgiu à criptografia baseada em atributos, conhecendo todos os mecanismos e conceitos matemáticos por trás dessa ideia. Além disso, são explicados os três principais esquemas desse género, porque são os modelos ou a base de muitas outras configurações. Mostrou-se de cada esquema todos os algoritmos que os compõem, junto de suas características, tornando assim claro o funcionamento de cada um.

Critérios importantes foram selecionados para observar características relevantes em esquemas criptográficos, e também para observar o cumprimento da proposta de criação dessas novas ferramentas. Foi feita a implementação dos três esquemas estudados, mas, para isso, foi apresentada a framework de prototipação rápida de sistemas criptográficos CHARM, permitindo assim a abstração de muitos dos problemas e a utilização de bibliotecas conhecidas e já estabilizadas.

Analisou-se cada critério escolhido em cada um dos esquemas, revelando a presença ou a ausência deles, e relacionando-os entre cada um deles trazendo à tona aspectos importantes. Os aspectos identificados são de grande importância para a avaliação dos esquemas, pois com eles torna-se possível entender as falhas ou ausências nos sistemas, e assim melhorá-las. Torna-se possível também, fazer uma melhor avaliação do estado da arte e distinguir quais foram os acréscimos dos novos modelos e o porquê deles.

Com a implementação de cada um dos esquemas foram realizados testes de performances, os quais foram expostos por meio de gráficos, permitindo assim que comparações pudessem ser feitas aliadas a uma avaliação mais concreta, possibilitando observar o quão factível e praticável realmente são os esquemas.

Pode ser contabilizada como falta nesse trabalho o estudo de mais esquemas, além dos principais, mas eles não foram estudados tendo como objetivo se ater aos “casos base”, e assim não acabar generalizando e perdendo o foco. Poderia ter sido feito também, um estudo da garantia da segurança e uma análise das reduções e dos problemas aos quais a dificuldade de quebra-los é

reduzida, ou ainda um estudo mais detalhado da complexidade computacional de cada um dos esquemas, mas isso especializaria o trabalho, o que não é o objetivo do mesmo.

5.1 Trabalhos futuros

Existem outros esquemas baseados em atributos que podem ser estudados, muitos deles derivados dos três principais apresentados nesse trabalho, como o esquema hierárquico de criptografia baseada em atributos, ou os esquemas de políticas na chave privada ou cifrotexto que utilizam estruturas de acesso com clausulas não monótonas (*non-monotonic*), possibilitando assim a utilização do “NÃO” lógico nas políticas.

É possível também acrescentar novos critérios a análise dos sistemas, tais como responsabilização (*accountability*) ou revogação de usuário, critérios esses que tem grande impacto em um sistema de controle por política. Pode-se partir para uma implementação de um software para controle de acesso a dados ou um de *broadcast encryption*, que são utilizações sugeridas por muitos autores dos sistemas de criptografia baseada em atributos.

Referências Bibliográficas

[1] Simon Singh, The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography, 1ª edição, Estados Unidos, 2000.

[2] Jonathan Katz and Yehuda Lindell, Introduction to Modern Cryptography: Principles and Protocols, 1ª edição, Estados Unidos, 2007.

[3] A. Sahai and B. Waters. Fuzzy Identity Based Encryption. In: Advances in Cryptology – Eurocrypt, volume 3494 of LNCS. Springer, 2005, p. 457–473.

[4] D. Nali, C. Adams, and A. Miri, "Using threshold attribute-based encryption for practical biometric-based access control," International Journal of Network Security, vol. 1, n° 3, p. 173-182, 2005.

[5] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," In Proceedings of the 13th ACM conference on Computer and communications security, p. 89-98, 2006.

[6] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," in *Proceedings of the 14th ACM conference on Computer and communications security*, p. 195-203, 2007.

[7] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of IEEE Symposium on Security and Privacy*, p. 321-334, 2007.

[8] K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi, "A ciphertext-policy attribute-based encryption scheme with constant ciphertext length," in *Proceedings of the Information Security Practice and Experience*, p. 13-23, 2009.

[9] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *Proceedings of the ICALP*, p. 579-591, 2008.

[10] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Mediated ciphertext-policy attribute-based encryption and its application," *Information Security Applications*, vol. 5932 of LNCS, p. 309-323, 2009.

[11] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," *Advances in Cryptology V EUROCRYPT*, vol. 6110 of LNCS, p. 62-91, 2010.

[12] T. Nishide, K. Yoneyama, and K. Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in *Proceedings of the Applied Cryptography and Network Security*, p. 111-129, 2008.

[13] B. Waters, “Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," *Public Key Cryptography V PKC*, vol. 6571 of LNCS, p. 53-70, 2011.

[14] G. Wang, Q. Liu, and J.Wu, “Hierachical attibute-based encryption for fine-grained access control in cloud storage services," in *Proceedings of the 17th ACM conference on Computer and communications security*, p. 735-737, 2010.

[15] G.Wang, Q. Liu, J.Wu, and M. Guo, “Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers," *Computer & Security*, vol. 30, p. 320-331, 2011.

[16] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, A. Rubin. "Charm: A Framework for Rapidly Prototyping Cryptosystems." In Journal of Cryptographic Engineering (JCEN) 2013.

[17] Eiichiro Fujisaki e Tatsunaki Okamoto, “Secure Integration of asymmetric and symmetric encryption schemes” In *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, p. 537-554. Spinger-Verlang, 1999.

[18] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *Proceedings of 40 IEEE Symp. On Foundations of Computer Science*, 1999.