Delegation and Tracing for Attribute-Based Cryptography

Lénaïck Gouriou

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'École Normale Supérieure de Paris

Délégation et Traçage pour la Cryptographie à Base d'Attributs

Soutenue par
Lénaïck Gouriou
Le 15 Mai 2023

École doctorale n°386
Sciences Mathématiques de Paris Centre

Spécialité
Informatique

Composition du jury :

Benoît Libert Zama	*Rapporteur*
Pascal Lafourcade Université Clermont-Auvergne	*Rapporteur*
Sébastien Canard Telecom Paris	*Examinateur*
Louis Goubin Université de Versailles-Saint-Quentin-En-Yvelines	*Examinateur*
Pierre-Alain Fouque Université Rennes 1	*Examinateur*
Cécile Delerablée Leanear	*Co-encadrante de thèse*
David Pointcheval CNRS	*Directeur de thèse*

ENS | PSL

Résumé

Le nombre d'appareils connectés par personne a massivement augmenté lors de la dernière décennie, en particulier dans les pays occidentaux. Étant donné que la sécurité des structures connectées au réseau peut être compromise à partir de n'importe quel point d'entrée par un attaquant malveillant, la sécurisation des trousseaux de clefs cryptographiques des utilisateurs devient centrale pour construire une infrastructure sécurisée. Afin d'explorer les réponses à cette problématique, nous proposons des solutions pour que les utilisateurs puissent gérer des appareils multiples. En particulier, nous nous penchons sur les pratiques cryptographiques qui sont compatibles avec les méthodologies modernes de contrôle d'accès basées sur des attributs. Ces pratiques doivent aussi intégrer les outils qui sont au cœur de la gestion d'appareils multiples par des utilisateurs. Le premier de ces outils est la délégation, qui permet aux utilisateurs de délimiter les capacités de déchiffrement de chacun de leurs appareils en fonction de leurs besoins. La délégation doit être possible sans nécessiter d'interaction avec une quelconque autorité, afin de préserver la vie privée et l'autonomie de l'utilisateur. Le second outil est le traçage, où nous exigeons que les appareils des utilisateurs puissent être identifiés en cas d'abus ou d'utilisation illégitime, afin que tout utilisateur puisse être tenu responsable de la gestion de ses appareils.

Nous commençons par présenter une approche pratique des Dual Pairing Vector Spaces (DPVS), qui est un système de construction et de preuves qui permet d'assurer le meilleur niveau de sécurité pour la cryptographie basée sur les attributs. Le DPVS est compatible avec des caractéristiques importantes de cette cryptographie telles que la richesse de l'expression des politiques de contrôle d'accès. Ensuite, nous présentons une nouvelle contribution pour le chiffrement basé sur les attributs sous la forme d'une nouvelle primitive : le Switchable-Attribute Key-Policy Attribute-Based Encryption (SA-KP-ABE). Dans un SA-KP-ABE, les attributs des utilisateurs et des chiffrés peuvent être "activés/désactivés" d'une manière indistinguable pour les utilisateurs. Nous prouvons que cette approche permet le traçage et qu'elle est compatible avec la délégation. Nous fournissons également une construction de SA-KP-ABE avec le DPVS. Notre dernière contribution est un schéma de signature basée sur des attributs qui permet deux méthodes de délégation. La première est la délégation habituelle de clefs, et la seconde permet de déléguer des politiques d'accès pré-approuvées qui peuvent être réutilisées pour signer différents messages. L'une ou l'autre de ces deux méthodes peut être utilisée en fonction du risque liée à une mauvaise gestion ou de la compromission de l'appareil recevant les clefs déléguées, car la seconde méthode implique moins de dommages potentiels que la première. De plus, nous prouvons également que notre schéma est compatible avec le traçage de signatures, où une autorité désignée peut lever l'anonymat des signatures suspectes.

Chiffrement basé sur les attributs ⋆ Signature basée sur les attributs ⋆ Délégation ⋆ Traçage

Abstract

The last decade has seen a massive increase in the number of connected devices per person, especially in western countries. As the security of connected structures can be compromised from any entry point by a malicious attacker, securing sets of cryptographic keys of users becomes an important keystone to build secure infrastructure. To explore answers to this problem, we propose solutions oriented towards the management of multiple devices for each user. In particular, we consider cryptographic practices that are compatible with modern access-control methodologies based on attributes. These practices should be compatible with features that are central to management of multiple devices. The first is delegation, as a means for users to delimit the decryption capabilities of each of their devices depending on their needs. Delegation should be doable without requiring interaction with any sort of authority, in order to preserve the user's intimacy and autonomy. The second is tracing, in the sense that we require that devices can be tracked in the case of abuse or illegitimate use so that any user can be held accountable.

We begin by presenting a practical approach to the Dual Pairing Vector Spaces (DPVS), a framework that allows for full security of attribute-based schemes, while being compatible with important features like expressive policies. Then, we present a new contribution for attribute-based encryption schemes in the form of a new primitive: Switchable-Attribute Key-Policy Attribute-Based Encryption (SA-KP-ABE). In a SA-KP-ABE, the attributes of users and ciphertexts can be "turned on/off" in a way that is indistinguishable for the users. We prove that this approach allows for tracing, and is fully compatible with delegation. We also provide a construction of SA-KP-ABE in the DPVS framework. Our last contribution is an attribute-based signature scheme that allows for two types of delegation. The first one is the usual delegation of keys, and the second one allows to delegate pre-approved policies that can be re-used to sign different messages. Either of these two approaches can be used depending on the risk of mismanagement or corruption of the device receiving the delegated keys, as the latter incurs less possible damage than the first one. Furthermore, we also prove our scheme to be compatible with tracing techniques, where a designated authority can lift the anonymity of suspicious signatures.

Attribute-Based Encryption ⋆ Attribute-Based Signature ⋆ Delegation ⋆ Traitor-Tracing

Remerciements

Pour commencer, j'aimerais remercier les deux personnes sans qui cette thèse n'aurait pas été possible.

David Pointcheval, qui m'a accueilli au sein de l'équipe CASCADE. David a été très présent tout au long de ma thèse, en étant toujours à l'écoute de mes intérêts de recherche. Ses propositions et conseils ont été indispensables au bon déroulement de ma thèse, et ce manuscrit n'aurait pas vu le jour sans son aide et sa patience indéfectibles. En particulier, j'ai pu compter sur lui pendant la période difficile du Covid-19, pendant laquelle il a su m'aider à garder un cap dans un monde qui le perdait.

Cécile Delerablée, qui a rendu ma thèse possible grâce à Leanear et au projet que cette équipe porte. Elle a su m'intégrer dans une équipe en agrandissement, qui est pleine de vie et d'objectifs ambitieux. Je ne peux m'empêcher de mentionner Yohann, pour sa capacité d'écoute sur nos projets communs et ses retours constructifs; Pierre, qui m'apprend à me méfier en bonne mesure des dangers d'internet; Bérenger, dont les courses nocturnes inspirent le dépassement de soi; ainsi que les nouveaux membres de l'équipe avec qui j'espère que nous aurons l'occasion de partager les grandes traditions de Leanear que sont les petits-déjeuners copieux et le barbecue, Steven, Tony, Hadrien, et Nathan. J'ai aussi une pensée pour Ange et Michele avec qui nous avons partagé de bons moments.

Je remercie aussi Benoît Libert et Pascal Lafourcade, pour avoir pris le temps de lire ma thèse et d'en faire un retour critique. J'adresse de même mes remerciements à Sébastien Canard, Louis Goubin, et Pierre-Alain Fouque qui me font l'honneur de participer au jury de thèse.

Mes remerciements vont également aux permanents de l'équipe CASCADE. J'aimerais en particulier remercier Brice, qui m'a guidé et éclairé dans ma première expérience d'enseignement, ainsi que pour toutes les fois où sa hauteur de vue a nourri ma réflexion et mon envie d'apprendre. Je pense aussi à Céline, qui a su débusquer avec intuition la personne derrière son cadeau de Secret Santa, ainsi que pour ses conseils avisés de danse. Je remercie aussi Phong, pour ses perspectives sur l'actualité du monde de la recherche. Enfin, je veux remercier Michel, Hieu, et toute l'équipe administrative du DI pour l'accueil qu'ils m'ont réservé et pour leur aide perpétuelle dans mon parcours.

Merci également à mes camarades de CASCADE, ainsi qu'à ceux des autres équipes qui occupent nos bureaux en bonne entente, sans lesquels mes journées de travail auraient été bien moroses. Mélissa, dont personne n'a su égaler le talent pour les événements conviviaux; Louiza, qui m'a amené voir des giraffes à Lyon (ainsi que des cryptologues à l'ANSSI); Chloé, avec qui je partage sur la perspective de jeune cryptologue; Antoine, pour sa passion pour les jeux et les nombreuses fois où nous avons joué ensemble; Balthazar, pour avoir pris soin de mon ancien smartphone; Romain, qui m'a guidé lors de mon premier Eurocrypt et pour son amitié; Jérémy, pour son esprit d'équipe dans les soirées jeux de société; Baptiste, pour son humour et son aide logistique; Huy, pour m'avoir initié au mensonge dans les jeux de société, bien que je reste un piètre menteur; Paola, dont l'humour est une inspiration pour moi; Ky, dont les manières de gentleman m'impressionnent toujours autant; Michael, pour sa patience lors de ses nombreuses explications du Cuckoo-hashing; Léonard, pour sa spontanéité et nos innombrables discussions; Paul, pour m'avoir transmis la passion du quantique et les nombreux partages que j'ai eu avec lui; Hugo, pour m'avoir initié au volleyball; Henry, mon cobureau de fortune; Robert, pour sa

gentillesse et son aide sur les réseaux; Nicolas, pour m'avoir réexpliqué plusieurs fois qu'il faisait du FHE et non pas du FE; Théo, pour ses conseils et nos soirées tardives au bureau; Antoine et Vincent, pour apporter un peu de pratique dans nos théories farfelues; ainsi qu'à tous les autres que j'ai rencontrés, Alexandra, Anca, Aurélien, Azam, Geoffroy, Jianwei, Marie, Michele, Quentin, et tous ceux que je n'ai pas nommés.

Je remercie aussi mes amis, en particulier mes témoins Corentin, Solenn, et Dorian, qui me supportent quotidiennement (au sens français et anglais du terme), ainsi que mes amis Liza, Constance, Thomas, Camille, Florent, Clément, Thibaut, l'équipe de la RSCDS, et plus largement tous mes amis de Rennes et de Paris. Merci particulièrement à Robin de m'avoir transmis l'opportunité de cette thèse.

Merci à ma famille, à mes mamans, Sha et Shadow, pour leur amour infini; ma grand-mère, pour son soutien indéfectible; Erika et Gerhard, pour m'avoir accueilli dans leur famille; Jean-Marc, pour les nombreux cours de snowboard; ainsi qu'a tous mes cousins, oncles et tantes qui font vivre chaque Noël avec énergie.

Pour finir, merci à Julie, ma fiancée, pour toutes les fois où elle m'a soutenu, pour tous les moments que nous avons partagés, et pour les instants de bonheur que nous vivons ensembles. C'est un grand honneur pour moi que de t'épouser.

Contents

Chapter

1 Introduction en Français

Chapter content

1.1 Contexte et Motivations

Aperçu. En 2023, le nombre moyen d'appareils connectés par personne est estimé à 9,4 en Europe occidentale, et à 13,6 en Amérique du Nord. À titre de comparaison, en 2018, la moyenne était de 5,6 en Europe occidentale et de 8,2 en Amérique du Nord. En seulement 5 ans, le nombre d'appareils a presque doublé dans ces régions, et cette croissance ne va probablement pas s'arrêter dans les années à venir si on considère le développement de l'Internet des objets. À mesure que le nombre d'appareils augmente, la surface d'attaque des entreprises et des particuliers qui les utilisent s'élargit naturellement. La fondation OWASP (`https://owasp.org`), l'une des plus grandes fondations dédiées à la sécurité des applications Web, a récemment placé les attaques sur le contrôle d'accès en tête des préoccupations de sécurité pour les applications Web, où les attaquants abusent des autorisations mal configurées, voire pas configurées du tout. Ces types de failles de sécurité ne peuvent qu'empirer avec l'augmentation du nombre d'appareils par personne, car le contrôle d'accès pour chaque appareil devient plus complexe à gérer, surtout si l'on considère que les applications peuvent être autorisées différemment par chaque appareil en fonction de la technologie utilisée, qui est susceptible d'être différente d'un appareil à l'autre.

Pour résoudre ces problèmes, des approches basées sur les attributs ont été proposées pour aider à gérer les autorisations. À haut niveau, les solutions basées sur les attributs autorisent les accès en évaluant les *attributs* de l'utilisateur contre la *politique* de sa requête. Par exemple, imaginons un utilisateur qui souhaite accéder et éditer du code dans un projet. Afin de protéger l'accès à ce projet, la politique pourrait spécifier que vous devez être un **Mainteneur** OU un **Developpeur** sur le projet. Par conséquent, sans les attributs appropriés, l'utilisateur se verrait refuser sa demande d'accès. L'avantage de ce paradigme est la granularité élevée qu'il fournit lorsqu'il faut déterminer les autorisations, ce que nous considérons comme un atout important pour répondre aux défis que nous avons mentionnés ci-dessus. En effet, chaque appareil d'un utilisateur présente des spécificités, certains étant hors de contrôle de l'utilisateur, comme les composants matériels ou les implémentations logicielles, mais beaucoup dépendent encore fortement de l'usage de l'utilisateur, par exemple dans le cadre personnel ou professionnel, ou bien si l'on utilise un ordinateur de bureau, un ordinateur portable ou un téléphone.

L'instanciation cryptographique concrète de cette approche a commencé avec le chiffrement basé sur les attributs (ABE), dans l'article fondateur de Goyal *et al.* [GPSW06]. Dans leur cryptosystème, la politique est intégrée à la clé privée de chaque utilisateur, tandis que les

attributs sont associés au chiffré. Tout utilisateur peut alors déchiffrer un message tant que les attributs du texte chiffré sont validés par la politique contenue dans ses clefs. Ce premier article introduit également la délégation de clefs, où tout utilisateur peut ajuster finement la politique d'accès pour chacun de ses appareils lorsqu'il crée une nouvelle clef à partir d'une clef existante, tant que la nouvelle politique est plus restrictive. Son équivalent naturel pour les signatures, la signature basée sur les attributs (ABS), a été introduit plus tard par Maji *et al.* [MPR11]. Dans un système ABS, les utilisateurs possèdent des attributs et peuvent signer dynamiquement un message avec une politique d'accès à condition d'avoir les attributs nécessaires pour valider la politique. Un vérificateur peut alors être convaincu que le signataire du message a effectivement respecté la politique associé au message qui a été signé.

Histoire de la Cryptographie Basée sur les Attributs. L'ABE est lui-même une généralisation du *chiffrement basé sur l'identité* (IBE), avec une première réalisation utilisant des appariement (ou couplage) en 2001 par Boneh et Franklin [Sha84, BF03], qui a été conçu à l'origine pour faciliter la gestion des certificats et des PKI, en ne nécessitant que l'identité du récepteur, typiquement une adresse e-mail, pour envoyer un message chiffré. Toutes les vérifications sont effectuées à l'avance par l'autorité centrale, qui n'a plus qu'à s'assurer que la clef secrète associée à une identité sera délivrée à la bonne personne.

Nous pouvons donc voir l'ABE, défini formellement pour la première fois en 2006 [GPSW06], comme une généralisation de l'IBE, en considérant les identités comme un type d'attributs qu'un expéditeur peut utiliser pour chiffrer. Il restait encore quelques défis à relever pour passer de l'IBE à l'ABE, en particulier comment exprimer les requêtes booléennes et les clefs impliquant plusieurs attributs (contrairement à une seule identité pour l'IBE), et comment empêcher les combinaisons de clefs provenant de plusieurs parties non autorisées qui formeraient une seule clef valide (mais non désirée), une propriété que l'on appelle résistance à la collusion.

En 2010, l'ABE a lui-même été généralisé comme faisant partie d'une nouvelle classe appelée *Functional Encryption* (FE), qui a été formellement introduite par Boneh *et al.* [BSW11]. Avec le FE, le récepteur peut obtenir *une fonction* du texte chiffré. Une fonction typique est la vérification booléenne : retourner le message si une certaine expression est vraie, sinon retourner $\perp$, fonction qui est exactement l'ABE.

De nouvelles contributions pour chacune de ces classes de fonctions sont faites régulièrement, car chaque classe de fonctions est généralement plus efficace que sa généralisation, et car certains problèmes restent ouverts pour chaque classe de fonctions (par exemple construire un schéma ABE sécurisé de manière adaptative à partir de l'hypothèse LWE, voir paragraphe suivant).

L'un des principaux défis pour les sous-classes de fonctions de FE était d'atteindre un niveau de sécurité plus élevé, la *sécurité adaptative*, qui n'a été réalisé qu'en 2009, d'abord pour l'IBE puis pour l'ABE, par Waters [Wat09] avec l'introduction de son nouveau paradigme : le Dual System Encryption (discuté plus loin dans la section 5.1). Ce paradigme s'est avéré particulièrement compatible avec le Dual Pairing Vector Spaces (DPVS) [OT09], qui repose sur deux espaces vectoriels avec des bases orthogonales pour l'opération de couplage. L'avantage du DPVS est de pouvoir facilement contrôler le comportement des éléments secrets dans les clefs et les chiffrés afin de faire des changements indistinguables pour les preuves, en utilisant cette orthogonalité. Les modifications dans les clefs et chiffrés sont effectuées par de simples changements de base, qui peuvent être regroupés en quelques catégories selon la disposition initiale des clefs et des chiffrés.

L'ABS a débuté en 2011 avec les travaux de Maji *et al.* [MPR11], en tant qu'approche complémentaire de l'ABE pour les signatures, mais a reçu globalement moins d'attention en termes de nombre de contributions. Bien que de nature différente, l'ABS remplit un rôle similaire au *Ring Signature* [RST01] et au *Group Signature* [Cv91], à savoir signer un message qui peut convaincre tout vérificateur que le signataire occupe une certaine position dans un groupe restreint. De plus, toutes ces primitives garantissent une forme d'anonymat pour les signataires, car les vérificateurs n'apprennent aucune information sauf que le signataire appartient au groupe

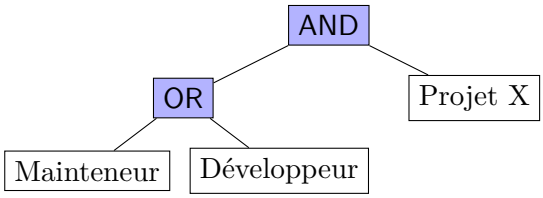

Figure 1.1: Un arbre comme politique d'accès, pour l'expression booléenne : ((Maintaineur OR Développeur) AND Projet X) comme condition pour être autorisé. Intuitivement, il faut posséder un attribut correspondant à une feuille pour la "rendre vraie", puis évaluer récursivement les portes des parents (AND, OR) dans l'arbre d'accès jusqu'à la racine. Les règles formelles d'évaluation de la politique d'un arbre d'accès sont présentées dans la section 3.2.

depuis lequel il a créé la signature. Cependant, contrairement à ces variantes de signature, dans l'ABS le groupe n'est pas formellement explicité avant la signature comme dans les signatures de groupe, ou explicité ad-hoc comme dans les Ring Signatures, mais exprimé dynamiquement par des politiques que le signataire peut valider.

Politiques d'Accès et Attributs. Historiquement, il y a eu deux modèles d'expression majoritaires pour exprimer les politiques d'accès. Le premier était l'arbre d'accès, issu des articles fondateurs de l'ABE [GPSW06, BSW07], dans lequel la politique est exprimée sous la forme d'un arbre qui peut être évalué depuis les feuilles jusqu'à la racine pour validation, chaque feuille étant considérée comme "vraie" si l'attribut qui lui correspond est présent (voir Figure 1.1). L'arbre d'accès contrôle de manière granulaire qui peut être autorisé par le biais de portes OR et AND situées dans les noeuds intérieurs, qui nécessitent soit d'avoir l'attribut pour un fils (portes OR), soit tous les fils (portes AND) pour passer la porte. Il existe également un autre type de portes appelées portes à seuil (également connues sous le nom de "portes k parmis n"), mais nous constatons qu'elles produisent des arbres d'accès aussi expressifs que ceux utilisant uniquement des portes OR/AND. L'autre modèle d'expression, qui est maintenant plus courant en raison de son expressivité et de ses propriétés algébriques, est le Linear Secret Sharing Scheme (LSSS), où la politique est représentée comme une matrice dont les vecteurs-lignes de la matrice sont les attributs. La politique peut alors être évaluée pour obtenir l'accès sous condition qu'une certaine combinaison linéaire des lignes (correspondant formellement à une certaine combinaison spécifique d'attributs) permette de reconstruire un vecteur prédéterminé, pour récupérer le message en clair. Un exemple est donné dans la figure 1.2. Les arbres d'accès sont faciles à comprendre et intuitifs, bien qu'ils ne soient pas le modèle le plus expressif pour les politiques d'accès, mais comme nous nous concentrons sur des outils de délégation ou de traçage pour les constructions basées sur les attributs, qui semblent agnostiques de ce choix, nous utiliserons des arbres d'accès à partir de maintenant. De plus, les arbres d'accès peuvent être facilement convertis en LSSS,

$$\begin{matrix}\text{Maintaineur}\\ \text{Développeur}\\ \text{Projet X}\end{matrix}\begin{bmatrix}1 & 1\\ 1 & 1\\ 0 & -1\end{bmatrix} \qquad \begin{aligned} L_1 + L_3 &= [1 \quad 0]\\ L_2 + L_3 &= [1 \quad 0]\end{aligned}$$

Figure 1.2: Une matrice LSSS comme politique d'accès, pour l'expression booléenne : ((Maintaineur OR Développeur) AND Projet X) comme condition pour être autorisé. Les lignes correspondent aux attributs (dans l'ordre de haut en bas) : Maintaineur, Développeur, Projet X. Pour valider l'accès, il faut être capable de calculer linéairement le vecteur ligne $[1 \quad 0]$ avec les lignes correspondant à ses attributs.

comme le montrent les travaux de [LW11, CPP17].

Une fois qu'un modèle de politique d'accès est choisi, la prochaine question naturelle qui vient à l'esprit est la suivante : comment les utiliser concrètement dans une construction ? Pour répondre à cette question, nous expliquons d'abord à haut niveau comment distribuer les politiques et les attributs dans un schéma cryptographique, puis nous poursuivons avec un aperçu de l'approche technique de nos constructions.

Comme nous l'avons mentionné plus haut, le premier article sur l'ABE plaçait la politique dans les clefs secrètes des utilisateurs, tandis que les attributs permettant d'évaluer les politiques se trouvaient dans le texte chiffré : nous appelons cela le KP-ABE (Key-Policy ABE). Naturellement, nous pouvons considérer sa contrepartie, Ciphertext-Policy ABE, où les clefs secrètes sont associées à des attributs, et le chiffreur décide d'une politique d'accès, au moment du chiffrement, pour accéder au message. Le choix entre KP-ABE et CP-ABE n'est pas trivial, car la partie en charge de la politique d'accès est la plus susceptible d'avoir un contrôle granulaire sur le déchiffrement, vu qu'elle contrôlera sous quelles règles les attributs peuvent être utilisés pour déchiffrer. C'est encore plus vrai en ce qui concerne la délégation : comme nous le verrons dans la Section 3.2, la délégation des arbres d'accès est très puissante et permet de nombreuses évolutions par rapport à l'arbre du délégant original, ce qui augmente la différence de contrôle sur le déchiffrement pour les utilisateurs entre KP-ABE et CP-ABE.

Examinons maintenant des instanciations de constructions basées sur les attributs utilisant des couplages, en particulier pour expliquer comment l'arbre d'accès (terme que nous utiliserons désormais pour faire référence aux politiques d'accès) et les attributs sont utilisés au sein de protocoles cryptographiques. En effet, nous voulons fournir des garanties solides pour les parties prenantes du protocole que la personne obtenant une autorisation par le biais d'un arbre d'accès l'a fait légitimement. Plus concrètement, l'arbre d'accès est intégré par le biais d'un partage secret, où chaque feuille de l'arbre d'accès reçoit une part. Les feuilles sont représentées par des clefs pour du KP-ABE, des chiffrés pour du CP-ABE, et des signatures pour de l'ABS. Parallèlement, les attributs, situés dans le texte chiffré pour du KP-ABE, et dans les clefs pour du CP-ABE et de l'ABS, sont liés par un aléa commun. Ensuite, grâce à la bilinéarité du couplage, on déchiffre en couplant chaque feuille de l'arbre d'accès avec son attribut correspondant. A ce moment, l'aléa commun se factorise, permettant ainsi de recombiner les parts du partage de secret initial, permettant le déchiffrement. Le lecteur peut alors se demander comment s'assurer que le "bon" attribut a été utilisé pour correspondre à une feuille. Ceci est garanti par l'utilisation d'un élément orthogonal entre la feuille et l'attribut : si le bon attribut est utilisé pour une feuille, la partie orthogonale s'annulera, sinon elle créera un bruit faussant le déchiffrement. Nous illustrons notre explication pour du KP-ABE avec la Figure 1.3.

Motivations. Notre objectif est l'exploration des solutions basées sur des attributs qui donnent une granularité sur le contrôle d'accès lors du déchiffrement pour les utilisateurs, et en particulier des solutions qui facilitent la gestion sur de différents appareils connectés. Pour cette raison, nous décidons de nous concentrer sur le KP-ABE et l'ABS avec délégation. Préserver autant que possible l'anonymat des utilisateurs est également un aspect important de notre mission, compte tenu des risques accrus de perte ou de corruption d'un appareil dans un environnement ou chaque utilisateur se doit de manipuler de nombreux appareils. Ainsi, les clefs et les signatures d'un utilisateur ne devraient idéalement rien révéler de son identité, et nous exigeons également que les clefs et les signatures ne soient pas reliables les unes aux autres, même si elles sont liées par délégation. Cependant, nous souhaitons également responsabiliser les utilisateurs, en particulier dans un système où ils disposent d'un grand pouvoir via la délégation, et nous exigeons donc une certaine forme de traçage par une autorité de confiance, qui puisse révéler l'utilisateur ayant effectué un déchiffrement ou une signature. Pour y parvenir, nous devons d'abord sélectionner un ensemble de méthodes compatible avec la sécurité adaptative, c'est à dire compatible avec la méthodologie de Dual Encryption System.

SubSpace-Ind: avec $\mathbf{b}_2^*$ caché

$$\mathbf{c}_t = \begin{pmatrix} \sigma_t t & -\sigma_t & \omega \end{pmatrix}_{\mathbb{B}}$$
$$\mathbf{k}_\lambda^* = \begin{pmatrix} \pi_\lambda & \pi_\lambda t_\lambda & a_\lambda \end{pmatrix}_{\mathbb{B}^*}$$

Le couplage des vecteurs peut être calculé comme un produit scalaire.

$$\begin{aligned} e(\mathbf{c}_t, \mathbf{k}_\lambda^*) &= [\pi_\lambda \sigma_t (1, t_\lambda)(t, -1) + \omega \ \cdot a_\lambda]_{\mathbb{G}_t} \\ &= [\omega \ \cdot a_\lambda + \pi_\lambda \sigma_t (t - t_\lambda)]_{\mathbb{G}_t} \\ &= [\omega \ \cdot a_\lambda]_{\mathbb{G}_t} \qquad \text{if } t = t_\lambda \end{aligned}$$

Figure 1.3: Approche par couplage de KP-ABE: la clef de la feuille λ est $\mathbf{k}_\lambda^*$, λ contient a_λ qui est un bout du partage de secret d'un certain a_0, et t_λ est l'attribut associé à λ. Le texte chiffré pour l'attribut t est $\mathbf{c}_t$, où ω est un aléa commun pour tous les attributs dans le chiffré. On peut voir les parties orthogonales : $\pi_\lambda(1, t)$ et $\sigma_t(t, -1)$. Les détails mathématiques complets sur la façon dont les couplages sont effectués peuvent être trouvés dans le chapitre 4.

1.2 Contributions

Comme première étape afin de construire nos ABE et ABS avec délégation, nous revenons sur les constructions proposées par Okamoto et Takashima [OT12a, OT11]. Dans les deux cas, nous prouvons une construction avec délégation et sécurité adaptative sous l'hypothèse SXDH au chapitre 5. La méthodologie commune à ces deux constructions, le Dual Pairing Vector Spaces (DPVS), est présenté au chapitre 4.

1.2.1 Dual Pairing Vector Spaces

Notre contribution pour le DPVS est une compilation de ses principaux outils de preuve présentés sous la forme de théorèmes pratiques sous l'hypothèse DDH. En effet, les résultats et preuves des travaux d'Okamoto-Takashima sont spécialisés pour leurs propres constructions, avec peu de réutilisabilité. Nous présentons donc de nouveaux outils génériques qui peuvent être adaptés à n'importe quelle preuve utilisant du DPVS. Ces résultats sont présentés dans la Section 4.4, avec un tableau récapitulatif rapide dans la Figure 4.1. Nous prouvons également un nouveau théorème pour des constructions en univers illimité pour les attributs dans les constructions avec le DPVS (voir Théorème 7).

1.2.2 Chiffrement basé sur les attributs

Nous commençons par un KP-ABE avec délégation, qui permet de plus de dissimuler des attributs dans le chiffré pour obtenir un traçage indétectable. Dans ce but, nous détaillons d'abord l'une des principales limitations que nous devons surmonter pour réunir délégation et traçabilité. Avec l'approche originale de [GPSW06], les attributs associés au chiffré sont explicitement indiqués dans le chiffré. La suppression de certains attributs peut permettre d'isoler des clefs privées spécifiques, mais il s'agit d'un processus public, et donc incompatible avec toute procédure de traçage, qui serait alors détectable par l'adversaire. L'alternative serait d'utiliser la sécurité sémantique du schéma pour envoyer différents contenu de chiffré en fonction des attributs possibles du traître [LT18], mais cette approche augmente drastiquement la taille des clefs, car il faut créer un double complet de la clef pour chaque attribut de traçage. Afin éviter cela, **notre première contribution sur l'ABE** est la nouvelle primitive : le Switchable-Attribute Key-Policy Attribute-Based Encryption (SA-KP-ABE), où il est possible d'invalider certains attributs dans

Caractéristiques	[OT12a]	[LW15]	[CGW18]	Ours
Securité	Adaptative	Adaptative	Adaptative	Adaptative
Hypothèse	DLIN	q-type	XDLIN	SXDH
Construction	CP/KP ABE	CP/KP ABE	IPE	KP ABE
Délégation	✓	×	×	✓
Traçage	×	✓	×	✓

Figure 1.4: Comparaison avec les autres travaux sur KP-ABE.

le chiffré de manière indistinguable. Plus précisément, nous apportons de nouvelles propriétés aux attributs dans les chiffrés (pour un traçage indétectable) mais aussi symétriquement aux feuilles dans les clefs (pour l'anonymat).

Dans un SA-KP-ABE, les attributs d'un chiffré et les feuilles d'un arbre d'accès $\mathcal{T}$ définissant la politique d'une clef peuvent être changés dans deux états différents : Les attributs peuvent être définis comme valides ou invalides dans un chiffré au moment du chiffrement, grâce a l'aide d'une clé de chiffrement secrète. Nous désignons alors $\Gamma = \Gamma_v \uplus \Gamma_i$ l'ensemble des attributs d'un chiffré, comme l'union disjointe des attributs valides et invalides ; les feuilles peuvent être définies comme passives ou actives dans l'arbre d'accès d'une clef au moment de sa génération, en utilisant la clef maître habituelle. Nous désignons également $\mathcal{L} = \mathcal{L}_p \uplus \mathcal{L}_a$, l'ensemble des feuilles, comme l'union disjointe des feuilles passives et actives. Un ensemble d'attributs valides/invalides $\Gamma = \Gamma_v \uplus \Gamma_i$ est accepté par un arbre d'accès $\mathcal{T}$ avec des feuilles passives/actives $\mathcal{L} = \mathcal{L}_p \uplus \mathcal{L}_a$, si l'arbre $\mathcal{T}$ est acceptant lorsque toutes les feuilles dans $\mathcal{L}$ associées à un attribut dans Γ sont à True, sauf si une feuille est active (dans $\mathcal{L}_a$) et l'attribut associé à la feuille dans le chiffré est invalide (dans Γ_i). Comme dit ci-dessus, les feuilles passives/actives dans $\mathcal{L}$ sont décidées pendant la procédure de génération des clefs par l'autorité centrale, en utilisant sa clef maître. Les clefs sont ensuite remises aux utilisateurs. Pendant le chiffrement, un chiffré est généré pour les attributs de Γ, mais il est possible de spécifier que certains attributs sont invalides en utilisant une clef secrète de traçage, ce qui rend virtuellement inutile les feuilles actives des arbres d'accès dans les clefs privées. Les feuilles passives ne sont pas affectées par les attributs invalides.

Notre deuxième contribution sur l'ABE est une instanciation concrète et efficace d'un SA-KP-ABE, avec des preuves de sécurité sous l'hypothèse SXDH. Nous expliquons finalement comment construire un KP-ABE délégable et traçable à partir d'une telle primitive. Comme le montre la Figure 1.4, notre schéma est le premier à combiner explicitement à la fois la délégation et la traçabilité des clefs pour un KP-ABE.

Discussions. Notre cadre présente des caractéristiques communes avec les approches modernes de KP-ABE, avec cependant quelques différences majeures. Tout d'abord, Waters [Wat09] a introduit la technique de Dual System Encryption (DSE), pour améliorer le niveau de sécurité des KP-ABE, depuis la sécurité sélective de [GPSW06] vers de la sécurité adaptative. Dans le DSE, les clefs et les chiffrés peuvent être définis comme étant *semi-fonctionnels*, dans la même veine que nos feuilles actives dans les clefs et les attributs invalides dans les chiffrés. Cependant, le DSE utilise uniquement les clefs et chiffrés semi-fonctionnels pendant la simulation, dans la preuve de sécurité, alors que notre construction les exploite dans la construction pratique. La preuve de sécurité nécessite donc d'utiliser de nouvelles techniques de preuve pour utiliser les éléments semi-fonctionnels dans la construction.

Par ailleurs, des notions de dissimulation d'attributs existent déjà, et sont des propriétés fortes qui ont été bien étudiées dans la littérature (en particulier pour du Inner-Product Encryption). Cependant, il n'est pas nécessaire de requérir des propriétés aussi fortes pour le traçage. En effet, notre propriété de (Distinct) Attribute-Indistinguishability est finement adapté au KP-ABE avec traçage.

Pour finir, quelques mots sur l'avantage de notre solution par rapport à une approche avec

KEM qui combinerait un KP-ABE avec délégation et un schéma de traçage de traître de manière générique. Cette solution fonctionne si l'on ne cherche pas l'optimalité de la résistance aux collusion pendant le traçage. En effet, le principal problème de cette utilisation de deux schémas indépendants est que pour chaque utilisateur, la clé KP-ABE et la clé de traçage ne sont pas liées. Par conséquent, les chiffrés de la partie ABE et de la partie traçage sont calculés indépendamment. Les utilisateurs en collusion peuvent donc tous participer pour déjouer le traçage sans restriction : la résistance à la collusion pour le traçage dans le schéma global sera exactement la résistance à la collusion du schéma de traçage du traître. Au contraire, notre construction tire parti de la résistance aux collusions du KP-ABE pour améliorer la résistance aux collusions du traçage : seuls les joueurs non révoqués par la partie KP-ABE peuvent s'allier contre la méthode de traçage. Ainsi, pendant le traçage, on peut révoquer des utilisateurs arbitraires grâce à la partie politique d'accès. Cela permet de réduire le nombre de traîtres actifs, en les maintenant éventuellement en dessous du seuil de résistance aux collusions du schéma de traçage de traîtres, de sorte que le traçage reste efficace.

1.2.3 Signature basée sur les attributs

Nous passons maintenant à nos contributions d'ABS, où nous proposons et prouvons deux constructions qui s'appuient sur celle adaptée des travaux d'Okamoto-Takashima [OT11] :

- **Notre contribution sur l'ABS** est une construction qui augmente la précédente avec différents types de délégation : la délégation d'attributs et la délégation de politiques d'accès;
- La deuxième construction ajoute la traçabilité en utilisant une nouvelle méthode utilisant un schéma de signature linéairement homomorphe.

Toutes ces constructions sont existentiellement infalsifiables sous les deux conditions suivantes : l'hypothèse SXDH dans le modèle standard, et la résistance aux collisions de certaines fonctions de hachage. Nos deux premières constructions sont également parfaitement anonymes, tandis que la troisième est calculatoirement anonyme sous une variante proche de l'hypothèse DDH. La traçabilité de notre dernière construction est prouvée dans le modèle d'oracle aléatoire, et repose sur la sécurité d'un schéma de signature linéairement homomorphe, et la simulation-extractability et la soundness de deux preuves à divulgation nulle de connaissance non-interactives (NIZK). Nous exploitons le schéma de [HPP20], dont la sécurité est prouvée dans le modèle générique pour les groupes avec couplage.

Discussions. Nos trois schémas se comportent de manière très proche en terme de performances. Les clefs et signatures sont linéaires en le nombre d'attributs utilisés pour signer, avec des performances comparables à [OT11]. L'un des principaux avantages du DPVS est de s'appuyer directement sur l'hypothèse de sécurité, ce qui simplifie le protocole, contrairement aux autres approches qui doivent intégrer des preuves non-interactives à divulgation nulle de connaissances plus complexes, qui sont au coeur des premières constructions d'ABS, afin de prouver la validation de la politique sans révéler d'information qui puisse compromettre l'anonymat du signataire. En particulier, nous mentionnons que l'intrication entre la délégation et le traçage de [GM19] rend impossible d'effectuer de la délégation sans avoir recours à des NIZK, tandis que notre approche modulaire permet d'éviter ce problème.

Nous avons séparé notre construction avec délégation de notre construction avec traçage afin de pouvoir souligner les spécificités de chacune pour le lecteur, cependant nous notons que ces constructions sont entièrement compatibles. On pourrait donc faire un ABS avec deux types de délégation, et de la traçabilité, en assemblant chacun des blocs spécifiques de nos constructions. On obtiendrait ainsi un système avec une infalsifiabilité existentielle sous l'hypothèse SXDH, mais un anonymat calculatoire sous une hypothèse proche du DDH. Nous notons que le traçage remonte à la clef originale remise à l'utilisateur, même lorsqu'il s'agit de tracer une signature

Caractéristiques	[OT11]	[GM19]	Ours
Hypothèses	DLIN	q-type + SXDH	SXDH
Délégation	×	✓	✓
Traçage	✓	✓	✓
Taille de signature	Linéaire en attributs	Linéaire en attributs et autorités	Linéaire en attributs

Figure 1.5: Comparaison avec d'autres travaux.

faite avec à partir d'une clef déléguée. En effet, la propriété de linéarité du schéma de signature garantit que le même scalaire w_{id} se trouve dans toutes les clefs déléguées et dans toutes les signatures provenant de la même clef de signature initiale, avec tout de même un anonymat calculatoire et une impossibilité de lier des clefs entre elles. Cela permet donc de traiter le cas où un utilisateur délègue ses droits de signature à plusieurs appareils : quel que soit la clef utilisée pour la signature, le traçage ciblera l'utilisateur initial.

Chapter

2 Introduction

Chapter content

2.1 Context and Motivations

Overview. As of 2023, the average number of connected device per person is estimated to be 9.4 in Western Europe, and 13.6 in North America. For comparison, in 2018, the average was 5.6 in Western Europe and 8.2 in North America. In only 5 years, the number of devices has almost doubled in those regions, and this growth is probably not going to stop in the incoming years when considering the development of the Internet of Things. As the number of devices grows, naturally the attack surface of companies and individuals using them also becomes larger. The OWASP foundation (`https://owasp.org`), one of the biggest foundation dedicated to the security of Web Applications, recently placed Broken Access Control as the number one security concern for Web applications, where attackers abuse poorly configured authorization of actions, if configured at all. These sorts of security breaches can only worsen with the increase of devices per person, as the access-control for each device becomes more complex to manage, particularly when considering how applications may be authorized differently by each device depending on the technology it uses, which is likely to be different from device to device.

As a solution to these problems, approaches based on attributes have been brought forward to help manage authorizations. At a high level, attribute-based solutions authorize actions by evaluating the *attributes* of the user against the *policy* of its request. As an example, let's suppose a user that wants to access to some code in a project. In order to protect the access to this project, the policy could specify that you need to be a **Maintainer** OR a **Developer** on the project, hence without the proper attributes the user wouldn't be granted access. The advantage of this paradigm is the high granularity it provides when determining authorizations, which we consider to be an important feature to answer the challenges we addressed above. Indeed, each device of a user bear specificities, with some being out of control of the user as hardware components, or software implementations, but with many still being strongly dependent on the usage of the users, for example personal of professional, or desktop or laptop or mobile.

The concrete cryptographic instanciation of this approach first started with Attribute-Based Encryption (ABE), in the seminal paper of Goyal *et al.* [GPSW06]. In such a cryptosystem, the policy is embedded inside each user's private key, while the attributes are situated in the ciphertext. Any user can then decrypt a message as long as the attributes in the ciphertext are validated by the policy in his keys. This first paper also introduces delegation of keys, where any user can finely-tune the policy for each of his devices when creating a new key from an existing one, for any more restrictive policy. Its natural counterpart for signatures, Attribute-Based Signature (ABS), was later introduced by Maji *et al.* [MPR11]. In an ABS scheme, users hold attributes and may dynamically sign any policy for a message for which their attributes validate said policy. Any verifier can then be convinced that the signer of the message effectively fulfilled the policy with its set of attributes.

History of Attribute-Based Cryptography. ABE itself is a generalization of *Identity-Based Encryption* (IBE), with a first realization using pairings in 2001 by Boneh and Franklin [Sha84, BF03], which was originally designed to ease the management of certificates and PKI structures, by requiring only the identity of the receiver, typically an email address, to send an encrypted message. All verifications would be done ahead of time by the Central Authority, who just needs to make sure that the secret key associated to an identity would be delivered to the right person.

We can then see ABE, with its first scheme in 2006 [GPSW06], as a generalization of IBE, by considering identities as simply a type of attributes that a sender can encrypt for. There were still some challenges to go from IBE to ABE, in particular how to express boolean queries and keys involving multiple attributes (contrary to a single identity), and how to prevent key combinations from many unauthorized parties that would result in an (unwanted) authorized key, which is a property called collusion-resistance.

In 2010, ABE was itself generalized as part of a new class called *Functional Encryption* (FE), which was formally introduced by Boneh *et al.* [BSW11]. In FE, the receiver can obtain *a function* of the ciphertext. A typical function is the boolean test: return the message if some expression is True, else return $\perp$, which is exactly ABE.

New contributions for each of these classes of functions are still common, as each class of function is generally more efficient than its generalization, and as some problems remain open for each class of functions (for example constructing an adaptively secure ABE scheme from the LWE assumption, see next paragraph).

One of the main challenge for subclasses of functions of FE was to reach a higher level of security, the *adaptive security*, which was only realized in 2009, first for IBE and then for ABE, by Waters [Wat09] with the introduction of his new paradigm: The Dual System Encryption (discussed later in Section 5.1). This paradigm proved particularly compatible with Dual Pairing Vector Spaces (DPVS) [OT09], a framework relying on two vector-spaces with orthogonal bases regarding the pairing operation. The advantage of the DPVS is that, using this orthogonality, one can easily control the behavior of secret elements in the keys and ciphertexts in order to make indistinguishable changes for proofs. The changes are made through simple changes of basis, which can be regrouped in a few categories depending on the disposition of keys and ciphertexts.

ABS starts off in 2011 with the work of Maji *et al.* [MPR11], as the complementary approach of ABE for signatures, but received overall less attention in terms of number of contributions. Although different in nature, ABS fills in a role similar to *Ring Signature* [RST01] and *Group Signature* [Cv91], that is to sign a message that can convince any verifier that the signer holds some position in a restricted group. Furthermore, all these primitives guarantee a form of anonymity for the signers, where all verifiers do not learn any information except that the signer belongs to the group he created a signature from. However, contrary to these variants of signature, in ABS the group is not formally explicited prior to the signature as in Group signatures, or explicited ad-hoc as in Ring signatures, but expressed dynamically through policies that the signer can fullfill.

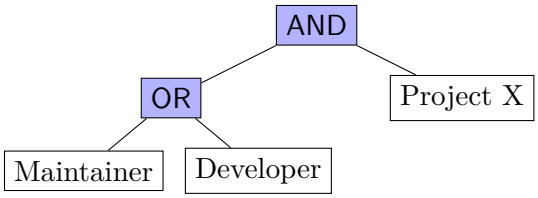

Figure 2.1: An access-tree as a policy, for the boolean expression: ((Maintainer OR Developer) AND Project X) as the condition to be authorized. Intuitively, one must possess an attribute matching a leaf to "make it True", and then recursively evaluate the (AND, OR) parent gates in the access-tree until the root. The formal rules for evaluating the policy of an access-tree are presented in Section 3.2.

Policies and Attributes. There have been historically two major expression models for policies. The first was access-tree, from the original papers for ABE [GPSW06, BSW07], in which the policy is expressed as a tree which can be evaluated from the leaves up to the root for validation, where a leaf is considered as "true" if it matches an attribute (see Figure 2.1). The access-trees granularly controls who can be authorized through OR and AND gates located in the interior nodes, which requires either to have the attribute for one child (OR-gates), or all the children (AND-gates) to pass the gate. There also exists of another type of gates called threshold gates (also known as "k-out-of-n gates"), but we note that they produce access-trees that are as expressive as the one using only OR/AND-gates. The other expression model, which is now more common due to its expressiveness and its algebraic properties, is the Linear Secret Sharing Scheme (LSSS), where the policy is represented as a matrix, and the vector-rows of the matrix are the attributes. The policy can then be evaluated for authorization if some specific linear combination of the rows (formally corresponding to some specific combination of attributes) allow to reconstruct a predetermined vector, to retrieve the original message. An example is given in Figure 2.2. Access-trees are easy to understand and intuitive, despite not being the most expressive model for policies, but as our focus in on additional features of attribute-based constructions, we will work with access-trees from now on. Furthermore, access-trees can be easily converted into LSSS, as shown in the work of [LW11, CPP17].

Once a policy model is chosen, the next natural question that comes to mind is: how do we use them concretely in a construction? To answer this, we first explain at a high-level how to distribute policies and attributes, and then proceed with an outline of our technical approach.

As we mentioned above, the first paper written on ABE put the policy inside the secret keys of the users, while the attributes to evaluate the policies are in the ciphertext: we call this Key-Policy ABE (KP-ABE). Naturally, we can consider its counterpart, Ciphertext-Policy ABE, where the secret keys are associated with attributes, and the encrypter decides a policy, at encryption time, to be matched to access the message. The choice between KP-ABE and CP-ABE

$$\begin{matrix}\text{Maintainer}\\ \text{Developer}\\ \text{Project X}\end{matrix}\begin{bmatrix}1 & 1\\ 1 & 1\\ 0 & -1\end{bmatrix} \qquad \begin{aligned} L_1 + L_3 &= [1 \quad 0]\\ L_2 + L_3 &= [1 \quad 0]\end{aligned}$$

Figure 2.2: An LSSS matrix as a policy, for the boolean expression: ((Maintainer OR Developer) AND Project X) as the condition to be authorized. The rows correspond to attributes (in order from top to bottom): Maintainer, Developer, Project X. In order to validate the policy, one must be able to linearly compute the row vector $[1 \quad 0]$ with the rows corresponding to his attributes.

SubSpace-Ind: with $\mathbf{b}_2^*$ hidden

$$\mathbf{c}_t = \begin{pmatrix} \sigma_t t & -\sigma_t & \omega \end{pmatrix}_{\mathbb{B}}$$
$$\mathbf{k}_\lambda^* = \begin{pmatrix} \pi_\lambda & \pi_\lambda t_\lambda & a_\lambda \end{pmatrix}_{\mathbb{B}^*}$$

The pairing of both vectors can be computed as a scalar product.

$$\begin{aligned} e(\mathbf{c}_t, \mathbf{k}_\lambda^*) &= [\pi_\lambda \sigma_t (1, t_\lambda)(t, -1) + \omega \ \cdot a_\lambda]_{\mathbb{G}_t} \\ &= [\omega \ \cdot a_\lambda + \pi_\lambda \sigma_t (t - t_\lambda)]_{\mathbb{G}_t} \\ &= [\omega \ \cdot a_\lambda]_{\mathbb{G}_t} \qquad \text{if } t = t_\lambda \end{aligned}$$

Figure 2.3: Pairing approach to KP-ABE: the key for leaf λ is $\mathbf{k}_\lambda^*$, which contains a_λ as a share to reconstruct some secret a_0, and t_λ the attribute associated to the leaf λ. The ciphertext for attribute t is $\mathbf{c}_t$, where ω is a common random for all attributes in the ciphertext. One can see the orthogonal parts: $\pi_\lambda(1, t)$ and $\sigma_t(t, -1)$. Full mathematical details for how pairings are done can be found in Chapter 4.

is not trivial, as the side in charge of the policy is the one most likely to have a granular control over decryption, as it will control under which rules the attributes can be used to validate the policy. This is even more true regarding delegation: as we will see in Section 3.2, delegation of access-trees is very powerful and allow for a lot of evolutions from the original delegator tree, increasing the difference of control over decryption for users between KP-ABE and CP-ABE.

We now delve deeper into the instanciations for Attribute-based constructions using bilinear pairings, in particular to explain how the access-tree (which we will now use to refer to policies) and attributes are put inside the cryptographic scheme. Indeed, we want to provide cryptographically sound guarantees for either sides of the protocol that the person obtaining authorization through an access-tree did so legitimately. More concretely, the access-tree will be embedded through a secret sharing, where each leaf of the access-tree is given a share. The leaves are represented by keys for KP-ABE, ciphertexts for CP-ABE, and signatures for ABS. Meanwhile, the attributes, situated in the ciphertext for KP-ABE, and in the keys for CP-ABE and ABS, are bound by a common random. Then, through the bilinearity of the pairing, one decrypts by pairing each leaf of the access-tree with its corresponding attribute. There, the common random will factorize, thus allowing to recombine the shares into the initial secret sharing, ensuring decryption. The reader might wonder how we can ensure whether the "good" attribute has been used to match a leaf. This is achieved through use of an orthogonal element between the leaf and the attribute: if the proper attribute is used for a leaf, the orthogonal part will cancel itself out, else it will create a noise tampering with the decryption. We illustrate our explanation for KP-ABE with Figure 2.3.

Motivations. Our aim is to explore attribute-based solutions that give granularity over access-control over decryption to users, and in particular solutions that ease the management over different devices. For this reason, we decide to focus on KP-ABE and ABS with delegation. Preserving anonymity of users as much as possible is also an important feature of our work, considering the increased risks of a device being lost or getting corrupted in a multi-device setting compared to a setting with a single device. Thus, keys and signatures from a user should ideally not reveal anything about their identity, and we also require that keys and signatures are somewhat independent from one another, even if they are related by delegation. However, we also want to keep users accountable, particularly in a system where they have great power through delegation, and as such we will also require some form of tracing by a trusted authority, as a way to reveal the user behind a decryption or signature. In order to achieve this, we

first need to select a framework that is compatible with full security using the Dual Encryption System methodology.

2.2 Related Work

2.2.1 Frameworks for Attribute-Based Cryptography

The first frameworks for ABE relied on two groups compatible for bilinear pairings [GPSW06, BSW07]. Security proofs used a partitioning technique, which limited security of schemes to *selective security*, where the adversary has to announce his target set of attributes before the beginning of the game. In order to reach a higher level of security, the *adaptive security*, Waters [Wat09] introduced a new paradigm: The Dual System Encryption (discussed later in Section 5.1). Although originally designed for IBE, it is also adequate for adaptive security in ABE. In particular, a new approach proved compatible with this new paradigm: groups of composite order [LOS+10]. However, as the composite order setting is quite inefficient [Gui13], tools were developped to convert composite order setting into a prime order setting [Fre10, Lew12]. One of these technique relied on the Dual Pairing Vector Spaces (DPVS) framework [OT09], which uses multi-dimensional vector spaces of prime order groups. As the DPVS proved useful for a quantity of works in ABE and other similar primitives [OT11, OT12a, DOT18], we decided to use it as well for the basis of our work. However, the tools from Okamoto-Takashima are hardly reusable for new schemes, as they use a great number of problems and sub-problems that are specific to their constructions, thus we must adapt them to our work. We also note the existence of other generic tools that capture the expressiveness of policies and the central proof argument for attribute-based schemes, but are neutral regarding our target application of delegation or tracing [CGW15], [Att16].

2.2.2 Traitor-Tracing

In a traitor-tracing system, a tracing authority can interact with a Pirate Decoder (PD) that non-legitimately decrypts ciphertexts, using one or more decryption keys of legitimate users (the traitors), in order to determine which user's private keys are used by the PD. Depending on the possible interactions with the PD, the model may vary on a spectrum from white-box, where the tracer can access information about the PD implementation, to black-box, where the tracer can only use the PD as a decryption oracle. While the white-box scenario offers the best properties of consistency and user privacy [Zha21], it is also the most demanding setting, as access to full implementation of the Pirate Decoder might be difficult. On the other side, the black-box setting offers the most reasonable scenario, as it can even work when interacting remotely with the PD. For this reason, we will focus on black-box traitor tracing. The first traditional approach is to embed codewords (also called "fingerprints") with specific properties in the decryption keys. These codewords can then be recovered, under some marking assumptions that address collusion of traitors, after interacting with the PD. Boneh and Shaw [BS95] proposed a tracing technique by embedding codewords in each ciphertext. With this approach, the ciphertext size has to be linear in the length of the codeword, and this length quickly increases with the size of the possible collusion. Boneh and Naor [BN08] improved this approach with a shorter ciphertext: only some bits of the codeword are involved in each ciphertext, but in this case tracing requires additional assumptions on the decryption capabilities of the PD.

Boneh *et al.* [BSW06], followed by [BW06], proposed traceability (and revocation) whatever the size of the collusion using a new linear tracing technique, with ciphertexts of size $\sqrt{N}$, where N is the maximal number of users. Much later, [GKW18] showed that this approach had a flaw in the secret-key setting and proposed a method to fix it.

2.2.3 Multi-Receiver Encryption

Broadcast Encryption. When used with efficient revocation, broadcast encryption is a nice tool for fine-grained access-control, as proposed in the seminal NNL paper [NNL01]. In a broadcast encryption system, the sender specifies a target set T of users who can read the message. All other users cannot access the message.

While it may seem possible to trace a compromised key exploited by a PD using revocation, by a trial and error approach or more sophisticated techniques, the PD might stop answering if it detects the tracing procedure. And this is an easy task for an adversary with classical broadcast encryption schemes that leak information about the target set, and so the revoked devices.

Anonymous broadcast encryption addresses this leakage of the target set, but even the most efficient solutions [LPQ12] cannot avoid a ciphertext being linear in the target set or the revocation set. Ak *et al.* [AKPS12] proposed a generic method to transform a broadcast encryption scheme into a trace and revoke scheme, using fingerprinting codes with additional samplability property, to make the usual mode and the tracing mode indistinguishable. But it does not provide fine-grained access-control as ABE schemes.

Attribute-Based Encryption (ABE). ABE has first been proposed in the paper by Goyal *et al.* [GPSW06]. They propose a concrete construction of KP-ABE, for any monotonous access structure defined by a policy expressed as an access-tree with threshold internal gates and leaves associated to attributes. Attributes in the ciphertext are among a large universe $\mathcal{U}$ (not polynomially bounded). Given an access-tree $\mathcal{T}$ embedded in a private key, and a set of attributes $\Gamma \subset \mathcal{U}$ associated to a ciphertext, one can decrypt if and only if Γ satisfies $\mathcal{T}$.

Furthermore, they laid down the bases for delegation of users' private keys: one can delegate a new key, associated with a more restrictive access-tree. This first paper on KP-ABE allows fine-grained access-control for multiple devices, dealing with delegation of keys for more restrictive policies. However, few works have followed from here, except for some isolated work on Hierarchical CP-ABE [WLW10, LYZ19], which bears inherent differences from KP-ABE as keys consist in set of attributes which are less structured than policies. An important contribution on the topic made by Shi and Waters [SW08] was to remark that that original definitions of Hierarchical IBE [HL02] suffered from an incomplete perspective on how the adversary obtained keys during the query phase, in particular regarding keys that could come from agents other than the authority. To take into accounts these various scenarios, they proposed a generalized model of delegation where the delegation history of the adversary must be tracked.

Predicate Encryption/Inner-Product Encryption (IPE). Okamoto and Takashima presented many IPE [OT10, OT12a, OT12b], together with LSSS: the receiver can read the message if a predicate is satisfied on some information in the decryption key and in the ciphertext. Inner-product encryption (where the predicate checks whether the vectors embedded in the key and in the ciphertext are orthogonal) is the major tool for their work, which is instanciated in the DPVS framework. On top of achieving adaptive security, their work also provide attribute-hiding (from [KSW08]) where no information leaks about the attributes associated to the ciphertext, except for the fact that they are accepted or not by the policies in the keys. It gets closer to our goals, as tracing might become undetectable. However, it does not seem any longer compatible with delegation, as the security proofs require all the key generation material to remain a secret information for the key issuer only.

As follow-up works, Chen *et al.* [CGW15, CGW18] designed multiple systems for IPE, with adaptive security, and explored full attribute-hiding with weaker assumptions and shorter ciphertexts and secret keys than in the previous work of Okamoto-Takashima. However, it does not fit our expectations on delegation, for the same reasons. On the other hand, Attrapadung also proposed new ABE schemes based on Pair Encoding Systems, which allow for extremely general predicates and large universes [AT20], but this deals neither with delegation nor with any kind of attribute-hiding, as we would need.

Traceable ABE. Wong *et al.* [LW15, LLLW17] combined the technique of [BW06] into a CP-ABE, with policy encoded in a Linear Secret Sharing Scheme (LSSS). Those techniques nevertheless seem incompatible with delegation properties. Intuitively, their approach assigns each single user to a different cell in a table, and then methodically tests each cell of the table for a traitor, with linear tracing. This is quite exclusive with delegation for the users, as one cannot add more cells in the table.

Lai and Tang [LT18] proposed a framework for traitor-tracing in ABE. Their technique is a generic transformation to make any ABE into a traceable ABE, following Boneh and Shaw [BS95] methodology with codewords. However, their construction is a generic one, and the additional layer for tracing incurs an expensive overhead on keys by requiring a full copy of the access-tree per character of the tracing word. Nevertheless, our approach will be in this vein, but for a more specific construction.

2.2.4 Anonymous Signatures

Group Signatures. In a Group Signature scheme [Cv91], a group is represented by a group public key, which can be used by any verifier to check signatures done by a member of the group. Each group member owns a distinct secret signing key, which can be used to anonymously sign an message in the name of the group. Meanwhile, the group manager holds a group manager's secret key, which can be used to generate the group, as well as to trace a signature generated by a member of the group to the original signer. The first fully secure realizations of Group Signatures were static [BMW03], where the group is fixed during the Setup, until later realizations with a dynamic group were proposed [KY05], allowing users to be added on the fly. There has been a proposition to use attributes-like properties by Khader [Kha07], however only the identity of the signer stays hidden during a signature as in Group Signature, while the attributes used to sign will be known to any verifier.

Ring Signatures. In a Ring Signature scheme [RST01], the group is dynamically formed at Signature time by the signer, who arbitrarily choses public keys of other "potential" signers to create an ad-hoc group for which the signature is valid. Verifiers then access the public keys used in the signature, and can only learn that the signature was indeed performed by someone from that group. Ring Signature is extremely flexible, as no key generation from a central authority step is necessary and users can join-in freely without management. This also comes at a cost, as user traceability and accountability is almost impossible. Due to this condition, some suggestions have been proposed, for example including a tag to allow identification by an authority [BCC+15], or allowing the linkability between different signatures by the same person [LWW04]. Nonetheless, Ring Signatures are oriented rather towards applications that focus on anonymity and flexibility of join, for example e-voting, rather than fine-grained key management and delegation for users.

Attribute-Based Signature (ABS). Attribute-Based Signature was introduced in [MPR11], as the signature version of Attribute-Based Encryption (ABE) [GPSW06]. They define what one can expect as unforgeability for ABS: one is unable to produce a convincing signature for any policy he wouldn't satisfy. They also introduce privacy (or anonymity) for ABS, where any verifier does not learn anything on the identity nor the attributes of the signer when seeing a signature, except that the signature is valid or not, with respect to the claimed policy.

Our constructions are based on Okamoto and Takashima's [OT11] original work in the Dual-Pairing Vector Space (DPVS) framework, which is still the basis for their recent work [DOT19]. However, while most of previous works are based on the DLIN assumption or variants, ours are based on the SXDH assumption. Along with Attrapadung *et al.*, they focused on the expressivity of the policy [SAH16, SKAH18].

Traceable ABS. Traceability, as a feature of ABS, has been studied by different authors and with different applications in mind. A first approach is the tracing of signers provided

in [EGK14], however without delegation. An approach which focuses on delegation has been proposed by [DGM18] to trace the full path of the intermediary authorities that delivered the keys to the final signers. For efficiency reasons, we decided to exploit a Linearly-Homomorphic signature scheme [HPP20], while more classical approaches use public-key encryption to encrypt the identity of the signer, together with NIZK proofs for consistency guarantees.

Hierarchical ABS. As far as delegation in ABS is concerned, very few works have been conducted. A line of work from Manulis *et al.* [DGM18, GM19] explores a Hierarchical ABS with a focus on the management of intermediate authorities, and the delegation keeps track of the delegation path containing all the authorities that participated in the creation of someone's key. While this gives a strong granularity to identify the keys, especially during tracing, it also means that key and signature sizes are linear in the number of attributes *and* the length of the delegation path. There bas been no intermediary solution giving more control over secret keys for delegation to users, to the best of our knowledge.

2.3 Contributions

As a first step to construct our ABE and ABS with delegation, we go back from Okamoto and Takashima's original constructions [OT12a, OT11]. In both cases, we prove a scheme with delegation and adaptive security under the SXDH assumption in Chapter 5. The common framework of both these schemes, the Dual-Pairing Vector Spaces (DPVS), is presented in chapter 4.

2.3.1 Dual Pairing Vector Spaces

Our contribution on DPVS is a compilation of the main tools from DPVS for indistinguishability in proofs by game, presented as practical theorems under the DDH assumption. Since the original work of Okamoto-Takashima are cut for their own construction, with few reusability for new constructions, we present new generic tools that can be adapted to any proof in the DPVS framework. The results can be found in Section 4.4, with a quick recap table in Figure 4.1. We also adapt an existing theorem for static attributes into a new one that allows an unbounded universe for the attributes (see Theorem 7).

2.3.2 Attribute-Based Encryption

We start with delegatable KP-ABE with some additional attribute-hiding property in the ciphertext to obtain undetectable tracing. To this aim, we first detail one of the main limitation we have to overcome in order to get delegation and traceability: with the original approach of [GPSW06], attributes associated to the ciphertext are explicitly stated as elements in the ciphertext. Removing some attributes can thus allow to single out specific private keys, but this is a public process, and thus incompatible with any tracing procedure, that would then be detectable by the adversary. The alternative would be to use the semantic security of the scheme to send different ciphertexts depending on the possible attributes of the traitor [LT18], but this approach drastically increases the size of the keys, as the key must be fully duplicated for each tracing attribute. To avoid that, **our first contribution on ABE** is the new primitive: Switchable-Attribute Key-Policy Attribute-Based Encryption (SA-KP-ABE), where one can invalidate some attributes in the ciphertext in an indistinguishable way. More precisely, we will bring new properties to the attributes in ciphertexts (for undetectable tracing) but also symmetrically to the leaves in keys (for anonymity).

In a SA-KP-ABE scheme, attributes in a ciphertext and leaves in an access-tree $\mathcal{T}$ defining the policy in a key can be switched in two different states: Attributes can be set to valid or invalid in a ciphertext at encryption time, using a secret encryption key. We then denote $\Gamma = \Gamma_v \cup \Gamma_i$, the

Feature	[OT12a]	[LW15]	[CGW18]	Ours
Security	Adaptive	Adaptive	Adaptive	Adaptive
Assumptions	DLIN	q-type	XDLIN	SXDH
Construction type	CP/KP ABE	CP/KP ABE	IPE	KP ABE
Delegation	✓	×	×	✓
Traceability	×	✓	×	✓

Figure 2.4: Comparison with Related Work

set of attributes for a ciphertext, as the disjoint union of valid and invalid attributes; Leaves can be set to passive or active in the access-tree in a key at key generation time, using the master secret key. We also denote $\mathcal{L} = \mathcal{L}_p \cup \mathcal{L}_a$, the set of leaves, as the disjoint union of passive and active leaves. A set of valid/invalid attributes $\Gamma = \Gamma_v \cup \Gamma_i$ is accepted by an access-tree $\mathcal{T}$ with passive/active leaves $\mathcal{L} = \mathcal{L}_p \cup \mathcal{L}_a$, if the tree $\mathcal{T}$ is accepting when all the leaves in $\mathcal{L}$ associated to an attribute in Γ are set to True, except if the leaf is active (in $\mathcal{L}_a$) and the associated attribute invalid (in Γ_i). As already presented above, passive/active leaves in $\mathcal{L}$ are decided during the Key Generation procedure by the authority, using the master secret key. Then the keys are given to the users. During the Encryption procedure, a ciphertext is generated for attributes in Γ, but one might specify some attributes to be invalid by using a secret tracing key, which virtually and secretly switches some active leaves to False. Passive leaves are not impacted by invalid attributes.

A second contribution on ABE is a concrete and efficient instantiation of SA-KP-ABE, with security proofs under the SXDH assumption. We eventually explain how one can deal with delegatable and traceable KP-ABE from such a primitive. As shown on Figure 2.4, our scheme is the first one that combines explicitly both delegation and traceability of keys for KP-ABE.

Discussions. Our setting bears common characteristics with modern KP-ABE approaches, but with major differences. First, Waters [Wat09] introduced the Dual System Encryption (DSE) technique, to improve the security level of KP-ABE, from selective security in [GPSW06] to adaptive security. In DSE, keys and ciphertexts can be set *semi-functional*, which is in the same vein as our active leaves in keys and invalid attributes in ciphertexts. However, DSE solely uses semi-functional keys and ciphertexts during the simulation, in the security proof, while our construction exploits them in the real-life construction. The security proof thus needs another layer of tricks.

Second, the attribute-hiding notions are strong properties that have been well studied in different IPE works. However, one does not need to achieve such a strong result for tracing: Our (Distinct) Attribute-Indistinguishability is properly tailored for KP-ABE and tracing.

Finally, we detail the advantage of our solution over a generic KEM approach that would combine a Delegatable KP-ABE and a black-box traitor-tracing scheme. This generic solution works if one is not looking for optimal bounds on collusion-resistance during tracing: The main issue with such a use of two independent schemes is that for each user, the KP-ABE key and the traitor-tracing key are not linked. As a consequence, the encryptions of the ABE part and the tracing part are done independently. The colluding users can all try to defeat the traitor tracing without restriction: the collusion-resistance for tracing in the global scheme will exactly be the collusion-resistance of the traitor tracing scheme. On the other hand, our construction will leverage the collusion-resistance of KP-ABE to improve the collusion-resistance of tracing: only players non-revoked by the KP-ABE part can try to defeat the traitor tracing part. Hence, during tracing, one can revoke arbitrary users thanks to the policy/attributes part. This allows to lower the number of active traitors, possibly keeping them below the collusion-resistance of the traitor-tracing scheme, so that tracing remains effective.

Feature	[OT11]	[GM19]	Ours
Assumptions	DLIN	q-type + SXDH	SXDH
Delegation	×	✓	✓
Traceability	✓	✓	✓
Signature size	Linear in attributes	Linear in attributes and authorities	Linear in attributes

Figure 2.5: Comparison with Related Work

2.3.3 Attribute-Based Signature

We move on to our ABS contributions, where we propose and prove two constructions that build on the one adapted from Okamoto-Takashima's work [OT11]:

- **Our contribution on ABS** is a scheme that improves the previous ABS construction with different kinds of delegation: delegation of attributes and delegation of policies;
- The second scheme adds traceability to the initial ABS.

All these constructions are existentially unforgeable under the following two assumptions: the SXDH assumption in the standard model, and the collision-resistance of some hash functions. Our first and second constructions are also perfectly anonymous, while the third one is computationally anonymous under a close variant of the DDH assumption. The traceability of our last construction stands in the Random Oracle Model (ROM), and relies on the security of a Linearly-Homomorphic signature scheme, the simulation-extractability of some non-interactive zero-knowledge proof (NIZK) and the soundness of some other NIZK. We exploit the scheme from [HPP20], whose security is proven in the generic bilinear group model.

Discussions. Our three schemes behave very closely regarding performances. The keys and signatures are linear in the number of attributes involved, with performances comparable to [OT11]. One of the main advantage of the DPVS framework is to rely directly on the hardness assumption which simplifies the protocol, instead of other approaches that need to run more complex (NIZK) as subroutines, which were used as a way to prove the validity of the signature while guaranteeing the anonymity of the signer. In particular, we note that the intrication between delegation and traceability made by [GM19] makes it impossible to have delegation without NIZK, while our modular approach to both features does not encounter this phenomenon.

We separated our construction featuring delegation from our construction featuring tracing so we could underline the specificities of each to the reader, however we note that these new constructions are both fully compatible. One could then make an ABS with two kinds of delegation, and traceability, by putting together each of the specific construction blocks of our constructions. This would result in a scheme with existential unforgeability under the SXDH assumption, but computational anonymity under a DDH-like assumption. We note that the tracing would only trace back to the original key handed to a user, even when tracing a signature made with a delegated key. The linear-only property of the signature scheme ensures the same scalar w_{id} is in all the delegated keys and signatures originated from the same initial signing key, but still with computational anonymity, and unlinkability. This thus addresses the case where a user delegates his signing rights to multiple devices, with various restrictions: whatever the key used for signing, tracing will target the initial user.

Chapter

3 Definitions

Chapter content

3.1 Hardness Assumptions

We will make use of a pairing-friendly setting $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$, with a bilinear map e from $\mathbb{G}_1 \times \mathbb{G}_2$ into $\mathbb{G}_t$, where G_1 (respectively G_2) is a generator of $\mathbb{G}_1$ (respectively $\mathbb{G}_2$). We will use additive notation for $\mathbb{G}_1$ and $\mathbb{G}_2$, and multiplicative notation in $\mathbb{G}_t$. The Decisional Diffie-Hellman (DDH) Assumption, which is at the core of our work, is defined as follows:

Definition 1 (Decisional Diffie-Hellman Assumption) The DDH assumption in group $\mathbb{G}$, of prime order q with generator G, states that no algorithm can efficiently distinguish the two distributions, $a, b, c \xleftarrow{\$} \mathbb{Z}_q$,

$$\mathcal{D}_0 = \{(a \cdot G, b \cdot G, ab \cdot G)\} \qquad \mathcal{D}_1 = \{(a \cdot G, b \cdot G, c \cdot G)\}$$

And we will denote by $\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}(T)$ the best advantage an algorithm can get in distinguishing the two distributions within time bounded by T. Eventually, we will make the following more general Symmetric eXternal Diffie-Hellman (SXDH) Assumption which makes the DDH assumptions in both $\mathbb{G}_1$ and $\mathbb{G}_2$. Then, we define $\mathsf{Adv}^{\mathsf{sxdh}}_{\mathcal{G}}(T) = \max\{\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(T), \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(T)\}$.

Furthermore, for our proofs, we will sometimes use the following DSDH assumption, which is equivalent to the DDH assumption:

Definition 2 (Decisional Separation Diffie-Hellman Assumption) The DSDH assumption in $\mathbb{G}$, of prime order q with generator G, between two constant values x, y, states that no algorithm can efficiently distinguish the two distributions, where $a, b \xleftarrow{\$} \mathbb{Z}_q$,

$$\mathcal{D}_x = \{(a \cdot G, b \cdot G, (ab + x) \cdot G)\} \qquad \mathcal{D}_y = \{(a \cdot G, b \cdot G, (ab + y) \cdot G)\}$$

As $c + x$ and $c + y$ are perfectly indistinguishable for a random c, then the best advantage an algorithm can get in distinguishing the two distributions within time T is upper-bounded by $2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}(T)$.

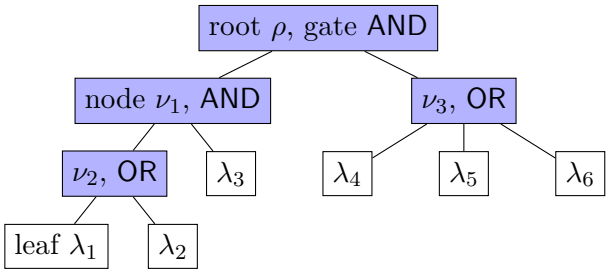

Figure 3.1: Example of an access-tree with all the basic notions. Each leaf λ_i is associated with an attribute in an universe $\mathcal{U}$. In future figures, we will skip the nodes and leaves names, and directly put their associated value: gates for nodes, attributes for leaves.

3.2 Access-Trees

Most of our work uses access-trees to express the policies used in our schemes, either for decryption or signature. We explain here what access-trees are and how to label access-trees as a secret sharing. We conclude with a small section on dual-trees, which is used for our signature scheme construction.

3.2.1 Definition of a Policy

Access Trees. As in the seminal paper of Goyal *et al.* [GPSW06], we will consider an access-tree $\mathcal{T}$ to model the policy on attributes in an unbounded universe $\mathcal{U}$, but with only AND and OR gates instead of more general threshold gates: an AND-gate being an n-out-of-n gate, whereas an OR-gate is a 1-out-of-n gate. This is also a particular case of the more general LSSS technique. Nevertheless, such an access-tree with only AND and OR gates is as expressive as with any threshold gates or LSSS.

For any monotonic policy, we define our access-tree in the following way: $\mathcal{T}$ is a rooted labeled tree from the root ρ, with internal nodes associated to AND and OR gates and leaves associated to attributes. More precisely, for each leaf $\lambda \in \mathcal{L}$, $A(\lambda) \in \mathcal{U}$ is an attribute, and any internal node $\nu \in \mathcal{N}$ is labeled with a gate $G(\nu) \in \{\mathsf{AND}, \mathsf{OR}\}$ as an AND or an OR gate to be satisfied among the children in $\mathsf{children}(\nu)$. We will implicitly consider that any access-tree $\mathcal{T}$ is associated to the attribute-labeling A of the leaves and the gate-labeling G of the nodes. For any leaf $\lambda \in \mathcal{L}$ of $\mathcal{T}$ or internal node $\nu \in \mathcal{N}\backslash\{\rho\}$, the function parent links to the parent node: $\nu \in \mathsf{children}(\mathsf{parent}(\nu))$ and $\lambda \in \mathsf{children}(\mathsf{parent}(\lambda))$. We illustrate these notions in Figure 3.1

On a given list $\Gamma \subseteq \mathcal{U}$ of attributes, each leaf $\lambda \in \mathcal{L}$ is either satisfied (considered or set to True), if $A(\lambda) \in \Gamma$, or not (ignored or set to False) otherwise:. Then, an internal node ν is

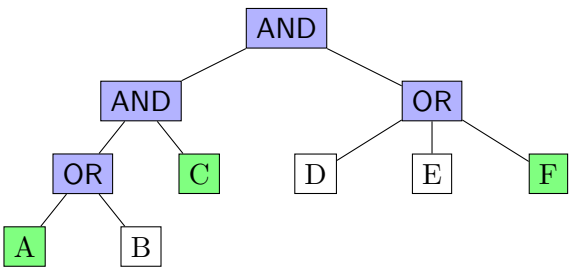

Figure 3.2: Example of an evaluation of an access-tree with the set $\Gamma = \{A, C, F\}$ where $\mathcal{T}(\Gamma) = 1$. Leaves with an attribute corresponding in Γ are in green and are considered as True to evaluate their parents, other leaves are considered False.

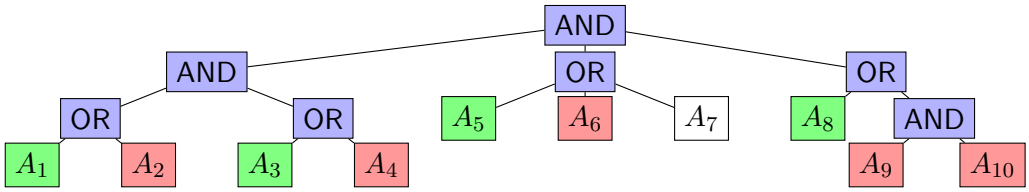

Figure 3.3: Example of an access-tree with two different evaluation pruned trees: in green for the set of attributes $\Gamma_1 = \{A_1, A_3, A_5, A_8\}$, in red for $\Gamma_2 = \{A_2, A_4, A_6, A_9, A_{10}\}$.

satisfied if all children (AND-gate) or at least one of the children (OR-gate) are satisfied.

We add two notations to formalize what we explained in the above paragraph: we denote $\mathcal{T}_\nu$ the subtree rooted at the node ν, and $\mathcal{L}_\Gamma$ the restriction of $\mathcal{L}$ to the satisfied leaves in the tree $\mathcal{T}$. A leaf $\lambda \in \mathcal{L}$ is satisfied if $\lambda \in \mathcal{L}_\Gamma$ then, recursively, $\mathcal{T}_\nu$ is satisfied if the AND/OR-gate associated to ν via $G(\nu)$ is satisfied with respect to status of the children in $\mathsf{children}(\nu)$: we note $\mathcal{T}_\nu(\Gamma) = 1$ when the subtree is satisfied, and 0 otherwise. In particular, the full access-tree is satisfied by Γ if $\mathcal{T}_\rho(\Gamma) = 1$. See Figure 3.2 for a full example.

We detail here the rules to evaluate leaves and nodes depending on their attributes and gates:

$$\begin{array}{lll} \mathcal{T}_\lambda(\Gamma) = 1 & \text{iff } \lambda \in \mathcal{L}_\Gamma & \text{for any leaf } \lambda \in \mathcal{L} \\ \mathcal{T}_\nu(\Gamma) = 1 & \text{iff } \forall \kappa \in \mathsf{children}(\nu), \mathcal{T}_\kappa(\Gamma) = 1 & \text{when } G(\nu) = \mathsf{AND} \\ \mathcal{T}_\nu(\Gamma) = 1 & \text{iff } \exists \kappa \in \mathsf{children}(\nu), \mathcal{T}_\kappa(\Gamma) = 1 & \text{when } G(\nu) = \mathsf{OR} \end{array}$$

Evaluation Pruned Trees. In the above definition, we considered an access-tree $\mathcal{T}$ and a set Γ of attributes, with the satisfiability $\mathcal{T}(\Gamma) = 1$ where the predicate defined by $\mathcal{T}$ is true when all the leaves $\lambda \in \mathcal{L}_\Gamma$ are set to True. We will consider a pruned version of $\mathcal{T}$ for Γ basically as a skeleton with only the necessary True leaves to evaluate the internal nodes up to the root. Formally, a Γ-evaluation pruned tree $\mathcal{T}' \subset \mathcal{T}$ is a pruned version of $\mathcal{T}$, where one children only is kept to OR-gate nodes, down to the leaves, so that $\mathcal{T}'(\Gamma) = 1$.

We will denote $\mathsf{EPT}(\mathcal{T}, \Gamma)$ the set of all the evaluation pruned trees of $\mathcal{T}$ with respect to Γ. $\mathsf{EPT}(\mathcal{T}, \Gamma)$ is non-empty if and only if $\mathcal{T}(\Gamma) = 1$.

Figure 3.3 gives an illustration of such an access-tree for a policy: when the colored leaves $\{\lambda_1, \lambda_3, \lambda_5, \lambda_8, \lambda_9, \lambda_{10}\}$ are True, the tree is satisfied, and there are two possible evaluation pruned trees: down to the leaves $\{\lambda_1, \lambda_3, \lambda_5, \lambda_8\}$ or $\{\lambda_1, \lambda_3, \lambda_5, \lambda_9, \lambda_{10}\}$.

Partial Order on Policies. We will introduce a notion of delegation to create more restrictive access-tree from an existing one. In order to do that, we define the following partial order, for any access-tree $\mathcal{T}, \mathcal{T}'$: $\mathcal{T}' \leq \mathcal{T}$, if and only if for any subset Γ of attributes, $\mathcal{T}'(\Gamma) = 1 \Longrightarrow \mathcal{T}(\Gamma) = 1$. In our case of access-trees, a more restrictive access-tree is, for each node ν:

- if $G(\nu) = \mathsf{AND}$, one or more children are added (*i.e.*, more constraints);
- if $G(\nu) = \mathsf{OR}$, one or more children are removed (*i.e.*, less flexibility);
- the node ν is moved one level below as a child of an AND-gate at node ν', with additional sub-trees as children to this AND-gate (*i.e.*, more constraints).

We illustrate the last rule, with a simple example in Figure 3.4.

3.2.2 Labeling of Access-Trees

Labeled Access-Trees. We will label access-trees with integers so that some labels on the leaves will be enough/necessary (according to the policy) to recover the labels above, up to the root. An example of labeling in $\mathbb{Z}/7\mathbb{Z}$ is presented on Figure 3.5, and we show how to do such a labeling in our next definition.

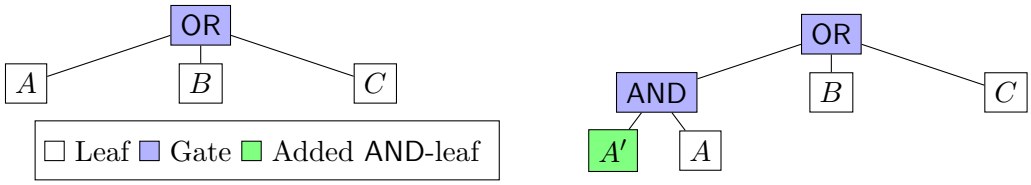

Figure 3.4: Access-tree (left-side) and delegated-tree (right-side) where the leaf associated with attribute A is changed into an AND-gate with a new child leaf associated with attribute A'

Definition 3 (Random y-Labeling) For an access-tree $\mathcal{T}$ and any $y \in \mathbb{Z}_p$, the probabilistic algorithm $\Lambda_y(\mathcal{T})$ sets $a_\rho \leftarrow y$ for the root, and then in a top-down manner, for each internal node ν, starting from the root:

- if $G(\nu) = \mathsf{AND}$, with n children, a random n-out-of-n sharing of a_ν is associated to each child: $(a_\kappa)_\kappa \xleftarrow{\$} \mathbb{Z}_p, \forall \kappa \in \mathsf{children}(\nu)$, such that $a_\nu = \sum_{\kappa \in \mathsf{children}(\nu)} a_\kappa \bmod p$;
- if $G(\nu) = \mathsf{OR}$, with n children, each child is associated to a_ν: $\forall \kappa \in \mathsf{children}(\nu), a_\kappa = a_\nu$.

Algorithm $\Lambda_y(\mathcal{T})$ outputs $\Lambda_y = (a_\lambda)_{\lambda \in \mathcal{L}}$, for all the leaves $\lambda \in \mathcal{L}$ of the tree $\mathcal{T}$.

Due to the linear nature of labelings, from any y-labeling $(a_\lambda)_\lambda$ of the tree $\mathcal{T}$, and a random z-labeling $(b_\lambda)_\lambda$ of $\mathcal{T}$, the sum $(a_\lambda + b_\lambda)_\lambda$ is a random $(y+z)$-labeling of $\mathcal{T}$. In particular, from any y-labeling $(a_\lambda)_\lambda$ of $\mathcal{T}$, and a random zero-labeling $(b_\lambda)_\lambda$ of $\mathcal{T}$, the values $c_\lambda \leftarrow a_\lambda + b_\lambda$ provide a random y-labeling of $\mathcal{T}$.

Labels on leaves are a secret sharing of the root that allows reconstruction of the secret if and only if the policy is satisfied, as explained below:

Properties of Labelings. For an acceptable set of attributes Γ for $\mathcal{T}$ and a labeling Λ_y of $\mathcal{T}$ for a random y, given only $(a_\lambda)_{\lambda \in \mathcal{L}_\Gamma}$, one can reconstruct $y = a_\rho$. Indeed, as $\mathcal{T}(\Gamma) = 1$, we use an evaluation pruned tree $\mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma)$. Then, in a bottom-up way, starting from the leaves, one can compute the labels of all the internal nodes, up to the root.

On the other hand, when $\mathcal{T}(\Gamma) = 0$, with a random labeling Λ_y of $\mathcal{T}$ for a random y, given only $(a_\lambda)_{\lambda \in \mathcal{L}_\Gamma}$, y is unpredictable: for any $y, y' \in \mathbb{Z}_p$, $\mathcal{D}_y$ and $\mathcal{D}_{y'}$ are perfectly indistinguishable, where $\mathcal{D}_y = \{(a_\lambda)_{\lambda \in \mathcal{L}_\Gamma}, (a_\lambda)_\lambda \xleftarrow{\$} \Lambda_y(\mathcal{T})\}$. Intuitively, given $(a_\lambda)_{\lambda \in \mathcal{L}_\Gamma}$, as $\mathcal{T}(\Gamma) = 0$, one can complete the labeling so that the label of the root is any y.

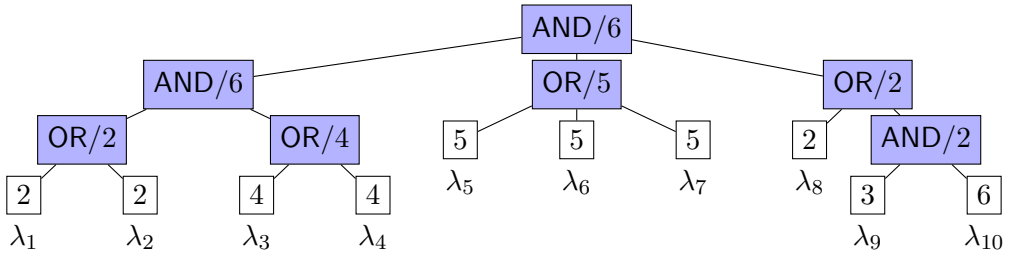

Figure 3.5: Example of a 6-labeling in $\mathbb{Z}/7\mathbb{Z}$. The value of the AND gates is the sum of the value of their children, the value of the OR gates is the same as the value of their children.

3.3 Attribute-Based Encryption (ABE)

3.3.1 Definition of ABE

We will define KP-ABE in the same way that the seminal work from [GPSW06], with access-trees to define policies in the keys. We will also use the usual description of Key Encapsulation Mechanism, that consists in generating an ephemeral key K and its encapsulation C. The encryption of the actual message under the key K, using a symmetric encryption scheme is then appended to C. We will however call C the *ciphertext*, and K the *encapsulated key* in C.

$\mathsf{Setup}(1^\kappa)$**.** From the security parameter κ, the algorithm defines all the global parameters PK and the master secret key MK;

$\mathsf{KeyGen}(\mathsf{MK}, \mathcal{T})$**.** For a master secret key MK and an access-tree $\mathcal{T}$, the algorithm outputs a private key $\mathsf{dk}_\mathcal{T}$;

$\mathsf{Encaps}(\mathsf{PK}, \Gamma)$**.** For a list Γ of attributes and global parameters PK, the algorithm generates the ciphertext C and an encapsulated key K;

$\mathsf{Decaps}(\mathsf{dk}_\mathcal{T}, C)$**.** Given the private key $\mathsf{dk}_\mathcal{T}$ and the ciphertext C, the algorithm outputs the encapsulated key K.

For correctness, the Decaps algorithm should output the encapsulated key K if and only if C has been generated for a set Γ that satisfies the policy $\mathcal{T}$ of the decryption key $\mathsf{dk}_\mathcal{T}$: $\mathcal{T}(\Gamma) = 1$.

Delegation. A major feature in [GPSW06] is the delegation of decryption keys, in which a user with a decryption key dk corresponding to an access-tree $\mathcal{T}$ can compute a new decryption key corresponding to any access-tree $\mathcal{T}' \leq \mathcal{T}$. There is thus the additional algorithm:

$\mathsf{Delegate}(\mathsf{dk}_\mathcal{T}, \mathcal{T}')$**.** Given a key $\mathsf{dk}_\mathcal{T}$, generated from either the KeyGen or the Delegate algorithms, for a policy $\mathcal{T}$ and a more restrictive policy $\mathcal{T}' \leq \mathcal{T}$, the algorithm outputs a decryption key $\mathsf{dk}_{\mathcal{T}'}$.

3.3.2 Security Model for ABE

We define the traditional notion of indistinguishability, with only KeyGen-queries. Then, we extend it to handle delegation queries in a new notion called Delegation-Indistinguishability: if one can ask several more restrictive delegations from an access-tree $\mathcal{T}$, one should not be able to distinguish an encapsulated key in a ciphertext under a non-trivial list of attributes, according to the obtained delegated keys only. Note that this definition allows for an adversary to make delegation requests on keys that are delegated keys themselves, without limit.

Definition 4 (Indistinguishability) IND-security for KP-ABE is defined by the following game:

Initialize: The challenger runs the Setup algorithm of KP-ABE and gives the public parameters PK to the adversary;

$\mathsf{OKeyGen}(\mathcal{T})$**:** The adversary is allowed to issue KeyGen-queries for any access-tree $\mathcal{T}$ of its choice, and gets back the decryption key $\mathsf{dk}_\mathcal{T}$;

$\mathsf{RoREncaps}(\Gamma)$**:** The adversary submits one real-or-random encapsulation query on a set of attributes Γ. The challenger asks for an encapsulation query on Γ and receives (K_0, C). It also generates a random key K_1. It eventually flips a random coin b, and outputs (K_b, C) to the adversary;

Finalize(b'): The adversary outputs a guess b' for b. If for some access-tree $\mathcal{T}$ asked to the $\mathsf{OKeyGen}$-oracle, $\mathcal{T}(\Gamma) = 1$, on the challenge set Γ, $\beta \xleftarrow{\$} \{0,1\}$, otherwise one sets $\beta = b'$. One outputs β.

The advantage of an adversary $\mathcal{A}$ in this game is defined as

$$\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}) = \Pr[\beta = 1 | b = 1] - \Pr[\beta = 1 | b = 0].$$

As usual, we do not consider the trivial attack where the adversary queries a key that give direct access to the challenge ciphertext. One could of course consider **Chosen-Ciphertext security**, where the adversary could have access to some decryption oracles, without the decryption key itself. We note that we present the **Adaptive-Set Security** version of the Indistinguishability game, which is standard since the introduction of the Dual System Encryption by Waters [Wat09], meaning the challenge set Γ can be chosen at the time of the challenge. Before that, most schemes were realized with **Selective-Set security**, where the adversary declares Γ at the initialization step, as in [GPSW06].

We now extend the definition to include limitless delegation:

Definition 5 (Delegation-Indistinguishability) $\mathsf{Del\text{-}IND}$ security for $\mathsf{KP\text{-}ABE}$ is defined by the following game between the adversary and a challenger:

Initialize: The challenger runs the Setup algorithm of $\mathsf{KP\text{-}ABE}$ and gives the public parameters PK to the adversary;

Oracles: The following oracles can be called in any order and any number of times, except for $\mathsf{RoREncaps}$ which can be called only once.

- $\mathsf{OKeyGen}(\mathcal{T})$**:** to model KeyGen-queries for any access-tree $\mathcal{T}$. It generates the decryption key but only outputs the index k of the key;
- $\mathsf{ODelegate}(k, \mathcal{T}')$**:** to model $\mathsf{Delegate}$-queries for any more restrictive access-tree $\mathcal{T}' \leq \mathcal{T}$, for the k-th generated decryption key for $\mathcal{T}$. It generates the decryption key but only outputs the index k' of the new key;
- $\mathsf{OGetKey}(k)$**:** the adversary gets back the k-th decryption key generated by $\mathsf{OKeyGen}$ or $\mathsf{ODelegate}$ oracles;
- $\mathsf{RoREncaps}(\Gamma)$**:** the challenge real-or-random encapsulation query on a set of attributes Γ is asked once only. The challenger asks for an encapsulation query on Γ and receives (K_0, C). It also generates a random key K_1. It eventually flips a random coin b, and outputs (K_b, C) to the adversary;

Finalize(b'): The adversary outputs a guess b' for b. If for some access-tree $\mathcal{T}$ corresponding to a key asked to the $\mathsf{OGetKey}$-oracle, $\mathcal{T}(\Gamma) = 1$, on the challenge set Γ, $\beta \xleftarrow{\$} \{0,1\}$, otherwise one sets $\beta = b'$. One outputs β.

The advantage of an adversary $\mathcal{A}$ in this game is defined as

$$\mathsf{Adv}^{\mathsf{del\text{-}ind}}(\mathcal{A}) = \Pr[\beta = 1 | b = 1] - \Pr[\beta = 1 | b = 0].$$

This Delegation-Indistinguishability is definitely stronger than basic Indistinguishability as the adversary can ask for an $\mathsf{OGetKey}(k)$-query right after the $\mathsf{OKeyGen}(\mathcal{T})$-query that provides index k to get the decryption key for $\mathcal{T}$. The counter k helps represent the different queries made by the adversary where he does not get the decryption key back, but instead serve to make future delegations.

3.4 Attribute-Based Signature (ABS)

3.4.1 Definition of ABS

We take the definition of Attribute-Based Signatures (ABS) from the original work of Maji *et al.* [MPR11], with attributes in the keys and access-trees as policies in the signatures:

$\mathsf{Setup}(1^\kappa)$**.** From the security parameter κ, the algorithm defines all the global parameters PK and the master secret key MK;

$\mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}, \Gamma)$**.** For a master secret key MK, an identity id and a list of attributes Γ, the algorithm outputs a private key $\mathsf{SK}_{\mathsf{id},\Gamma}$ specific to the user id and the set of attributes Γ;

$\mathsf{Sig}(\mathsf{SK}_{\mathsf{id},\Gamma}, m, \mathcal{T})$**.** For a private key $\mathsf{SK}_{\mathsf{id},\Gamma}$, on a set of attributes Γ, a message m and an access-tree $\mathcal{T}$ that accepts Γ, the algorithm outputs a signature σ;

$\mathsf{Verif}(\mathsf{PK}, m, \mathcal{T}, \sigma)$**.** Given the public parameters PK, a signature σ for a message m under an access-tree $\mathcal{T}$, the algorithm outputs 1 for accept or 0 for reject.

For correctness, the Verif algorithm should output 1 with overwhelming probability on $(\sigma, m, \mathcal{T})$ if σ has been generated on m and $\mathcal{T}$, with a private key $\mathsf{SK}_{\mathsf{id},\Gamma}$ that has been generated from the KeyGen algorithm associated on a set Γ accepted by $\mathcal{T}$.

3.4.2 Security Model for ABS

As for any signature scheme, one should not be able to produce an accepted signature without an appropriate signing key. In ABS, it translates as not being able to produce a signature under an access-tree $\mathcal{T}$ if one does not own the appropriate attributes to fulfill it:

Definition 6 (Existential Unforgeability) EUF for ABS is defined by the following game between the adversary and a challenger:

Initialize: The challenger runs the Setup algorithm of ABS and gives the public parameters PK to the adversary;

Oracles: The following oracles can be called in any order and any number of times.

- $\mathsf{OKeyGen}(\mathsf{id}, \Gamma)$**:** to model KeyGen-queries for any identity id and any set of attributes Γ of its choice, the adversary gets back the key $\mathsf{SK}_{\mathsf{id},\Gamma}$;
- $\mathsf{OSig}(\mathsf{id}, m, \mathcal{T})$**:** to model Sig-queries for any identity id and under any access-tree $\mathcal{T}$ of its choice for a message m, the adversary gets back the signature σ;

Finalize(b'): The adversary outputs a forgery $(m', \mathcal{T}', \sigma')$. If for some attribute set Γ asked to the OKeyGen-oracle, $\mathcal{T}'$ accepts Γ, or if the adversary queried OSig on $(m', \mathcal{T}')$, one outputs 0. Otherwise one outputs $\mathsf{Verif}(\mathsf{PK}, m', \mathcal{T}', \sigma')$.

The advantage $\mathsf{Adv}^{\mathsf{euf}}(\mathcal{A})$ of an adversary $\mathcal{A}$ in this game is defined as the probability to output 1.

As usual, the Finalize-step excludes trivial attacks, where the adversary owns a key able to generate an acceptable signature or just forwards a query asked to the signing oracle. We present a version with delegation, akin to the ABE with delegation, in a later section with our contributions.

Another security notion that should also be satisfied by an ABS scheme is that a signature generated for a given access-tree should be independent of the user, which is usually referred to as anonymity. Maji *et al.* [MPR11] deals with perfect anonymity, but we will be more flexible, in order not to exclude traceability:

Definition 7 (Anonymity) An ABS scheme is said anonymous if, for any $(\mathsf{PK}, \mathsf{MK}) \xleftarrow{\$} \mathsf{Setup}$, any message m, any identities $\mathsf{id}_0, \mathsf{id}_1$, any attribute sets Γ_0, Γ_1, any signing keys $\mathsf{SK}_0 \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}_0, \Gamma_0)$, $\mathsf{SK}_1 \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}_1, \Gamma_1)$, and any access-tree $\mathcal{T}$ that accepts both Γ_0 and Γ_1, then the distributions of the signatures generated by $\mathsf{Sig}(\mathsf{SK}_0, m, \mathcal{T})$ and $\mathsf{Sig}(\mathsf{SK}_1, m, \mathcal{T})$ are indistinguishable.

Indistinguishability can be perfect, statistical or computational, which leads to perfect, statistical or computational anonymity.

Whereas perfect anonymity excludes traceability, computational anonymity may allow the existence of a trapdoor leading to traceability. We will propose both in out later sections.

Chapter

4 Dual Pairing Vector Spaces (DPVS)

Chapter content

In this section, we present the Dual Pairing Vector Spaces (DPVS), that have been proposed for efficient constructions with adaptive security [OT08, LOS+10, OT10, OT12b], like the Dual Systems [Wat09], in prime-order groups under the DLIN assumption. In [LW10], Dual Systems were using pairing on composite order elliptic curves. Then, prime-order groups have been used with the SXDH assumption, in a pairing-friendly setting of primer order, which means that the DDH assumptions hold in both $\mathbb{G}_1$ and $\mathbb{G}_2$ [CLL+13]. In all theses situations, one exploited indistinguishability of sub-groups or sub-spaces. In this section, for the sake of efficiency, we use the SXDH assumption in a pairing-friendly setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$ of primer order q.

We first present the mathematical basis of the framework, which has been conceived by Okamoto and Takashima. However, the approach of Okamoto-Takashima is not appropriate for re-usability, as their problem and sub-problem methodology is specialized for their specific construction. Thus, we introduce a new generic approach to Okamoto and Takashima's results in the form of three ready-to-use generic theorems for DPVS proofs in Section 4.4. A recapitulative table of these theorems is available in Figure 4.1. Finally, we present a new technical result for unbounded universe of attributes in Theorem 7.

4.1 Pairing Vector Spaces

Let us be given any cyclic group $(\mathbb{G} = \langle G \rangle, +)$ of prime order q, denoted additively. We can define the $\mathbb{Z}_q$ vector space of dimension n,

$$\mathbb{G}^n = \{\mathbf{X} = \vec{x} \cdot G \stackrel{\text{def}}{=} (X_1 = x_1 \cdot G, \ldots, X_n = x_n \cdot G) \,|\, \vec{x} \in \mathbb{Z}_q^n\},$$

with the following laws:

$$(X_1, \ldots, X_n) + (Y_1, \ldots, Y_n) \stackrel{\text{def}}{=} (X_1 + Y_1, \ldots, X_n + Y_n)$$
$$a \cdot (X_1, \ldots, X_n) \stackrel{\text{def}}{=} (a \cdot X_1, \ldots, a \cdot X_n)$$

Essentially, all the operations between the vectors in $\mathbb{G}^n$ are applied on the vectors in $\mathbb{Z}_q^n$:

$$\vec{x} \cdot G + \vec{y} \cdot G \stackrel{\text{def}}{=} (\vec{x} + \vec{y}) \cdot G \qquad\qquad a \cdot (\vec{x} \cdot G) \stackrel{\text{def}}{=} (a \cdot \vec{x}) \cdot G$$

where $\vec{x} + \vec{y}$ and $a \cdot \vec{x}$ are the usual internal and external laws of the vector space $\mathbb{Z}_q^n$. For the sake of clarity, vectors will be row-vectors.

If we are using a pairing-friendly setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$, with a bilinear map e from $\mathbb{G}_1 \times \mathbb{G}_2$ into $\mathbb{G}_t$, we can have an additional law between an element $X \in \mathbb{G}_1^n$ and $Y \in \mathbb{G}_2^n$: $X \times Y \stackrel{\text{def}}{=} \prod_i e(X_i, Y_i)$, where $\mathbb{G}_t$ is usually denoted multiplicatively.

Note that if $\mathbf{X} = (X_1, \ldots, X_n) = \vec{x} \cdot G_1 \in \mathbb{G}_1^n$ and $\mathbf{Y} = (Y_1, \ldots, Y_n) = \vec{y} \cdot G_2 \in \mathbb{G}_2^n$:

$$\begin{aligned}(\vec{x} \cdot G_1) \times (\vec{y} \cdot G_2) = \mathbf{X} \times \mathbf{Y} &= \prod_i e(X_i, Y_i) = \prod_i e(x_i \cdot G_1, y_i \cdot G_2) \\ &= \prod_i g_t^{x_i \cdot y_i} = g_t^{\vec{x} \cdot \vec{y}^\top} = g_t^{\langle \vec{x}, \vec{y} \rangle}\end{aligned}$$

where $g_t = e(G_1, G_2)$ and $\langle \vec{x}, \vec{y} \rangle$ is the inner product between vectors $\vec{x}$ and $\vec{y}$.

4.2 Dual Pairing Vector Spaces

We define $\mathcal{E} = (\vec{e}_i)_i$ the canonical basis of $\mathbb{Z}_q^n$, where $\vec{e}_i = (\delta_{i,1}, \ldots, \delta_{i,n})$, with the classical $\delta_{i,j} = 1$ if $i = j$ and $\delta_{i,j} = 0$ otherwise, for $i, j \in \{1, \ldots, n\}$. We can also define $\mathbb{E} = (\mathbf{e}_i)_i$ the canonical basis of $\mathbb{G}^n$, where $\mathbf{e}_i = \vec{e}_i \cdot G = (\delta_{i,j} \cdot G)_j$. More generally, given any basis $\mathcal{B} = (\vec{b}_i)_i$ of $\mathbb{Z}_q^n$, we can define the basis $\mathbb{B} = (\mathbf{b}_i)_i$ of $\mathbb{G}^n$, where $\mathbf{b}_i = \vec{b}_i \cdot G$.

Choosing a random basis $\mathbb{B}$ of $\mathbb{G}^n$ is equivalent to choosing a random invertible matrix $B \stackrel{\$}{\leftarrow} \mathrm{GL}_n(\mathbb{Z}_q)$. Given B, we define $\mathcal{B} = (\vec{b}_i)_i$ as a basis of $\mathbb{Z}_q^n$ (B is essentially the matrix with $\vec{b}_i$ as its i-th row, as $\vec{b}_i = \sum_j B_{i,j} \cdot \vec{e}_j$), and note $\mathcal{B} \leftarrow B \times \mathcal{E}$ the change of basis from the canonical basis $\mathcal{E}$. Then, we can define $\mathbb{B} = (\mathbf{b}_i)_i$ where $\mathbf{b}_i = \vec{b}_i \cdot G$. Thus, $\mathbb{B}$ is the basis of $\mathbb{G}^n$ associated to the matrix B as

$$\mathbf{b}_i = \vec{b}_i \cdot G = \sum_j B_{i,j} \cdot \vec{e}_j \cdot G = \sum_j B_{i,j} \cdot \mathbf{e}_j$$

So we can write $\mathbb{B} = B \cdot \mathbb{E}$

In case of pairing-friendly setting, for a dimension n, we will denote $\mathbb{E} = (\mathbf{e}_i)_i$ and $\mathbb{E}^* = (\mathbf{e}_i^*)_i$ the canonical bases of $\mathbb{G}_1^n$ and $\mathbb{G}_2^n$, respectively:

$$\mathbf{e}_i \times \mathbf{e}_j^* = (\vec{e}_i \cdot G_1) \times (\vec{e}_j \cdot G_2) = g_T^{\langle \vec{e}_i, \vec{e}_j \rangle} = g_T^{\delta_{i,j}}.$$

The same way, if we denote $\mathbb{B} = (\mathbf{b}_i)_i = B \cdot \mathbb{E}$ the basis of $\mathbb{G}_1^n$ associated to a matrix B, and $\mathbb{B}^* = (\mathbf{b}_i^*)_i = B' \cdot \mathbb{E}^*$ the basis of $\mathbb{G}_2^n$ associated to the matrix $B' = (B^{-1})^\top$, as $B \cdot B'^\top = I_n$,

$$\mathbf{b}_i \times \mathbf{b}_j^* = (\vec{b}_i \cdot G_1) \times (\vec{b}'_j \cdot G_2) = g_t^{\langle \vec{b}_i, \vec{b}'_j \rangle} = g_t^{\delta_{i,j}}.$$

$\mathbb{B}$ and $\mathbb{B}^*$ are called *Dual Orthogonal Bases.*

We have seen above the canonical bases $\mathbb{E}$ and $\mathbb{E}^*$ are dual orthogonal bases, but for any random invertible matrix $U \stackrel{\$}{\leftarrow} \mathrm{GL}_n(\mathbb{Z}_q)$, the bases $\mathbb{U}$ of $\mathbb{G}_1^n$ associated to the matrix U and $\mathbb{U}^*$ of $\mathbb{G}_2^n$ associated to the matrix $(U^{-1})^\top$ are *Random Dual Orthogonal Bases.*

A pairing-friendly setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$, with such dual orthogonal bases $\mathbb{U}$ and $\mathbb{U}^*$ of size n, is called a *Dual Pairing Vector Space* (DPVS).

4.3 Change of Basis

4.3.1 Definition

Let us consider the basis $\mathbb{U} = (\mathbf{u}_i)_i$ of $\mathbb{G}^n$ associated to a matrix $U \in \mathrm{GL}_n(\mathbb{Z}_q)$, and the basis $\mathbb{B} = (\mathbf{b}_i)_i$ of $\mathbb{G}^n$ associated to the product matrix BU, for any $B \in \mathrm{GL}_n(\mathbb{Z}_q)$. For a vector $\vec{x} \in \mathbb{Z}_q$, we can define

$$\begin{aligned}(\vec{x})_{\mathbb{B}} &= \sum_i x_i \cdot \mathbf{b}_i = \sum_i x_i \cdot \vec{b}_i \cdot G = \vec{x} \cdot BU \cdot G = (\vec{x} \cdot B) \cdot U \cdot G = \vec{y} \cdot U \cdot G \\ &= \sum_i y_i \cdot \vec{u}_i \cdot G = \sum_i y_i \cdot \mathbf{u}_i = (\vec{y})_{\mathbb{U}} \text{ where } \vec{y} = \vec{x} \cdot B.\end{aligned}$$

Hence, $(\vec{x})_{\mathbb{B}} = (\vec{x} \cdot B)_{\mathbb{U}}$ and $(\vec{x} \cdot B^{-1})_{\mathbb{B}} = (\vec{x})_{\mathbb{U}}$ where we denote $\mathbb{B} \stackrel{\mathsf{def}}{=} B \cdot \mathbb{U}$. For any invertible matrix B, if $\mathbb{U}$ is a random basis, then $\mathbb{B} = B \cdot \mathbb{U}$ is also a random basis. Then, with $B^{-1} = (\vec{b'}_1^{\top}, \ldots, \vec{b'}_n^{\top})$, $\vec{x} = \vec{y} \cdot (\vec{b'}_1^{\top}, \ldots, \vec{b'}_n^{\top})$:

$$\mathbb{B} = B \cdot \mathbb{U}, B' = \begin{pmatrix} \vec{b'}_1 \\ \vdots \\ \vec{b'}_n \end{pmatrix}, \text{ and } (\vec{x})_{\mathbb{B}} = (\vec{y})_{\mathbb{U}} \Longrightarrow \vec{x} = (\langle \vec{y}, \vec{b'}_1 \rangle, \ldots, \langle \vec{y}, \vec{b'}_n \rangle).$$

Let us consider the random dual orthogonal bases $\mathbb{U} = (\mathbf{u}_i)_i$ and $\mathbb{U}^* = (\mathbf{u}_i^*)_i$ of $\mathbb{G}_1^n$ and $\mathbb{G}_2^n$ respectively associated to a matrix U (which means that $\mathbb{U}$ is associated to the matrix U and $\mathbb{U}^*$ is associated to the matrix $(U^{-1})^{\top}$): the bases $\mathbb{B} = B \cdot \mathbb{U}$ and $\mathbb{B}* = (B^{-1})^{\top} \cdot \mathbb{U}^*$ are also dual orthogonal bases:

$$\mathbf{b}_i \times \mathbf{b}_j^* = g_t^{\vec{b}_i \cdot \vec{b'}_j^{\top}} = g_t^{(\sum_l B_{i,l} \cdot \vec{u}_l) \cdot (\sum_k (B^{-1}_{j,k})^{\top} \cdot \vec{u}_k^{*\top})} = g_t^{\sum_{l,k} B_{i,l} B^{-1}_{k,j} \cdot \vec{u}_l \cdot \vec{u}_k^{*\top}} = g_t^{\sum_k B_{i,k} B^{-1}_{k,j}} = g_t^{\delta_{i,j}}$$

as $\forall (k,l), \vec{u}_k \cdot \vec{u}_l = \delta_{k,l}$ since $\mathbb{U}$ and $\mathbb{U}^*$ are orthogonal bases.

4.3.2 Partial Change of Basis

We will often just have to partially change a basis, on a few vectors only: the transition matrix

$$B = (t)_{i_1,\ldots,i_m} = \begin{pmatrix} t_{1,1} & \ldots & t_{1,m} \\ \vdots & & \vdots \\ t_{m,1} & \ldots & t_{m,m} \end{pmatrix}_{i_1,\ldots,i_m}$$

means the $n \times n$ matrix B where

$$B_{i,j} = \delta_{i,j}, \text{ if any } i,j \notin \{i_1,\ldots,i_m\} \qquad B_{i_k,i_\ell} = t_{k,\ell}, \text{ for all } k,\ell \in \{1,\ldots,m\}$$

As a consequence, from a basis $\mathbb{U}$, $\mathbb{B} = B \cdot \mathbb{U}$ corresponds to the basis

$$\mathbf{b}_i = \mathbf{u}_i, \text{ if } i \notin \{i_1,\ldots,i_m\} \qquad \mathbf{b}_{i_k} = \sum_\ell t_{k,\ell} \cdot \mathbf{u}_{i_\ell}, \text{ if } k \notin \{i_1,\ldots,i_m\}$$

As we need to have $\mathbb{B}^* = (B^{-1})^{\top} \cdot \mathbb{U}^*$, we need the dual transition matrix B' to be $B' = (t')_{i_1,\ldots,i_m}$ where $t' = (t^{-1})^{\top}$. Indeed, in such a case, we have

$$\mathbf{b}_i^* = \mathbf{u}_i^*, \text{ if } i \notin \{i_1,\ldots,i_m\} \qquad \mathbf{b}_{i_k}^* = \sum_\ell t'_{k,\ell} \cdot \mathbf{u}_{i_\ell}^*, \text{ if } k \notin \{i_1,\ldots,i_m\}$$

so,

- if both $i, j \notin \{i_1, \ldots, i_m\}$, $\mathbf{b}_i \times \mathbf{b}_j^* = \mathbf{u}_i \times \mathbf{u}_j^* = g_t^{\delta_{i,j}}$;
- if $i = i_k \in \{i_1, \ldots, i_m\}$, but $j \notin \{i_1, \ldots, i_m\}$,

$$\mathbf{b}_i \times \mathbf{b}_j^* = \mathbf{b}_{i_k} \times \mathbf{u}_j^* = \left(\sum_\ell t_{k,\ell} \cdot \mathbf{u}_{i_\ell}\right) \times \mathbf{u}_j^* = \prod_\ell (\mathbf{u}_{i_\ell} \times \mathbf{u}_j^*)^{t_{k,\ell}} = 1$$

- if $i \notin \{i_1, \ldots, i_m\}$, but $j = i_k \in \{i_1, \ldots, i_m\}$,

$$\mathbf{b}_i \times \mathbf{b}_j^* = \mathbf{u}_i \times \mathbf{b}_{i_k}^* = \mathbf{u}_i \times \left(\sum_\ell t'_{k,\ell} \cdot \mathbf{u}_{i_\ell}^*\right) = \prod_\ell (\mathbf{u}_i \times \mathbf{u}_{i_\ell}^*)^{t'_{k,\ell}} = 1$$

- if $i = i_k$ and $j = i_\ell$,

$$\begin{aligned}\mathbf{b}_i \times \mathbf{b}_j^* &= \left(\sum_p t_{k,p} \cdot \mathbf{u}_{i_p}\right) \times \left(\sum_p t'_{\ell,p} \cdot \mathbf{u}_{i_p}^*\right) \\ &= \prod_p (\mathbf{u}_{i_p} \times \mathbf{u}_{i_p}^*)^{t_{k,p} \cdot t'_{\ell,p}} = g_t^{\sum_p t_{k,p} \cdot t'_{\ell,p}} = g_t^{\sum_p t_{k,p} \cdot {t'}_{p,\ell}^\top} = g_t^{\delta_{k,\ell}} = g_t^{\delta_{i,j}}\end{aligned}$$

4.4 Particular Changes of Bases

4.4.1 Diffie-Hellman Tuple in Basis Change

Let us consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, that is either a Diffie-Hellman tuple (*i.e.*, $c = ab \bmod q$) or a random tuple (*i.e.*, $c = ab + \tau \bmod q$, for $\tau \xleftarrow{\$} \mathbb{Z}_q^*$). For any random dual orthogonal bases $\mathbb{U}$ and $\mathbb{U}^*$ associated to the matrices U and $U' = (U^{-1})^\top$, respectively, we can set

$$B = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{1,2} \qquad B' = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{1,2} \qquad \mathbb{B} = B \cdot \mathbb{U} \qquad \mathbb{B}^* = B' \cdot \mathbb{U}^*$$

Note that we can compute $\mathbb{B} = (\mathbf{b}_i)_i$, as we know $a \cdot G_1$ and all the scalars in U:

$$\mathbf{b}_i = \sum_k B_{i,k} \cdot \mathbf{u}_k \qquad \mathbf{b}_{i,j} = \sum_k B_{i,k} \cdot \mathbf{u}_{k,j} = \sum_k B_{i,k} U_{k,j} \cdot G_1 = \sum_k U_{k,j} \cdot (B_{i,j} \cdot G_1).$$

Hence, to compute $\mathbf{b}_i$, one needs all the scalars in U, but only the group elements $B_{i,j} \cdot G_1$, and so G_1 and $a \cdot G_1$. This is the same for $\mathbb{B}^*$, except for the vector $\mathbf{b}_2^*$ as $a \cdot G_2$ is not known. One can thus publish $\mathbb{B}$ and $\mathbb{B}^* \backslash \{\mathbf{b}_2^*\}$.

4.4.2 Indistinguishability of Sub-Spaces (SubSpace-Ind)

Let us again choose B, B' for bases $\mathbb{B}, \mathbb{B}^*$ as :

$$B = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{1,2} \qquad B' = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{1,2} \qquad \mathbb{B} = B \cdot \mathbb{U} \qquad \mathbb{B}^* = B' \cdot \mathbb{U}^*$$

As already remarked, for such a fixed matrix B, if $\mathbb{U}$ is random, so is $\mathbb{B}$ too. We remind that $(\vec{x})_\mathbb{B} = (\vec{x} \cdot B)_\mathbb{U}$, and $(\vec{x})_\mathbb{U} = (\vec{x} \cdot B^{-1})_\mathbb{B}$. Note that $B^{-1} = {B'}^\top$. So, in particular

$$\begin{aligned}(b, c, 0, \ldots, 0)_\mathbb{U} &+ (x_1, x_2, x_3, \ldots, x_n)_\mathbb{B} \\ &= (b, c - ab, 0, \ldots, 0)_\mathbb{B} + (x_1, x_2, x_3, \ldots, x_n)_\mathbb{B} \\ &= (x_1 + b, x_2 + \tau, x_3, \ldots, x_n)_\mathbb{B}\end{aligned}$$

where τ can be either 0 (in case of a Diffie-Hellman tuple) or random.

At the same time, whereas we cannot compute $\mathbf{b}_2^*$, this does not exclude this second component in vectors computed in $\mathbb{G}_2$: $(\vec{y})_{\mathbb{U}^*} = (\vec{y} \cdot B'^{-1})_{\mathbb{B}^*} = (\vec{y} \cdot B^\top)_{\mathbb{B}^*}$. Thus, for every $\vec{y} \in \mathbb{Z}_q$:

$$(y_1, \ldots, y_n)_{\mathbb{U}^*} = (y_1 + ay_2, y_2, \ldots, y_n)_{\mathbb{B}^*}.$$

Theorem 1 *Under the DDH Assumption in $\mathbb{G}_1$, for random dual orthogonal bases $\mathbb{B}$ and $\mathbb{B}^*$, once having seen $\mathbb{B}$ and $\mathbb{B}^*\backslash\{\mathbf{b}_2^*\}$, and any number of vectors $(y_{i,1}, y_{i,2}, \ldots, y_{i,n})_{\mathbb{B}^*}$, for chosen $\{y_{i,2}, \ldots, y_{i,n}\}_i \in \mathbb{Z}_q^{n-1}$, but unknown random $\{y_{i,1}\}_i \xleftarrow{\$} \mathbb{Z}_q$, one cannot distinguish the vectors $(x_1, x_2', x_3, \ldots, x_n)_{\mathbb{B}}$ and $(x_1, x_2, x_3, \ldots, x_n)_{\mathbb{B}}$, for chosen $x_2, \ldots, x_n \in \mathbb{Z}_q$, but unknown random $x_1, x_2' \xleftarrow{\$} \mathbb{Z}_q$.*

Using the DSDH assumption instead of the DDH assumption, on two chosen values x_2 and x_2', one can show that no algorithm can efficiently distinguish the distribution $(x_1, x_2, x_3, \ldots, x_n)_{\mathbb{B}}$ from $(x_1, x_2', x_3, \ldots, x_n)_{\mathbb{B}}$, for chosen $x_2', x_2, \ldots, x_n \in \mathbb{Z}_q$, but unknown random $x_1 \xleftarrow{\$} \mathbb{Z}_q$:

Theorem 2 (SubSpace-Ind Property) *Under the DSDH Assumption in $\mathbb{G}_1$, for random dual orthogonal bases $\mathbb{B}$ and $\mathbb{B}^*$, once having seen $\mathbb{B}$ and $\mathbb{B}^*\backslash\{\mathbf{b}_2^*\}$, and any number of vectors $(y_{i,1}, y_{i,2}, \ldots, y_{i,n})_{\mathbb{B}^*}$, for chosen $\{y_{i,2}, \ldots, y_{i,n}\}_i \in \mathbb{Z}_q^{n-1}$, but unknown random $\{y_{i,1}\}_i \xleftarrow{\$} \mathbb{Z}_q$, one cannot distinguish the vectors $(x_1, x_2', x_3, \ldots, x_n)_{\mathbb{B}}$ and $(x_1, x_2, x_3, \ldots, x_n)_{\mathbb{B}}$, for any chosen $x_2', x_2, \ldots, x_n \in \mathbb{Z}_q$, but unknown random $x_1 \xleftarrow{\$} \mathbb{Z}_q$.*

We stress that for this property, we only work with $(\mathbf{b}_1, \mathbf{b}_2)$ and $(\mathbf{b}_1^*, \mathbf{b}_2^*)$, but without publishing $\mathbf{b}_2^*$.

4.4.3 Indistinguishability of Position (Pos-Ind)

Let us consider another change of basis:

$$B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ a & -a & 1 \end{pmatrix}_{1,2,3} \qquad B' = \begin{pmatrix} 1 & 0 & -a \\ 0 & 1 & a \\ 0 & 0 & 1 \end{pmatrix}_{1,2,3} \qquad \mathbb{B} = B \cdot \mathbb{U} \qquad \mathbb{B}^* = B' \cdot \mathbb{U}^*$$

In this case, we can compute $\mathbb{B} = (\mathbf{b}_i)_i$, but not the vectors $\mathbf{b}_1^*$ and $\mathbf{b}_2^*$ as $a \cdot G_2$ is missing.

$$(c, -c, b, x_4, \ldots, x_n)_{\mathbb{U}} = (c - ab, -c + ab, b, x_4, \ldots, x_n)_{\mathbb{B}} = (\tau, -\tau, b, x_4, \ldots, x_n)_{\mathbb{B}}$$
$$(\theta, \theta, y_3, y_4, \ldots, y_n)_{\mathbb{U}^*} = (\theta, \theta, a\theta - a\theta + y_3, y_4, \ldots, y_n)_{\mathbb{B}^*} = (\theta, \theta, y_3, \ldots, y_n)_{\mathbb{B}^*}$$

There is the limitation for the first two components in $\mathbb{B}^*$ to be the same:

Theorem 3 (Pos-Ind Property) *Under the DDH Assumption in $\mathbb{G}_1$, for random dual orthogonal bases $\mathbb{B}$ and $\mathbb{B}^*$, once having seen $\mathbb{B}$ and $\mathbb{B}^*\backslash\{\mathbf{b}_1^*, \mathbf{b}_2^*\}$ and any number of vectors $(y_{i,1}, y_{i,1}, \ldots, y_{i,n})_{\mathbb{B}^*}$, for chosen $\{y_{i,1}, y_{i,3}, \ldots, y_{i,n}\}_i \in \mathbb{Z}_q^{n-1}$, one cannot distinguish the vectors $(x_1, -x_1, x_3, x_4, \ldots, x_n)_{\mathbb{B}}$ and $(0, 0, x_3, x_4, \ldots, x_n)_{\mathbb{B}}$, for chosen $x_4, \ldots, x_n \in \mathbb{Z}_q$, but unknown random $x_1, x_3 \xleftarrow{\$} \mathbb{Z}_q$.*

We stress again that for this property, we only work with $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)$ and $(\mathbf{b}_1^*, \mathbf{b}_2^*, \mathbf{b}_3^*)$, but without publishing $(\mathbf{b}_1^*, \mathbf{b}_2^*)$.

But more useful, using the DSDH assumption on 0 and x_1, which claims indistinguishability between $(a \cdot G, b \cdot G, (ab + 0) \cdot G)$ and $(a \cdot G, b \cdot G, (ab + x_1) \cdot G)$, we have indistinguishability

between

$$\begin{aligned}
&(0, x_1, x_3, \ldots, x_n)_{\mathbb{B}} + (ab, -ab, b, 0, \ldots, 0)_{\mathbb{U}} \\
&\qquad = (0, x_1, x_3, \ldots, x_n)_{\mathbb{B}} + (ab - ab, -ab + ab, b, 0, \ldots, 0)_{\mathbb{B}} \\
&\qquad = (0, x_1, x_3, \ldots, x_n)_{\mathbb{B}} \\
&(0, x_1, x_3, \ldots, x_n)_{\mathbb{B}} + (ab + x_1, -ab - x_1, b, 0, \ldots, 0)_{\mathbb{U}} \\
&\qquad = (0, x_1, x_3, \ldots, x_n)_{\mathbb{B}} + (ab + x_1 - ab, -ab - x_1 + ab, b, 0, \ldots, 0)_{\mathbb{B}} \\
&\qquad = (x_1, 0, x_3, \ldots, x_n)_{\mathbb{B}} \\
&(y_1, y_1, y_3, y_4, \ldots, y_n)_{\mathbb{U}^*} = (y_1, y_1, ay_1 - ay_1 + y_3, y_4, \ldots, y_n)_{\mathbb{B}^*} \\
&\qquad = (y_1, y_1, y_3, \ldots, y_n)_{\mathbb{B}^*}
\end{aligned}$$

Hence,

Theorem 4 (Swap-Ind Property) *Under the DSDH Assumption in $\mathbb{G}_1$, for random dual orthogonal bases $\mathbb{B}$ and $\mathbb{B}^*$, once having seen $\mathbb{B}$ and $\mathbb{B}^*\backslash\{\mathbf{b}_1^*, \mathbf{b}_2^*\}$ and any number of vectors $(y_{i,1}, y_{i,1}, \ldots, y_{i,n})_{\mathbb{B}^*}$, for chosen $\{y_{i,1}, y_{i,3}, \ldots, y_{i,n}\}_i \in \mathbb{Z}_q^{n-1}$, one cannot distinguish the vectors $(x_1, 0, x_3, x_4, \ldots, x_n)_{\mathbb{B}}$ and $(0, x_1, x_3, x_4, \ldots, x_n)_{\mathbb{B}}$, for chosen $x_1, x_4, \ldots, x_n \in \mathbb{Z}_q$, but unknown random $x_3 \xleftarrow{\$} \mathbb{Z}_q$.*

Again, for this property, we only work with $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)$ and $(\mathbf{b}_1^*, \mathbf{b}_2^*, \mathbf{b}_3^*)$, but without publishing $(\mathbf{b}_1^*, \mathbf{b}_2^*)$.

4.4.4 Indexing and Randomness Amplification (Index-Ind)

Static version The crucial tool introduced in [OT12b] is the following change of basis, for chosen scalars $t \neq p \in \mathbb{Z}_q$. We start with what we call the static version, where t and p must be known at the beginning of the game.

$$B = \frac{1}{t-p} \times \begin{pmatrix} t & -p & at \\ -1 & 1 & -a \\ 0 & 0 & t-p \end{pmatrix}_{1,2,3} \qquad B' = \begin{pmatrix} 1 & 1 & 0 \\ p & t & 0 \\ -a & 0 & 1 \end{pmatrix}_{1,2,3}$$

In this case, we can compute $\mathbb{B} = (\mathbf{b}_i)_i$, but not the vectors $\mathbf{b}_3^*$ as $a \cdot G_2$ is missing.

$$\begin{aligned}
(b, 0, c, x_4, \ldots, x_n)_{\mathbb{U}} &= (b, bp, c - ab, x_4, \ldots, x_n)_{\mathbb{B}} \\
&= (b \cdot (1, p), \tau, x_4, \ldots, x_n)_{\mathbb{B}} \\
((t-p) \cdot (\pi, 0), \delta, y_4, \ldots, y_n)_{\mathbb{U}^*} &= (\pi t + at\delta/(t-p), -\pi - a\delta/(t-p), \delta, y_4, \ldots, y_n)_{\mathbb{B}^*} \\
&= ((\pi + a\delta/(t-p)) \cdot (t, -1), \delta, y_4, \ldots, y_n)_{\mathbb{B}^*}
\end{aligned}$$

There is the limitation for the first two components in $\mathbb{B}$ and $\mathbb{B}^*$ not to be orthogonal: $\langle (1, p), (t, -1) \rangle = (t - p) \neq 0$:

Theorem 5 *Under the DDH Assumption in $\mathbb{G}_1$, for random dual orthogonal bases $\mathbb{B}$ and $\mathbb{B}^*$, once having seen $\mathbb{B}$ and $\mathbb{B}^*\backslash\{\mathbf{b}_3^*\}$, and any number of vectors $(\pi_i \cdot (t, -1), y_{i,3}, \ldots, y_{i,n})_{\mathbb{B}^*}$, for chosen $\{y_{i,3}, \ldots, y_{i,n}\} \in \mathbb{Z}_q^{n-2}$, but unknown random $\{\pi_i\}_i \xleftarrow{\$} \mathbb{Z}_q$, and for any chosen $t \neq p \in \mathbb{Z}_q$, one cannot distinguish the vectors $(b \cdot (1, p), \tau, x_4, \ldots, x_n)_{\mathbb{B}}$ and $(b \cdot (1, p), 0, x_4, \ldots, x_n)_{\mathbb{B}}$, for chosen $x_4, \ldots, x_n \in \mathbb{Z}_q$, but unknown random $b, \tau \xleftarrow{\$} \mathbb{Z}_q$.*

As above, we can have a more convenient theorem under the DSDH assumption:

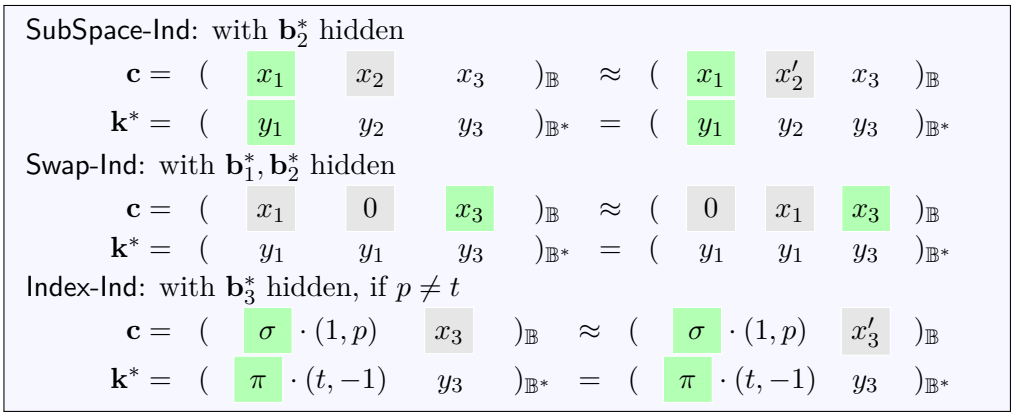

Figure 4.1: Computationally indistinguishable Changes of Basis

Theorem 6 ((Static) Index-Ind Property) *Under the DSDH Assumption in $\mathbb{G}_1$, for random dual orthogonal bases $\mathbb{B}$ and $\mathbb{B}^*$, once having seen $\mathbb{B}$ and $\mathbb{B}^*\backslash\{\mathbf{b}_3^*\}$, and any number of vectors $(\pi_i \cdot (t,-1), y_{i,3}, \ldots, y_{i,n})_{\mathbb{B}^*}$, for chosen $\{y_{i,3}, \ldots, y_{i,n}\} \in \mathbb{Z}_q^{n-2}$, but unknown random $\{\pi_i\}_i \xleftarrow{\$} \mathbb{Z}_q$, and for any chosen $t \neq p \in \mathbb{Z}_q$, one cannot distinguish the vectors $(\sigma \cdot (1,p), x_3, x_4, \ldots, x_n)_{\mathbb{B}}$ and $(\sigma \cdot (1,p), x_3', x_4, \ldots, x_n)_{\mathbb{B}}$, for chosen $x_3', x_3, x_4, \ldots, x_n \in \mathbb{Z}_q$, but unknown random $\sigma \xleftarrow{\$} \mathbb{Z}_q$.*

For this property, we only work with $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)$ and $(\mathbf{b}_1^*, \mathbf{b}_2^*, \mathbf{b}_3^*)$, but without publishing $\mathbf{b}_3^*$. For a fixed t, we can iteratively update all the other indices $p \neq t$.

We stress that, in this static version, t and p must be fixed, and known before the simulation starts in the security analysis, as they will appear in the matrix B. In the Okamoto-Takashima's constructions [OT10, OT12b], such values t and p were for bounded names of attributes.

Adaptive version In the following, we want to consider unbounded attributes, we thus conclude this section with an adaptive version, where t and p do not need to be known in advance, from a large universe. We will also take the opportunity to illustrate how proofs are conducted in the DPVS, as the proof for Adaptive Index-Ind is done using the Indistinguishable properties presented earlier in the section. A high level overview of the proof is presented on Figure 4.2.

Theorem 7 (Adaptive Index-Ind Property) *Under the DDH Assumption in $\mathbb{G}_1$, for random dual orthogonal bases $\mathbb{B}$ and $\mathbb{B}^*$, once having seen $\mathbb{B}$ and $\mathbb{B}^*\backslash\{\mathbf{b}_3^*\}$, and any number of vectors $(\pi_i \cdot (t,-1), y_{i,3}, 0, 0, y_{i,6}, \ldots, y_{i,n})_{\mathbb{B}^*}$, for any $t \in \mathbb{Z}_q, \{y_{i,3}, y_{i,6}, \ldots, y_{i,n}\}_i \in \mathbb{Z}_q^{n-4}$, but unknown random $\{\pi_i\}_i \xleftarrow{\$} \mathbb{Z}_q$, one cannot distinguish between $(\sigma \cdot (1,p), x_3, 0, 0, x_6, \ldots, x_n)_{\mathbb{B}}$ and $(\sigma \cdot (1,p), x_3', 0, 0, x_6, \ldots, x_n)_{\mathbb{B}}$, for any $x_3, x_3', x_6, \ldots, x_n \in \mathbb{Z}_q$, and $p \neq t$, but unknown random $\sigma \xleftarrow{\$} \mathbb{Z}_q$, with an advantage better than $8 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(T) + 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(T)$, where T is the running time of the adversary.*

Proof For the sake of simplicity, we will prove indistinguishability between $(\sigma \cdot (1,p), 0, 0, 0)_{\mathbb{B}}$ and $(\sigma \cdot (1,p), x_3, 0, 0)_{\mathbb{B}}$, in dimension 5 only, instead of n. Additional components could be chosen by the adversary. Applied twice, we obtain the above theorem. The proof follows a sequence of games.

Game $\mathbf{G}_0$: The adversary can choose $p \neq t$ and x_3, y_3 in $\mathbb{Z}_q$, but $\pi, \sigma \xleftarrow{\$} \mathbb{Z}_q$ are unknown to it:

$$\mathbf{k}^* = (\pi(t,-1), y_3, 0, 0)_{\mathbb{B}^*} \qquad \mathbf{c}_0 = (\sigma(1,p), 0, 0, 0)_{\mathbb{B}}$$
$$\mathbf{c}_1 = (\sigma(1,p), x_3, 0, 0)_{\mathbb{B}}$$

$\mathbf{G}_0$ Initial situation before indexing

$$\begin{array}{lllllll} \mathbf{c}_0 & = (& \pi(t, & -1) & \beta & 0 & 0 &)_{\mathbb{B}} \\ \mathbf{k}^* & = (& \sigma(1, & p) & 0 & 0 & 0 &)_{\mathbb{B}^*} \end{array}$$

$\mathbf{G}_1$ SubSpace-Ind on (1,2,5,6)

$$\begin{array}{lllllll} \mathbf{c}_0 & = (& \pi(t, & -1) & \beta & \rho(t, & -1) &)_{\mathbb{B}} \\ \mathbf{k}^* & = (& \sigma(1, & p) & 0 & 0 & 0 &)_{\mathbb{B}^*} \end{array}$$

$\mathbf{G}_2$ SubSpace-Ind on (1,2,3,5,6)

$$\begin{array}{lllllll} \mathbf{c}_0 & = (& \pi(t, & -1) & \beta & \rho(t, & -1) &)_{\mathbb{B}} \\ \mathbf{k}^* & = (& \sigma(1, & p) & 0 & \sigma(1, & p) &)_{\mathbb{B}^*} \end{array}$$

$\mathbf{G}_3$ Formal change on (5,6)

$$\begin{array}{lllllll} \mathbf{c}_0 & = (& \pi(t, & -1) & \beta & u_1 & u_2 &)_{\mathbb{B}} \\ \mathbf{k}^* & = (& \sigma(1, & p) & 0 & v_1 & v_2 &)_{\mathbb{B}^*} \end{array}$$

$\mathbf{G}_4$ SubSpace-Ind on (5,4)

$$\begin{array}{lllllll} \mathbf{c}_0 & = (& \pi(t, & -1) & \beta & u_1 & u_2 &)_{\mathbb{B}} \\ \mathbf{k}^* & = (& \sigma(1, & p) & \alpha & v_1 & v_2 &)_{\mathbb{B}^*} \end{array}$$

$\mathbf{G}_5$ Formal change on (5,6)

$$\begin{array}{lllllll} \mathbf{c}_0 & = (& \pi(t, & -1) & \beta & \rho(t, & -1) &)_{\mathbb{B}} \\ \mathbf{k}^* & = (& \sigma(1, & p) & \alpha & \sigma(1, & p) &)_{\mathbb{B}^*} \end{array}$$

$\mathbf{G}_6$ SubSpace-Ind on (1,2,3,5,6)

$$\begin{array}{lllllll} \mathbf{c}_0 & = (& \pi(t, & -1) & \beta & \rho(t, & -1) &)_{\mathbb{B}} \\ \mathbf{k}^* & = (& \sigma(1, & p) & \alpha & 0 & 0 &)_{\mathbb{B}^*} \end{array}$$

$\mathbf{G}_7$ SubSpace-Ind on (1,2,5,6)

$$\begin{array}{lllllll} \mathbf{c}_0 & = (& \pi(t, & -1) & \beta & 0 & 0 &)_{\mathbb{B}} \\ \mathbf{k}^* & = (& \sigma(1, & p) & \alpha & 0 & 0 &)_{\mathbb{B}^*} \end{array}$$

Figure 4.2: Sequence of Games for Index-Ind Property

Vectors $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_1^*, \mathbf{b}_2^*)$ and $(\mathbf{c}_b, \mathbf{k}^*)$ are provided to the adversary that must decide on b: Adv_0 is its advantage in correctly guessing b. Only $\mathbf{k}^*$ and $\mathbf{c}_0$ will be modified in the following games, so that eventually $\mathbf{c}_0 = \mathbf{c}_1$ in the last game, which leads to perfect indistinguishability.

Game $\mathbf{G}_1$: We replicate the first sub-vector $(t, -1)$, with $\rho \xleftarrow{\$} \mathbb{Z}_q$, in the hidden components: $\mathbf{k}^* = (\pi(t,-1), y_3, \rho(t,-1))_{\mathbb{B}^*}$. To show the indistinguishability, one applies the SubSpace-Ind property on $(\mathbb{B}^*, \mathbb{B})_{1,2,4,5}$. Indeed, we can consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \tau \bmod q$ with either $\tau = 0$ or random, which are indistinguishable under the DDH assumption in $\mathbb{G}_2$. Let us assume we start from random dual orthogonal bases $(\mathbb{V}, \mathbb{V}^*)$. We define

$$B' = \begin{pmatrix} 1 & 0 & a & 0 \\ 0 & 1 & 0 & a \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}_{1,2,4,5} \qquad B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -a & 0 & 1 & 0 \\ 0 & -a & 0 & 1 \end{pmatrix}_{1,2,4,5} \qquad \mathbb{B}^* = B' \cdot \mathbb{V}^* \quad \mathbb{B} = B \cdot \mathbb{V}$$

The vectors $\mathbf{b}_4, \mathbf{b}_5$ can not be computed, but they are hidden from the adversary's view, and are not used in any vector. We compute the new vectors:

$$\begin{aligned} \mathbf{k}^* &= (b(t,-1), y_3, c(t,-1))_{\mathbb{V}^*} \\ &= (b(t,-1), y_3, (c-ab)(t,-1))_{\mathbb{B}^*} \\ &= (b(t,-1), y_3, \tau(t,-1))_{\mathbb{B}^*} \end{aligned} \qquad \mathbf{c}_0 = (\sigma(1,p), 0, 0, 0)_{\mathbb{B}}$$

One can note that when $\tau = 0$, this is the previous game, and when τ random, we are in the new game, with $\pi = b$ and $\rho = \tau$: $\mathsf{Adv}_0 - \mathsf{Adv}_1 \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(T)$.

Game $\mathbf{G}_2$: We replicate the non-orthogonal sub-vector $(1, p)$, with $\theta \xleftarrow{\$} \mathbb{Z}_q$:

$$\mathbf{k}^* = (\pi(t,-1), y_3, \rho(t,-1))_{\mathbb{B}^*} \qquad \mathbf{c}_0 = (\sigma(1,p), 0, \theta(1,p))_{\mathbb{B}}$$

To show the indistinguishability, one applies the SubSpace-Ind property on $(\mathbb{B}, \mathbb{B}^*)_{1,2,4,5}$. Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \tau \bmod q$ with either $\tau = 0$ or random, which are indistinguishable under the DDH assumption in $\mathbb{G}_1$. Let us assume we start from random dual orthogonal bases $(\mathbb{V}, \mathbb{V}^*)$. Then we define the matrices

$$B = \begin{pmatrix} 1 & 0 & a & 0 \\ 0 & 1 & 0 & a \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}_{1,2,4,5} \qquad B' = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -a & 0 & 1 & 0 \\ 0 & -a & 0 & 1 \end{pmatrix}_{1,2,4,5} \qquad \mathbb{B} = B \cdot \mathbb{V} \quad \mathbb{B}^* = B' \cdot \mathbb{V}^*$$

The vectors $\mathbf{b}_4^*, \mathbf{b}_5^*$ can not be computed, but they are hidden from the adversary's view. We compute the new vectors in $\mathbb{V}$ and $\mathbb{V}^*$:

$$\begin{aligned} \mathbf{c}_0 &= (b(1,p), 0, c(1,p))_{\mathbb{V}} \\ &= (b(1,p), 0, (c-ab)(1,p))_{\mathbb{B}} \\ &= (b(1,p), 0, \tau(1,p))_{\mathbb{B}} \end{aligned} \qquad \begin{aligned} \mathbf{k}^* &= (\pi'(t,-1), y_3, \rho(t,-1))_{\mathbb{V}^*} \\ &= ((\pi' + a\rho)(t,-1), y_3, \rho(t,-1))_{\mathbb{B}^*} \end{aligned}$$

One can note that when $\tau = 0$, this is the previous game, and when τ random, we are in the new game, with $\pi = \pi' + a\rho$, $\sigma = b$, and $\theta = \tau$: $\mathsf{Adv}_1 - \mathsf{Adv}_2 \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(T)$.

Game $\mathbf{G}_3$: We randomize the two non-orthogonal sub-vectors, with random scalars $u_1, u_2, v_1, v_2 \xleftarrow{\$} \mathbb{Z}_p$:

$$\mathbf{k}^* = (\pi(t,-1), y_3, u_1, u_2)_{\mathbb{B}^*} \qquad \mathbf{c}_0 = (\sigma(1,p), 0, v_1, v_2)_{\mathbb{B}}$$

To show the indistinguishability, one makes a formal change of basis on $(\mathbb{B}^*, \mathbb{B})_{4,5}$, with a random unitary matrix Z, with $z_1 z_4 - z_2 z_3 = 1$:

$$B' = Z = \begin{pmatrix} z_1 & z_2 \\ z_3 & z_4 \end{pmatrix}_{4,5} \qquad B = \begin{pmatrix} z_4 & -z_3 \\ -z_2 & z_1 \end{pmatrix}_{4,5} \qquad \mathbb{B}^* = B' \cdot \mathbb{V}^* \qquad \mathbb{B} = B \cdot \mathbb{V}$$

This only impacts the hidden vectors $(\mathbf{b}_4, \mathbf{b}_5)$, $(\mathbf{b}_4^*, \mathbf{b}_5^*)$. If one defines $\mathbf{k}^*$ and $\mathbf{c}_0$ in $(\mathbb{V}^*, \mathbb{V})$, this translates in $(\mathbb{B}^*, \mathbb{B})$:

$$\mathbf{k}^* = (\pi(t,-1), y_3, \rho(t,-1))_{\mathbb{V}^*} = (\pi(t,-1), y_3, \rho(tz_1 - z_3, tz_2 - z_4))_{\mathbb{B}^*}$$
$$\mathbf{c}_0 = (\sigma(1,p), 0, \theta(1,p))_{\mathbb{V}} = (\sigma(1,p), 0, \theta(z_4 - pz_2, -z_3 + pz_1))_{\mathbb{B}}$$

Let us consider random $u_1, u_2, v_1, v_2 \xleftarrow{\$} \mathbb{Z}_p$, and solve the system in z_1, z_2, z_3, z_4. This system admits a unique solution, if and only if $t \neq p$. And with random ρ, θ, and random unitary matrix Z,

$$\mathbf{k}^* = (\pi(t,-1), y_3, u_1, u_2)_{\mathbb{B}^*} \qquad \mathbf{c}_0 = (\sigma(1,p), 0, v_1, v_2)_{\mathbb{B}}$$

with random scalars $u_1, u_2, v_1, v_2 \xleftarrow{\$} \mathbb{Z}_p$. In bases $(\mathbb{V}, \mathbb{V}^*)$, we are in the previous game, and in bases $(\mathbb{B}, \mathbb{B}^*)$, we are in the new game, if $p \neq t$: $\mathsf{Adv}_2 = \mathsf{Adv}_3$.

Game $\mathbf{G}_4$: We now randomize the third component in $\mathbf{c}_0$:

$$\mathbf{k}^* = (\pi(t,-1), y_3, u_1, u_2)_{\mathbb{B}^*} \qquad \mathbf{c}_0 = (\sigma(1,p), x_3, v_1, v_2)_{\mathbb{B}}$$

To show the indistinguishability, one applies the SubSpace-Ind property on $(\mathbb{B}, \mathbb{B}^*)_{4,3}$. Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \tau \bmod q$ with either $\tau = 0$ or $\tau = x_3$, which are indistinguishable under the DDH assumption in $\mathbb{G}_1$. Let us assume we start from random dual orthogonal bases $(\mathbb{V}, \mathbb{V}^*)$. Then we define the matrices

$$B = \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix}_{3,4} \qquad B' = \begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix}_{3,4} \qquad \mathbb{B} = B \cdot \mathbb{V} \qquad \mathbb{B}^* = B' \cdot \mathbb{V}^*$$

The vectors $\mathbf{b}_3^*$ can not be computed, but it is not into the adversary's view. We compute the new vectors:

$$\begin{aligned} \mathbf{k}^* &= (\pi(t,-1), y_3, u_1', u_2)_{\mathbb{V}^*} \\ &= (\pi(t,-1), y_3, u_1' + ay_3, u_2)_{\mathbb{B}^*} \end{aligned} \qquad \begin{aligned} \mathbf{c}_0 &= (\sigma(1,p), c, b, v_2)_{\mathbb{V}} \\ &= (\sigma(1,p), c - ab, b, v_2)_{\mathbb{B}} \\ &= (\sigma(1,p), \tau, b, v_2)_{\mathbb{B}} \end{aligned}$$

One can note that when $\tau = 0$, this is the previous game, and when $\tau = x_3$, we are in the new game, with $v_1 = b$ and $u_1 = u_1' + ay_3$: $\mathsf{Adv}_3 - \mathsf{Adv}_4 \leq 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(T)$, by applying twice the Diffie-Hellman indistinguishability game.

We can undo successively games $\mathbf{G}_3$, $\mathbf{G}_2$, and $\mathbf{G}_1$ to get, after a gap bounded by $\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$: $\mathbf{k}^* = (\pi(t,-1), y_3, 0, 0)_{\mathbb{B}^*}$ and $\mathbf{c}_0 = (\sigma(1,p), x_3, 0, 0)_{\mathbb{B}}$. In this game, the advantage of any adversary is 0. The global difference of advantages is bounded by $4 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(T) + 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(T)$, which concludes the proof.

Chapter

5 Basic ABE and ABS Constructions

Chapter content

In this section, we present two constructions from Okamoto and Takashima based on the DPVS. While OT constructions were originally made under the DLIN assumptions, we adapt them under the SXDH assumption.

The first construction is an ABE scheme, with unbounded attributes (the setup allows for an unlimited number of attributes during encryption and decryption) which is in the same vein as [OT12b].

The second construction is an ABS scheme with non-monotone predicates (a signature can use NOT gates in the policies), which is then extended to the multi-authority setting [OT11]. While both of these features are interesting, we do not keep them in order to focus on delegation for our contributions in a later section.

While both these constructions are not strictly original, they illustrate that our new proof formalism for DPVS is fully compatible with these classic constructions, while relying only on the three theorems proved previously in Section 4.4.

But before that, we explain an important proof paradigm for Attribute-based constructions, the Dual System Encryption.

5.1 Overview of the Dual System Encryption (DSE)

The DSE paradigm was invented by Waters in [Wat09] to solve the problem of constructing Identity-Based constructions with adaptive security. At the time, the known method of proof for selective security in IBE and ABE was the *partitioning* technique, where the simulator would hide the challenge in the target set specified by the adversary at the very beginning of the game. This method can be used for adaptive security in IBE [Wat05], as the simulator can try and guess a single identity during the initialization that will be the adversary's choice for the challenge. But for ABE this approach is impossible, as there are conflicts for even a single

attribute depending on wether they are in an access-tree queried by the adversary, or in the challenge set of the adversary, or used in multiple different keys. More details are provided in [LOS+10].

The main idea of the DSE is as follows: Suppose that, only during the proof, the authority was able to make *semi-functional* keys and ciphertexts, which are completely undistinguishable from normal keys and ciphertexts. These semi-functional elements behave exactly the same as their normal counterparts, with a single exception: no semi-functional key can decipher a semi-functional ciphertext. Then, during the proof, the only thing the simulator needs to do is to change all the keys issued to the adversary into semi-functional ones, and make the challenge answer to the adversary a semi-functional ciphertext. This way, the adversary can extract no information from the challenge ciphertext because all his keys are semi-functional, and thus he has no advantage to answer the final challenge. We note that regular ciphertexts created by the adversary during the game are unaffected by this strategy, as only the authority can create semi-functional ciphertexts. This implies that the adversary does not even know that semi-functional keys are handed to him during the game, as these keys will decipher any ciphertexts he might create himself.

In the DPVS setting, semi-functionality is achieved by increasing the dimension of the vector space in order to obtain distinct subspaces: the normal one, and the semi-functional one. Only the normal subspace is used concretely during the construction to manipulate the usual elements of the scheme, typically the parts of the secret sharing and the random of the ciphertext. The other "unused" semi-functional subspace is always zero for keys and ciphertexts, except during the proof. At that moment, the authority will use this subspace to put some kind of noise in the semi-functional keys and challenge ciphertext. As the normal keys and challenge ciphertext have only zero elements in this subspace, the pairing between a normal and a semi-functional element will naturally suppress the noise added in the semi-functional subspace (since the pairing is done subspace by subspace). However, when two semi-functional elements are paired together, the noise doesn't cancel out: decryption is impossible. This is the reason why in our constructions, the bases for the vector spaces is approximately twice the size necessary for the scheme, the first half of vectors is for the normal subspace, and the second half is for the semi-functional subspace.

5.2 A Key-Policy Attribute-Based Encryption (KP-ABE) Construction

Our goal for KP-ABE is to design of a new scheme with Switchable Attributes. Before that, we start with a construction that has all the important properties for an ABE. We choose the construction from Okamoto-Takashima that provides some kind of attribute-hiding property, where no one can tell which attributes has been used in the policy of the encrypted ciphertext. On top of that, except for the adaptive security, this construction is in the same vein as [GPSW06] regarding the expressiveness of the policy, or the delegation capabilities, which is compatible with our final objective. We note that OT's original construction doesn't allow for delegation. For the sake of clarity, just using the static Index-Ind theorem, this construction can only handle a polynomially-bounded universe of attributes and delegation, but with adaptive-set security (see Definition 5).

5.2.1 Description of the KP-ABE Scheme

For the construction, we will use two DPVS, of dimensions 3 and 6 respectively, in a pairing-friendly setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$, using the notations introduced in Section 4:

$\mathsf{Setup}(1^\kappa)$**.** The algorithm chooses two random dual orthogonal bases

$$\mathbb{B} = (\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3) \qquad \mathbb{B}^* = (\mathbf{b}_1^*, \mathbf{b}_2^*, \mathbf{b}_3^*)$$
$$\mathbb{D} = (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6) \qquad \mathbb{D}^* = (\mathbf{d}_1^*, \mathbf{d}_2^*, \mathbf{d}_3^*, \mathbf{d}_4^*, \mathbf{d}_5^*, \mathbf{d}_6^*).$$

It sets the public parameters $\mathsf{PK} = \{(\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}_1^*), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_1^*, \mathbf{d}_2^*, \mathbf{d}_3^*)\}$, whereas the master secret key $\mathsf{MK} = \{\mathbf{b}_3^*\}$. Other basis vectors are kept hidden.

$\mathsf{KeyGen}(\mathsf{MK}, \mathcal{T})$**.** For an access-tree $\mathcal{T}$, the algorithm first chooses a random $a_0 \xleftarrow{\$} \mathbb{Z}_q$, and a random a_0-labeling $(a_\lambda)_\lambda$ of the access-tree $\mathcal{T}$, and builds the key:

$$\mathbf{k}_0^* = (a_0, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}_\lambda^* = (\pi_\lambda(1, t_\lambda), a_\lambda, 0, 0, 0)_{\mathbb{D}^*}$$

for all the leaves λ, where $t_\lambda = A(\lambda)$ and $\pi_\lambda \xleftarrow{\$} \mathbb{Z}_q$. The decryption key $\mathsf{dk}_\mathcal{T}$ is then $(\mathbf{k}_0^*, (\mathbf{k}_\lambda^*)_\lambda)$.

$\mathsf{Delegate}(\mathsf{dk}_\mathcal{T}, \mathcal{T}')$**.** The algorithm first generates zero-label credentials for the new attributes, with $\mathbf{k}_\lambda^* \leftarrow (\pi_\lambda \cdot (1, t_\lambda), 0, 0, 0, 0)_{\mathbb{D}^*}$, with $\pi_\lambda \xleftarrow{\$} \mathbb{Z}_q$, for a new leaf. Keeping only the credentials useful in $\mathcal{T}'$, it gets a valid key from $\mathsf{dk}_\mathcal{T}$. It can thereafter be randomized with a random $a_0' \xleftarrow{\$} \mathbb{Z}_q$ and a random a_0'-labeling (a_λ') of $\mathcal{T}'$, with $\mathbf{k}_0^* \leftarrow \mathbf{k}_0^* + (a_0', 0, 0)_{\mathbb{B}^*}$, and $\mathbf{k}_\lambda^* \leftarrow \mathbf{k}_\lambda^* + (\pi_\lambda' \cdot (1, t_\lambda), a_\lambda', 0, 0, 0)_{\mathbb{D}^*}$, for $\pi_\lambda' \xleftarrow{\$} \mathbb{Z}_q$.

$\mathsf{Encaps}(\mathsf{PK}, \Gamma)$**.** For the set Γ of attributes, the algorithm first chooses random scalars $\omega, \xi \xleftarrow{\$} \mathbb{Z}_q$. It then sets $K = g_t^\xi$ and generates the ciphertext $C = (\mathbf{c}_0, (\mathbf{c}_t)_{t\in\Gamma})$ where $\mathbf{c}_0 = (\omega, 0, \xi)_\mathbb{B}$ and $\mathbf{c}_t = (\sigma_t(t, -1), \omega, 0, 0, 0)_\mathbb{D}$, for all the attributes $t \in \Gamma$ and $\sigma_t \xleftarrow{\$} \mathbb{Z}_q$.

$\mathsf{Decaps}(\mathsf{dk}_\mathcal{T}, C)$**.** The algorithm first selects an evaluation pruned tree $\mathcal{T}'$ of $\mathcal{T}$ that is satisfied by Γ. This means that the labels a_λ for all the leaves λ in $\mathcal{T}'$ allow to reconstruct a_0 by simple additions.

Note that from every leaf λ in $\mathcal{T}'$ and $t = t_\lambda = A(\lambda) \in \Gamma$, it can compute

$$\mathbf{c}_t \times \mathbf{k}_t^* = g_t^{\sigma_t \cdot \pi_\lambda \cdot \langle (t,-1),(1,t)\rangle + \omega \cdot a_\lambda} = g_t^{\omega \cdot a_\lambda}.$$

Hence, it can derive $g_t^{\omega \cdot a_0}$. From $\mathbf{c}_0$ and $\mathbf{k}_0^*$, it gets $\mathbf{c}_0 \times \mathbf{k}_0^* = g_t^{\omega \cdot a_0 + \xi}$ which then easily leads to $K = g_t^\xi$.

Correctness We stress that in the above decryption, one can recover $g_t^{\omega \cdot a_0}$ if and only if there is an evaluation pruned tree $\mathcal{T}'$ of $\mathcal{T}$ that is satisfied by Γ. And this holds if and only if $\mathcal{T}(\Gamma) = 1$.

Additionally, since $\mathbf{b}_3^*$ is not public but in MK only, for the key issuer, only the latter can issue keys, but anybody can delegate a key for a tree $\mathcal{T}$ into a key for a more restrictive tree $\mathcal{T}'$. As everything can be randomized (the random coins π_λ and the labeling), the delegated keys are perfectly indistinguishable from fresh keys. Hence, given two keys possibly delegated from a common key, one cannot decide whether they have been independently generated or delegated.

5.2.2 Security Analysis of the KP-ABE

We first consider the security analysis, without delegation, as it is quite similar to [OT12b], but under the SXDH assumption instead of the DLIN assumption:

Theorem 8 *Under the SXDH assumption, no adversary can win the IND security game (without delegation) against our KP-ABE scheme, in the Adaptive-Set setting, with non-negligible advantage.*

$\mathbf{G}_0$ Real IND-Security game (without delegation)

$$\begin{array}{llllll} \mathbf{c}_0 = & (\ \omega & 0 & \xi &) & \mathbf{c}_t = \ (\ \sigma_t(1,t) \quad \omega \mid 0 \quad 0 \quad 0 \ ) \\ \mathbf{k}^*_{\ell,0} = & (\ a_{\ell,0} & 0 & 1 &) & \mathbf{k}^*_{\ell,\lambda} = \ (\ \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad 0 \ ) \end{array}$$

$\mathbf{G}_1$ SubSpace-Ind Property, on $(\mathbb{B},\mathbb{B}^*)_{1,2}$ and $(\mathbb{D},\mathbb{D}^*)_{3,4}$, between 0 and $\tau \xleftarrow{\$} \mathbb{Z}_q$

$$\begin{array}{llllll} \mathbf{c}_0 = & (\ \omega & \tau & \xi &) & \mathbf{c}_t = \ (\ \sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad 0 \ ) \\ \mathbf{k}^*_{\ell,0} = & (\ a_{\ell,0} & 0 & 1 &) & \mathbf{k}^*_{\ell,\lambda} = \ (\ \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad 0 \ ) \end{array}$$

$\mathbf{G}_2$ SubSpace-Ind Property, on $(\mathbb{D},\mathbb{D}^*)_{1,2,6}$, between 0 and τz_t

$$\begin{array}{llllll} \mathbf{c}_0 = & (\ \omega & \tau & \xi &) & \mathbf{c}_t = \ (\ \sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad \tau z_t \ ) \\ \mathbf{k}^*_{\ell,0} = & (\ a_{\ell,0} & 0 & 1 &) & \mathbf{k}^*_{\ell,\lambda} = \ (\ \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad 0 \ ) \end{array}$$

$\mathbf{G}_3$ Introduction of an additional random-labeling. See Figure 5.2

$$\begin{array}{llllll} \mathbf{c}_0 = & (\ \omega & \tau & \xi &) & \mathbf{c}_t = \ (\ \sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad \tau z_t \ ) \\ \mathbf{k}^*_{\ell,0} = & (\ a_{\ell,0} & r_{\ell,0} & 1 &) & \mathbf{k}^*_{\ell,\lambda} = \ (\ \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad s_{\ell,\lambda}/z_{t_{\ell,\lambda}} \ ) \end{array}$$

$\mathbf{G}_4$ Formal basis change, on $(\mathbb{B},\mathbb{B}^*)_{2,3}$, to randomize ξ

$$\begin{array}{llllll} \mathbf{c}_0 = & (\ \omega & \tau & \xi'' &) & \mathbf{c}_t = \ (\ \sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad \tau z_t \ ) \\ \mathbf{k}^*_{\ell,0} = & (\ a_{\ell,0} & r_{\ell,0} & 1 &) & \mathbf{k}^*_{\ell,\lambda} = \ (\ \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad s_{\ell,\lambda}/z_{t_{\ell,\lambda}} \ ) \end{array}$$

Gray cells x mean they have been changed in this game.

Figure 5.1: Global sequence of games for the IND-security proof of the KP-ABE

$\mathbf{G}_{2.k.0}$ Hybrid game for $\mathbf{G}_2$, with $1 \le k \le K+1$ (from Figure 5.1)

$$\begin{array}{lllllll} & \mathbf{c}_0 = & (\ \omega & \tau & \xi &) & \mathbf{c}_t = \ (\ \sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad \tau z_t \ ) \\ \ell < k & \mathbf{k}^*_{\ell,0} = & (\ a_{\ell,0} & r_{\ell,0} & 1 &) & \mathbf{k}^*_{\ell,\lambda} = \ (\ \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad s_{\ell,\lambda}/z_{t_{\ell,\lambda}} \ ) \\ \ell \ge k & \mathbf{k}^*_{\ell,0} = & (\ a_{\ell,0} & 0 & 1 &) & \mathbf{k}^*_{\ell,\lambda} = \ (\ \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad 0 \ ) \end{array}$$

$\mathbf{G}_{2.k.1}$ SubSpace-Ind Property, on $(\mathbb{B}^*,\mathbb{B})_{1,2}$ and $(\mathbb{D}^*,\mathbb{D})_{3,4}$, between 0 and $s_{k,*}$

$$\mathbf{k}^*_{k,0} = (\ a_{k,0} \quad s_{k,0} \quad 1 \ ) \quad \mathbf{k}^*_{k,\lambda} = \ (\ \pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \mid s_{k,\lambda} \quad 0 \quad 0 \ )$$

$\mathbf{G}_{2.k.2}$ Masking of the labeling. See Figure 5.3

$$\mathbf{k}^*_{k,0} = (\ a_{k,0} \quad s_{k,0} \quad 1 \ ) \quad \mathbf{k}^*_{k,\lambda} = \ (\ \pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \mid 0 \quad 0 \quad s_{k,\lambda}/z_{t_{k,\lambda}} \ )$$

$\mathbf{G}_{2.k.3}$ Limitations on KeyGen-queries: $s_{k,0}$ unpredictable, replaced by a random $r_{k,0}$

$$\mathbf{k}^*_{k,0} = (\ a_{k,0} \quad r_{k,0} \quad 1 \ ) \quad \mathbf{k}^*_{k,\lambda} = \ (\ \pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \mid 0 \quad 0 \quad s_{k,\lambda}/z_{t_{k,\lambda}} \ )$$

Figure 5.2: Sequence of games on the K keys for the IND-security proof of the KP-ABE

This theorem is proven with exact bound for an adversary with running time bounded by t, where P is the size of the universe of the attributes and K is the number of queries to the OKeyGen-oracle:

$$\begin{aligned} \mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}) &\le 2(KP^2+1) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (3P+1)K \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) \\ &\le (2KP^2+3KP+K+2) \times \mathsf{Adv}^{\mathsf{sxdh}}(t) \end{aligned}$$

The global sequence of games is described on Figure 5.1, with another sequence of sub-games on Figure 5.2.

In the two first games $\mathbf{G}_1$ and $\mathbf{G}_2$, one is preparing random τ and random masks z_t in the ciphertexts $\mathbf{c}_t$ (actually, the challenge ciphertext corresponding to the attribute t), making them *semi-functional*. Note that until the challenge query is asked, one does not exactly know the attributes in the challenge set Γ (as we are in the adaptive-set setting), but we prepare all the material for all possible $\mathbf{c}_t$, and only the ones corresponding to attributes in Γ will be provided to the adversary at challenge time. The main step is to get to Game $\mathbf{G}_3$, with an additional

labeling $(s_{\ell,0}, (s_{\ell,\lambda})_\lambda)$ put inside the keys to make them *semi-functional*, using hybrid games starting from Game $\mathbf{G}_2$. The sequence on Figure 5.2 gives more details: the new labelling is added in each ℓ-th key (in $\mathbf{G}_{2.k.1}$), then each label is masked by the random z_t for each attribute t (in $\mathbf{G}_{2.k.2}$). In order to go to game $\mathbf{G}_{2.k.3}$ one exploits the limitations on the adversary in the security game: the adversary cannot ask keys on access-trees $\mathcal{T}$ such that $\mathcal{T}(\Gamma) = 1$, for the challenge set Γ.

This construction makes more basis vectors public than in the original proof from [OT12b], and only $\mathbf{b}_3^*$ is for the key issuer. This is the reason why we can deal with delegation for any user, as the delegator has access to all the informations to create a new key from his own. In addition, as delegation provides keys that are perfectly indistinguishable from fresh keys, one can easily get the full result:

Corollary 1 *Under the SXDH assumption, no adversary can win the Del-IND security game against the KP-ABE scheme, in the Adaptive-Set setting, with non-negligible advantage.*

The bound is the same, except K is the global number of OKeyGen and ODelegate queries.

In this section, we will focus on the IND-security proof of the KP-ABE scheme.

Security proofs The global sequence of games will follow the steps shown on Figure 5.1. But while the first steps (from $\mathbf{G}_0$ to $\mathbf{G}_2$) will be simple, the big step from $\mathbf{G}_2$ to $\mathbf{G}_3$ will need multiple hybrid games, presented on Figure 5.2. All theses games work in a pairing-friendly setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$, with two random dual orthogonal bases $(\mathbb{B}, \mathbb{B}^*)$ and $(\mathbb{D}, \mathbb{D}^*)$ of size 3 and 6, respectively.

In the following proof, we will use t to denote attributes, and thus the indices for the possible ciphertexts $\mathbf{c}_t$ associated to each attribute in the challenge ciphertext. We indeed anticipate all the possible $\mathbf{c}_t$, before knowing the exact set Γ, as we are in the adaptive setting. The variable p will be used in hybrid proofs to specify a particular attribute. We will denote P the size of the universe of attributes. Then $1 \leq t, p \leq P$. Similarly, we will use ℓ to denote key queries, and thus the index of the global ℓ-th key $\mathbf{k}_\ell^*$, whereas λ will we used for the leaf in the tree of the key-query: $\mathbf{k}_{\ell,\lambda}^*$ is thus the specific key for leaf λ in the global ℓ-th key. The variable k will be used in hybrid proofs to specify a particular key-query index. We will denote K the maximal number of key-queries. Then $1 \leq \ell, k \leq K$.

Game $\mathbf{G}_0$: This is the real game where the simulator generates all the private information and sets $\mathsf{PK} = \{(\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}_1^*), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_1^*, \mathbf{d}_2^*, \mathbf{d}_3^*)\}$ and $\mathsf{MK} = \{\mathbf{b}_3^*\}$. The public parameters PK are provided to the adversary

OKeyGen($\mathcal{T}_\ell$): The adversary is allowed to issue KeyGen-queries on an access-tree $\mathcal{T}_\ell$ (for the ℓ-th query), for which the challenger chooses a random scalar $a_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$ and a random $a_{\ell,0}$-labeling $(a_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_\ell$, and builds the key:

$$\mathbf{k}_{\ell,0}^* = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}_{\ell,\lambda}^* = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0)_{\mathbb{D}^*}$$

for all the leaves λ, where $t_{\ell,\lambda} = A(\lambda)$ is the attribute associated to the leaf λ in $\mathcal{T}_\ell$ and $\pi_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q$. The decryption key dk_ℓ is then $(\mathbf{k}_{\ell,0}^*, (\mathbf{k}_{\ell,\lambda}^*)_\lambda)$;

RoREncaps(Γ): On the unique query on a set of attributes Γ, the challenger chooses random scalars $\omega, \xi, \xi' \xleftarrow{\$} \mathbb{Z}_q$. It then sets $K_0 = g_t^\xi$ and $K_1 = g_t^{\xi'}$. It generates the ciphertext $C = (\mathbf{c}_0, (\mathbf{c}_t)_{t\in\Gamma})$ where

$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1, t), \omega, 0, 0, 0)_{\mathbb{D}}$$

for all the attributes $t \in \Gamma$ and $\sigma_t \xleftarrow{\$} \mathbb{Z}_q$. According to the real or random game (bit $b \xleftarrow{\$} \{0, 1\}$), one outputs (K_b, C).

Eventually, on adversary's guess b' for b, if for some $\mathcal{T}_\ell$, $\mathcal{T}_\ell(\Gamma) = 1$, then $\beta \xleftarrow{\$} \{0,1\}$, otherwise $\beta = b'$. Then $\mathsf{Adv}_0 = \Pr[\beta = 1|b = 1] - \Pr[\beta = 1|b = 0]$.

In the next games, we gradually modify the simulations of $\mathsf{OKeyGen}$ and $\mathsf{RoREncaps}$ oracles, but always (at least) with random $\omega, \xi, \xi', (\sigma_t) \xleftarrow{\$} \mathbb{Z}_q$, $(a_{\ell,0}), (\pi_{\ell,\lambda}) \xleftarrow{\$} \mathbb{Z}_q$, and random $a_{\ell,0}$-labeling $(a_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_\ell$ for each $\mathsf{OKeyGen}$-query.

Game $\mathbf{G}_1$: One chooses random $\tau \xleftarrow{\$} \mathbb{Z}_q$, and sets (which differs for the ciphertext only)

$$\mathbf{c}_0 = (\omega, \tau, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1,t), \omega, \tau, 0, 0)_{\mathbb{D}}$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0)_{\mathbb{D}^*}$$

The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_1$: one applies the $\mathsf{SubSpace\text{-}Ind}$ property from Theorem 2, on $(\mathbb{B}, \mathbb{B}^*)_{1,2}$ and $(\mathbb{D}, \mathbb{D}^*)_{3,4}$. Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \tau \bmod q$ with either $\tau = 0$ or $\tau \xleftarrow{\$} \mathbb{Z}_q^*$, which are indistinguishable situations under the DDH assumption.

Let us assume we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 6 respectively. Then we define the matrices

$$B = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{1,2} \quad B' = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{1,2} \quad D = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{3,4} \quad D' = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{3,4}$$
$$\mathbb{B} = B \cdot \mathbb{U} \qquad \mathbb{B}^* = B' \cdot \mathbb{U}^* \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

Note that we can compute all the basis vectors excepted $\mathbf{b}_2^*$ and $\mathbf{d}_4^*$, that nobody needs: the vectors below have these coordinates at zero. So one can set

$$\mathbf{c}_0 = (b, c, \xi)_{\mathbb{U}} = (b, \tau, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1,t), b, c, 0, 0)_{\mathbb{V}} = (\sigma_t(1,t), b, \tau, 0, 0)_{\mathbb{D}}$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0)_{\mathbb{D}^*}$$

When $\tau = 0$, this is exactly the previous game, with $\omega = b$, for a random τ, this is the current game: $\mathsf{Adv}_0 - \mathsf{Adv}_1 \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

Game $\mathbf{G}_2$: One continues to modify the ciphertext, with random $\tau, (z_t) \xleftarrow{\$} \mathbb{Z}_q$:

$$\mathbf{c}_0 = (\omega, \tau, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1,t), \omega, \tau, 0, \tau z_t)_{\mathbb{D}}$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0)_{\mathbb{D}^*}$$

The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_1$: one applies again the $\mathsf{SubSpace\text{-}Ind}$ property from Theorem 2, on $(\mathbb{D}, \mathbb{D}^*)_{(1,2),6}$. Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \zeta \bmod q$, with either $\zeta = 0$ or $\zeta \xleftarrow{\$} \mathbb{Z}_q^*$, which are indistinguishable situations under the DDH assumption.

Let us assume we start from random dual orthogonal bases $(\mathbb{B}, \mathbb{B}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 6 respectively. Then we define the matrices

$$D = \begin{pmatrix} 1 & 0 & a \\ 0 & 1 & a \\ 0 & 0 & 1 \end{pmatrix}_{1,2,6} \qquad D' = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -a & -a & 1 \end{pmatrix}_{1,2,6} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

Note that we can compute all the basis vectors excepted $\mathbf{d}_6^*$, that nobody needs: the vectors below have these coordinates at zero. One chooses additional random scalars $\alpha_t, \beta_t \xleftarrow{\$} \mathbb{Z}_q$ to

virtually set $b_t = \alpha_t \cdot b + \beta_t$ and $c_t = \alpha_t \cdot c + \beta_t \cdot a$, which makes $c_t - ab_t = \alpha_t \cdot \zeta$. One can set

$$\begin{aligned} \mathbf{c}_0 &= (\omega, \tau, \xi)_{\mathbb{B}} & \mathbf{c}_t &= (b_t(1,t), \omega, \tau, 0, c_t(t+1))_{\mathbb{V}} \\ & & &= (b_t(1,t), \omega, \tau, 0, c_t(t+1) - ab_t - ab_t t)_{\mathbb{D}} \\ & & &= (b_t(1,t), \omega, \tau, 0, c_t(t+1) - ab_t(1+t))_{\mathbb{D}} \\ & & &= (b_t(1,t), \omega, \tau, 0, \alpha_t \cdot \zeta \cdot (t+1))_{\mathbb{D}} \\ & & &= (b_t(1,t), \omega, \tau, 0, \tau z_t)_{\mathbb{D}} \\ \mathbf{k}^*_{\ell,0} &= (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} & \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

where $z_t = \alpha_t \cdot \zeta \cdot (t+1)/\tau$. When $\zeta = 0$, this is exactly the previous game, as $z_t = 0$, with $\pi_t = b_t = \alpha_t \cdot b + \beta_t$, whereas for a random ζ, this is the current game: $\mathsf{Adv}_1 - \mathsf{Adv}_2 \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

Game $\mathbf{G}_3$: We introduce a second independent $s_{\ell,0}$-labeling $s_{\ell,\lambda}$ for each access-tree $\mathcal{T}_\ell$ and a random $r_{\ell,0}$ to define

$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, r_{\ell,0}, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, s_{\ell,\lambda}/z_{t_{k,\lambda}})_{\mathbb{D}^*}$$

But to this, we move to a sub-sequence of hybrid games, with distinct ways for answering the $k-1$ first key queries and the last ones, as explained on Figure 5.2: for the ℓ-th key generation query on $\mathcal{T}_\ell$, the challenger chooses three random scalars $a_{\ell,0}, r_{\ell,0}, s_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$, and two random $a_{\ell,0}$-labeling $(a_{\ell,\lambda})_\lambda$ and $s_{\ell,0}$-labeling $(s_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_\ell$, and builds the key $(\mathbf{k}^*_{\ell,0}, (\mathbf{k}^*_{\ell,\lambda})_\lambda)$, with $\pi_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q$:

$$\begin{aligned} \ell < k \quad & \mathbf{k}^*_{\ell,0} = (a_{\ell,0}, r_{\ell,0}, 1)_{\mathbb{B}^*} & \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, s_{\ell,\lambda}/z_{t_{\ell,\lambda}})_{\mathbb{D}^*} \\ \ell \geq k \quad & \mathbf{k}^*_{\ell,0} = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} & \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

For this game, we have to anticipate the values z_t, for each attribute t, before knowing Γ, for the challenge ciphertext, as we have to introduce $z_{t_{\ell,\lambda}}$ during the creation of the leaves. These z_t are thus random values chosen as soon as an attribute t is involved in the security game.

When $k = 1$, this is exactly the game $\mathbf{G}_2$: $\mathbf{G}_2 = \mathbf{G}_{2.1.0}$, whereas for $k = K+1$ this is exactly the expected game $\mathbf{G}_3$: $\mathbf{G}_3 = \mathbf{G}_{2.K+1.0}$. We now consider any $k \in \{1, \ldots, K\}$, to show that $\mathbf{G}_{2.k.3} = \mathbf{G}_{2.k+1.0}$, where all the keys for $\ell \neq k$ will be defined using the basis vectors of $(\mathbb{B}^*, \mathbb{D}^*)$ and known scalars. We only focus on the k-th key and the ciphertext, but still with random $\omega, \tau, \xi, \xi', (\sigma_t), (z_t) \xleftarrow{\$} \mathbb{Z}_q$, random $a_{k,0}, (\pi_{k,\lambda}) \xleftarrow{\$} \mathbb{Z}_q$, as well as a random $a_{k,0}$-labeling $(a_{k,\lambda})_\lambda$ of the access-tree $\mathcal{T}_k$, but also $s_{k,0} \xleftarrow{\$} \mathbb{Z}_q$ and a second independent random $s_{k,0}$-labeling $(s_{k,\lambda})_\lambda$ of the access-tree $\mathcal{T}_k$:

Game $\mathbf{G}_{2.k.0}$: This is exactly as described above, for $\ell = k$:

$$\begin{aligned} \mathbf{c}_0 &= (\omega, \tau, \xi)_{\mathbb{B}} & \mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t)_{\mathbb{D}} \\ \mathbf{k}^*_{k,0} &= (a_{k,0}, 0, 1)_{\mathbb{B}^*} & \mathbf{k}^*_{k,\lambda} &= (\pi_{k,\lambda}(t_{k,\lambda}, -1), a_{k,\lambda}, 0, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

Game $\mathbf{G}_{2.k.1}$: One now introduces the second labeling:

$$\begin{aligned} \mathbf{c}_0 &= (\omega, \tau, \xi)_{\mathbb{B}} & \mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t)_{\mathbb{D}} \\ \mathbf{k}^*_0 &= (a_{k,0}, s_{k,0}, 1)_{\mathbb{B}^*} & \mathbf{k}^*_{k,\lambda} &= (\pi_{k,\lambda}(t_{k,\lambda}, -1), a_{k,\lambda}, s_{k,\lambda}, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

This game is indistinguishable from the previous one under the DDH assumption in $\mathbb{G}_2$: one applies the SubSpace-Ind property from Theorem 2 on $(\mathbb{B}^*, \mathbb{B})_{1,2}$ and $(\mathbb{D}^*, \mathbb{D})_{3,4}$. Indeed, we can consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \rho \bmod q$, with either $\rho = 0$ or $\rho \xleftarrow{\$} \mathbb{Z}_q^*$, which are indistinguishable situations under the DDH assumption.

Let us assume we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 6 respectively. Then we define the matrices

$$B' = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{1,2} \quad B = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{1,2} \quad D' = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{3,4} \quad D = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{3,4}$$
$$\mathbb{B}^* = B' \cdot \mathbb{U}^* \qquad \mathbb{B} = B \cdot \mathbb{U} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^* \qquad \mathbb{D} = D \cdot \mathbb{V}$$

Note that we can compute all the basis vectors excepted $\mathbf{b}_2$ and $\mathbf{d}_4$. But we can define the ciphertext vectors in the original bases $(\mathbb{U}, \mathbb{V})$, and all the keys in bases $(\mathbb{B}^*, \mathbb{D}^*)$, excepted the k-th one:

$$\begin{aligned}
\mathbf{c}_0 &= (\omega, \tau, \xi)_{\mathbb{U}} = (\omega + a\tau, \tau, \xi)_{\mathbb{B}} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t)_{\mathbb{V}} = (\sigma_t(1,t), \omega + a\tau, \tau, 0, \tau z_t)_{\mathbb{D}} \\
\mathbf{k}^*_{k,0} &= (b_0, 0, 1)_{\mathbb{B}^*} + (b, c, 0)_{\mathbb{U}^*} = (b_0, 0, 1)_{\mathbb{B}^*} + (b, \rho, 0)_{\mathbb{B}^*} = (b_0 + b, \rho, 1)_{\mathbb{B}^*} \\
\mathbf{k}^*_{k,\lambda} &= (\pi_{k,\lambda}(t_{k,\lambda}, -1), b_\lambda, 0, 0, 0)_{\mathbb{D}^*} + (0, 0, b \cdot b'_\lambda, c \cdot b'_\lambda, 0, 0)_{\mathbb{V}^*} \\
&= (\pi_{k,\lambda}(t_{k,\lambda}, -1), b_\lambda, 0, 0, 0)_{\mathbb{D}^*} + (0, 0, b \cdot b'_\lambda, \rho \cdot b'_\lambda, 0, 0)_{\mathbb{D}^*} \\
&= (\pi_{k,\lambda}(t_{k,\lambda}, -1), b_\lambda + b \cdot b'_\lambda, \rho \cdot b'_\lambda, 0, 0)_{\mathbb{D}^*}
\end{aligned}$$

with $b_0 \xleftarrow{\$} \mathbb{Z}_q$, a random b_0-labeling $(b_\lambda)_\lambda$, and a random 1-labeling $(b'_\lambda)_\lambda$ of $\mathcal{T}_k$. When $\rho = 0$, this is exactly the previous game, with $\omega = \omega + a\tau$, and $a_{k,0} = b_0 + b$, $a_{k,\lambda} = b_\lambda + b \cdot b'_\lambda$, whereas for a random ρ, this is the current game, with additional $s_{k,0} = \rho$, $s_{k,\lambda} = \rho \cdot b'_\lambda$: $\mathsf{Adv}_{2.k.0} - \mathsf{Adv}_{2.k.1} \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{2.k.2}$: With the same inputs, one just changes as follows

$$\mathbf{c}_0 = (\omega, \tau, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1,t), \omega, \tau, 0, \tau z_k)_{\mathbb{D}}$$
$$\mathbf{k}^*_{k,0} = (a_{k,0}, s_{k,0}, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda}, -1), a_{k,\lambda}, 0, 0, s_{k,\lambda}/z_k)_{\mathbb{D}^*}$$

Unfortunately, for the latter gap, which intuitively exploits the Swap-Ind property from Theorem 4, we cannot do all the changes at once. Then, the Index-Ind property will be applied first, with Theorem 6.

We will thus describe another sequence of games, as shown on Figure 5.3, where $\mathbf{G}_{2.k.1.p.0}$ with $p = 1$ is the previous game: $\mathbf{G}_{2.k.1} = \mathbf{G}_{2.k.1.1.0}$; for any p, $\mathbf{G}_{2.k.1.p.5}$ is $\mathbf{G}_{2.k.1.p+1.0}$; and $\mathbf{G}_{2.k.1.p.0}$ with $p = P + 1$ is the current game: $\mathbf{G}_{2.k.2} = \mathbf{G}_{2.k.1.P+1.0}$. For each p, we prove that

$$\mathsf{Adv}_{2.k.1.p.0} - \mathsf{Adv}_{2.k.1.p.5} \leq 2P \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 3 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t).$$

Hence, globally, we have

$$\mathsf{Adv}_{2.k.1} - \mathsf{Adv}_{2.k.2} \leq 2P^2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 3P \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t).$$

But before proving this huge gap, let us conclude the analysis.

Game $\mathbf{G}_{2.k.3}$: In the above game, to be a legitimate attack (that does not output a random bit β in the Finalize procedure, but the actual output b' of the adversary), for all the key queries $\mathcal{T}_\ell$, one must have $\mathcal{T}_\ell(\Gamma) = 0$. In particular, $\mathcal{T}_k(\Gamma) = 0$: this means that the access-tree is not accepting of Γ, and thus there must be at least one leaf whose attributes is not in Γ. More concretely, let t be one of these attributes, this means $\mathbf{c}_t$ is not provided to the adversary, and so no information about z_t is leaked. As the key only contains $s_{k,\lambda}/z_{t_{k,\lambda}}$, no information can leak about $s_{k,\lambda}$ either, since it is masked by this $z_{t_{k,\lambda}}$. Thus, only $(s_{k,\lambda})_{\lambda \in \mathcal{L}_\Gamma}$ is known, and by definition of labelings the root $s_{k,0}$ is unpredictable.

Remark 1 One may wonder whether previous keys that involve those $z_{t_{k,\lambda}}$ could leak some information and contradict the above argument. Let us focus on the leaf λ associated to the attribute p, and so the information one could get about z_p when $\mathbf{c}_p$ is not part of the challenge ciphertext. At least, this argument holds for the first key generation, when we are in the first sequence of games, in $\mathbf{G}_{2.k.2}$ with $k = 1$: z_p is only used in $\mathbf{c}_p$, that is not revealed to the adversary, and so $s_{1,\lambda}/z_p$ does not leak any information about $s_{1,\lambda}$. And this is the same for all the leaves associated to missing attributes. Then $s_{1,0}$ can definitely be replaced by a random and independent $r_{1,0}$: which is the current game $\mathbf{G}_{2.k.3}$ for $k = 1$. When we are in $\mathbf{G}_{2.k.2}$ for $k = 2$, the adversary may now have some information about $s_{1,\lambda}/z_p$ and $s_{2,\lambda}/z_p$, but no information about $s_{1,0}$ that has already been replaced by a random $r_{1,0}$, which makes $s_{1,\lambda}$ unpredictable, and so no additional information leaks about z_p: $s_{2,\lambda}$ is unpredictable. Again, the same argument holds for all the leaves associated to missing attributes: $s_{2,0}$ can also be replaced by a random and independent $r_{2,0}$.
This is the reason of this hybrid sequence of game: if we would have first introduced the z_p in all the keys, it would not have been possible to replace all the $s_{\ell,0}$ by $r_{\ell,0}$ in the end. This is only true when all the previous keys have already been modified.

One can thus modify the key generation algorithm for the k-th key, with an independent $r_{k,0} \xleftarrow{\$} \mathbb{Z}_q$:

$$\mathbf{c}_0 = (\omega, \tau, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1,t), \omega, \tau, 0, \tau z_t)_{\mathbb{D}}$$
$$\mathbf{k}^*_{k,0} = (a_{k,0}, r_{k,0}, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda}, -1), a_{k,\lambda}, 0, 0, s_{k,\lambda}/z_{t_{k,\lambda}})_{\mathbb{D}^*}$$

This concludes this sequence of sub-games with, for each k,

$$\mathsf{Adv}_{2.k.0} - \mathsf{Adv}_{2.k.3} \leq 2P^2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (3P+1) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t).$$

Hence, globally, we have

$$\mathsf{Adv}_2 - \mathsf{Adv}_3 \leq 2KP^2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (3P+1)K \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t).$$

Game $\mathbf{G}_4$: In this game, one chooses a random θ to define the matrices

$$B = \begin{pmatrix} 1 & -\theta \\ 0 & 1 \end{pmatrix}_{2,3} \qquad B' = \begin{pmatrix} 1 & 0 \\ \theta & 1 \end{pmatrix}_{2,3} \qquad \mathbb{B} = B \cdot \mathbb{U} \qquad \mathbb{B}^* = B' \cdot \mathbb{U}^*$$

which only modifies $\mathbf{b}_2$, which is hidden, and $\mathbf{b}^*_3$, which is kept secret:

$$\mathbf{c}_0 = (\omega, \tau, \xi)_{\mathbb{U}} = (\omega, \tau, \tau\theta + \xi)_{\mathbb{B}} = (\omega, \tau, \xi'')_{\mathbb{B}}$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, r_{\ell,0}, 1)_{\mathbb{U}^*} = (a_{\ell,0}, r'_{\ell,0}, 1)_{\mathbb{D}^*}$$

As a consequence, any value for θ can be used, without impacting the view of the adversary, as $r'_{\ell,0}$ is indeed independent of the other variables. In this last game, a random value ξ'' is used in the ciphertext, whereas $K_0 = g_t^{\xi}$ and $K_1 = g_t^{\xi'}$: the advantage of any adversary is 0 in this last game.

If we combine all the steps:

$$\begin{aligned}\mathsf{Adv}_0 &= \mathsf{Adv}_0 - \mathsf{Adv}_4 \\ &\leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 2KP^2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (3P+1)K \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) \\ &\leq 2(KP^2+1) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (3P+1)K \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)\end{aligned}$$

		$\mathbf{c}_0 =$	(ω	τ	ξ)	$\mathbf{h}_0^* =$	(δ	ρ 0)
$\mathbf{G}_{2.k.1.p.0}$	Hybrid game for $\mathbf{G}_{2.k.1}$, with $1 \leq p \leq P+1$ (from Figure 5.2)							
		$\mathbf{c}_t =$	($\sigma_t(1,t)$	ω	$\mid \tau$	0	τz_t)	
$t < p$		$\mathbf{h}_t^* =$	($\pi_t(t,-1)$	δ	$\mid 0$	0	ρ/z_t)	
$t \geq p$		$\mathbf{h}_t^* =$	($\pi_t(t,-1)$	δ	$\mid \rho$	0	0)	
$\mathbf{G}_{2.k.1.p.1}$	Formal basis change, on $(\mathbb{D},\mathbb{D}^*)_{4,5}$, to duplicate τ							
		$\mathbf{c}_t =$	($\sigma_t(1,t)$	ω	$\mid \tau$	τ	τz_t)	
$\mathbf{G}_{2.k.1.p.2}$	Swap-Ind Property, on $(\mathbb{D}^*,\mathbb{D})_{2,4,5}$, for 0 and ρ in $\mathbf{h}_p^*$ only							
		$\mathbf{c}_t =$	($\sigma_t(1,t)$	ω	$\mid \tau$	τ	τz_t)	
		$\mathbf{h}_p^* =$	($\pi_p(p,-1)$	δ	$\mid 0$	ρ	0)	
$\mathbf{G}_{2.k.1.p.3}$	Index-Ind Property, on $(\mathbb{D},\mathbb{D}^*)_{1,2,5}$, between τ and $\tau z_t/z_p$							
		$\mathbf{c}_p =$	($\sigma_p(1,p)$	ω	$\mid \tau$	τ	τz_p)	
$t \neq p$		$\mathbf{c}_t =$	($\sigma_t(1,t)$	ω	$\mid \tau$	$\tau z_t/z_p$	τz_t)	
$\mathbf{G}_{2.k.1.p.4}$	Formal basis change, on $(\mathbb{D},\mathbb{D}^*)_{5,6}$, to cancel τ							
		$\mathbf{c}_t =$	($\sigma_t(1,t)$	ω	$\mid \tau$	0	τz_t)	
		$\mathbf{h}_p^* =$	($\pi_p(p,-1)$	δ	$\mid 0$	α	ρ/z_p)	
$\mathbf{G}_{2.k.1.p.5}$	SubSpace-Ind Property, on $(\mathbb{D}^*,\mathbb{D})_{2,5}$, between α and 0							
		$\mathbf{c}_t =$	($\sigma_t(1,t)$	ω	$\mid \tau$	0	τz_t)	
$t < p$		$\mathbf{h}_t^* =$	($\pi_t(t,-1)$	δ	$\mid 0$	0	ρ/z_t)	
		$\mathbf{h}_p^* =$	($\pi_p(p,-1)$	δ	$\mid 0$	0	ρ/z_p)	
$t > p$		$\mathbf{h}_t^* =$	($\pi_t(t,-1)$	δ	$\mid \rho$	0	0)	

Figure 5.3: Sequence of sub-games on the P attributes for the IND-security proof of our KP-ABE, where $\mathbf{k}_{\ell,0}^* = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} + s_{\ell,0} \cdot \mathbf{h}_0^*$ and $\mathbf{k}_{\ell,\lambda}^* = (\Pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0)_{\mathbb{D}^*} + s_{\ell,\lambda} \cdot \mathbf{h}_{t_{k,\lambda}}^*$, for all the leaves λ of all the keys ℓ, with $\mathbf{h}_0^* = (\delta, \rho, 0)_{\mathbb{B}^*}$ and $\mathbf{h}_0^* = (\pi_t(t,-1), \delta, \rho, 0, 0)_{\mathbb{D}^*}$ for all the possible attributes t. We only make the latter $(\mathbf{h}_0^*, (\mathbf{h}_t^*)_t)$ to evolve along this sequence.

We now present the sub-sequence of games for proving the gap from the above $\mathbf{G}_{2.k.1}$ to $\mathbf{G}_{2.k.2}$. We still focus on the challenge ciphertext $(\mathbf{c}_0, (\mathbf{c}_t))$ and the k-th key we will denote, for the sake of clarity, as

$$\mathbf{k}^*_{k,0} = (a_0, 0, 1)_{\mathbb{B}^*} + s_0 \cdot \mathbf{h}^*_0$$
$$\mathbf{k}^*_{k,\lambda} = (\Pi_{k,\lambda}(t_{k,\lambda}, -1), a_\lambda, 0, 0, 0)_{\mathbb{D}^*} + s_\lambda \cdot \mathbf{h}^*_{t_{k,\lambda}}$$

where $\mathbf{h}^*_0 = (\delta, \rho, 0)_{\mathbb{B}^*}$ and $\mathbf{h}^*_t = (\pi_t(t, -1), \delta, \rho, 0, 0)_{\mathbb{D}^*}$ for all the possible attributes. This corresponds to

$$a_{k,0} = a_0 + \delta \cdot s_0 \qquad a_{k,\lambda} = a_\lambda + \delta \cdot s_\lambda$$
$$s_{k,0} = \rho \cdot s_0 \qquad s_{k,\lambda} = \rho \cdot s_\lambda$$
$$\pi_{k,\lambda} = \Pi_{k,\lambda} + s_\lambda \cdot \pi_{t_{k,\lambda}}$$

All the other keys will be generated using the basis vectors: we stress that they all have a zero 5-th component, then $\mathbf{d}^*_5$ will not be needed. In the new hybrid game, the critical point will be the p-th attribute, where, when $p = 1$, $\mathbf{G}_{2.k.1.p.0}$ is exactly the above Game $\mathbf{G}_{2.k.1}$, and when $p = P + 1$ this is the above Game $\mathbf{G}_{2.k.2}$. And it will be clear, for any p, that $\mathbf{G}_{2.k.1.p.5} = \mathbf{G}_{2.k.1.p+1.0}$: with random $\omega, \tau, \xi, \xi', \delta, \rho, (z_t), (\sigma_t), (\pi_t) \xleftarrow{\$} \mathbb{Z}_q$,

Game $\mathbf{G}_{2.k.1.p.0}$: One defines the hybrid game for p:

$$\mathbf{c}_0 = (\omega, \tau, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1, t), \omega, \tau, 0, \tau z_t)_{\mathbb{D}}$$
$$\mathbf{h}^*_0 = (\delta, \rho, 0)_{\mathbb{B}^*} \qquad \mathbf{h}^*_t = (\pi_t(t, -1), \delta, 0, 0, \rho/z_t)_{\mathbb{D}^*} \qquad t < p$$
$$\mathbf{h}^*_t = (\pi_t(t, -1), \delta, \rho, 0, 0)_{\mathbb{D}^*} \qquad t \geq p$$

Game $\mathbf{G}_{2.k.1.p.1}$: One defines the matrices

$$D = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix}_{4,5} \qquad D' = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}_{4,5} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

which modifies the hidden vectors $\mathbf{d}_4$ and $\mathbf{d}^*_5$, and so are not in the view of the adversary:

$$\mathbf{c}_t = (\sigma_t(1, t), \omega, \tau, 0, \tau z_t)_{\mathbb{V}} = (\sigma_t(1, t), \omega, \tau, \tau, \tau z_t)_{\mathbb{D}}$$
$$\mathbf{h}^*_t = (\pi_t(t, -1), \delta, 0, 0, \rho/z_t)_{\mathbb{V}^*} = (\pi_t(t, -1), \delta, 0, 0, \rho/z_t)_{\mathbb{D}^*} \qquad t < p$$
$$\mathbf{h}^*_t = (\pi_t(t, -1), \delta, \rho, 0, 0)_{\mathbb{V}^*} = (\pi_t(t, -1), \delta, \rho, 0, 0)_{\mathbb{D}^*} \qquad t \geq p$$

For all the other keys, as the 5-th component is 0, the writing in basis $\mathbb{V}^*$ is the same in basis $\mathbb{D}^*$. Hence, the perfect indistinguishability between the two games: $\mathsf{Adv}_{2.k.1.p.1} = \mathsf{Adv}_{2.k.1.p.0}$.

Game $\mathbf{G}_{2.k.1.p.2}$: We apply the Swap-Ind property from Theorem 4, on $(\mathbb{D}^*, \mathbb{D})_{2,4,5}$: Indeed, we can consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \theta \bmod q$ with either $\theta = 0$ or $\theta = \rho$, which are indistinguishable situations under the DSDH assumption. Let us assume we start from random dual orthogonal bases $(\mathbb{B}, \mathbb{B}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 6 respectively. Then we define the matrices

$$D' = \begin{pmatrix} 1 & a & -a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{2,4,5} \qquad D = \begin{pmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ a & 0 & 1 \end{pmatrix}_{2,4,5} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^* \qquad \mathbb{D} = D \cdot \mathbb{V}$$

Note that we can compute all the basis vectors excepted $\mathbf{d}_4, \mathbf{d}_5$, but we define the ciphertext on the original basis $\mathbb{V}$:

$$\begin{aligned}
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, \tau, \tau z_t)_{\mathbb{V}} = (\sigma_t, \sigma_t t + a\tau - a\tau, \omega, \tau, \tau, \tau z_t)_{\mathbb{D}} \\
&= (\sigma_t(1,t), \omega, \tau, \tau, \tau z_t)_{\mathbb{D}} \\
\mathbf{h}^*_t &= (\pi_t(t,-1), \delta, 0, 0, \rho/z_t)_{\mathbb{D}^*} && t < p \\
\mathbf{h}^*_p &= (\pi_p(p,-1), \delta, \rho, 0, 0)_{\mathbb{D}^*} + (b(p,-1), 0, -c, c, 0)_{\mathbb{V}^*} \\
&= (\pi_p(p,-1), \delta, \rho, 0, 0)_{\mathbb{D}^*} + (b(p,-1), 0, ab-c, -ab+c, 0)_{\mathbb{D}^*} \\
&= (\pi_p(p,-1), \delta, \rho-\theta, \theta, 0)_{\mathbb{D}^*} \\
\mathbf{h}^*_t &= (\pi_t(t,-1), \delta, \rho, 0, 0)_{\mathbb{D}^*} && t > p
\end{aligned}$$

With $\theta = 0$, this is as in the previous game, with $\theta = \rho$, this is the current game: $\mathsf{Adv}_{2.k.1.p.1} - \mathsf{Adv}_{2.k.1.p.2} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{2.k.1.p.3}$: We keep the τ value (at the 5-th hidden position) in the ciphertext for the p-th attribute only, and replace all the other values by $\tau z_t/z_p$:

$$\begin{aligned}
\mathbf{c}_p &= (\sigma_t(1,t), \omega, \tau, \tau, \tau z_t)_{\mathbb{D}} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, \tau z_t/z_p, \tau z_t)_{\mathbb{D}} && t \neq p
\end{aligned}$$

To show this is possible without impacting the other vectors, we use the Index-Ind property from Theorem 6, but in another level of sequence of hybrid games, for $\gamma \in \{1, \ldots, P\} \backslash \{p\}$:

Game $\mathbf{G}_{2.k.1.p.2.\gamma}$: We consider

$$\begin{aligned}
\mathbf{c}_p &= (\sigma_p(1,p), \omega, \tau, \tau, \tau z_p)_{\mathbb{D}} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, \tau z_t/z_p, \tau z_t)_{\mathbb{D}} && p \neq t < \gamma \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, \tau, \tau z_t)_{\mathbb{D}} && t \geq \gamma \\
\mathbf{h}^*_t &= (\pi_t(t,-1), \delta, 0, 0, \rho/z_t)_{\mathbb{D}^*} && t < p \\
\mathbf{h}^*_p &= (\pi_p(p,-1), \delta, 0, \rho, 0)_{\mathbb{D}^*} \\
\mathbf{h}^*_t &= (\pi_t(t,-1), \delta, \rho, 0, 0)_{\mathbb{D}^*} && t > p
\end{aligned}$$

When $\gamma = 1$, this is the previous game: $\mathbf{G}_{2.k.1.p.2.1} = \mathbf{G}_{2.k.1.p.2}$, whereas with $\gamma = P + 1$, this is the current game: $\mathbf{G}_{2.k.1.p.2.P+1} = \mathbf{G}_{2.k.1.p.3}$.

For any $\gamma \in \{1, \ldots, P\}$, we consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \zeta \bmod q$, with either $\zeta = 0$ or $\zeta = \tau(z_\gamma/z_p - 1)$, which are indistinguishable situations under the DSDH assumption. We define the matrices

$$D = \frac{1}{p-\gamma} \times \begin{pmatrix} p & -\gamma & ap \\ -1 & 1 & -a \\ 0 & 0 & p-\gamma \end{pmatrix}_{1,2,5} \qquad D' = \begin{pmatrix} 1 & 1 & 0 \\ \gamma & p & 0 \\ -a & 0 & 1 \end{pmatrix}_{1,2,5}$$

and then $\mathbb{D} = D \cdot \mathbb{V}$, $\mathbb{D}^* = D' \cdot \mathbb{V}^*$: we cannot compute $\mathbf{d}^*_5$, but the components on this

vector are all 0 excepted for $\mathbf{h}^*_p$ we will define in $\mathbb{V}^*$:

$$\begin{aligned}
\mathbf{c}_p &= (\sigma_p(1,p),\omega,\tau,\tau,\tau z_p)_{\mathbb{D}} \\
\mathbf{c}_\gamma &= (b,0,\omega,\tau,\tau+c,\tau z_\gamma)_{\mathbb{V}} = (b,b\gamma,\omega,\tau,\tau+c-ab,\tau z_\gamma)_{\mathbb{D}} \\
&= (b(1,\gamma),\omega,\tau,\tau+\zeta,\tau z_\gamma)_{\mathbb{D}} \\
\mathbf{c}_t &= (\sigma_t(1,t),\omega,\tau,\tau z_t/z_p,\tau z_t)_{\mathbb{D}} && p\neq t<\gamma \\
\mathbf{c}_t &= (\sigma_t(1,t),\omega,\tau,\tau,\tau z_t)_{\mathbb{D}} && t>\gamma \\
\mathbf{h}^*_t &= (\pi_t(t,-1),\delta,0,0,\rho/z_t)_{\mathbb{D}^*} && t<p \\
\mathbf{h}^*_p &= ((p-\gamma)\cdot(\pi,0),\delta,0,\rho,0)_{\mathbb{V}^*} \\
&= (p\cdot\pi+ap\rho,-\pi-a\rho,\delta,0,\rho,0)_{\mathbb{D}^*} \\
&= ((\pi+a\rho)\cdot(p,-1),\delta,0,\rho,0)_{\mathbb{D}^*} \\
\mathbf{h}^*_t &= (\pi_t(t,-1),\delta,\rho,0,0)_{\mathbb{D}^*} && t>p
\end{aligned}$$

which is the hybrid game with $\pi_p=\pi+a\rho$ and the 5-th component of $\mathbf{c}_\gamma$ is $\tau+\zeta$, which is either τ when $\zeta=0$, and thus the game $\mathbf{G}_{2.k.1.p.2.\gamma}$ or $\tau z_\gamma/z_p$ when $\zeta=\tau z_\gamma/z_p-\tau$, which is $\mathbf{G}_{2.k.1.p.2.\gamma+1}$: hence, the distance between two consecutive games is bounded by $\mathsf{Adv}^{\mathsf{dsdh}}_{\mathbb{G}_1}(t)$.

Hence, we have $\mathsf{Adv}_{2.k.1.p.2}-\mathsf{Adv}_{2.k.1.p.3}\leq 2P\times\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

Game $\mathbf{G}_{2.k.1.p.4}$: We can now insert $1/z_p$ in the p-th last component, and then make some cleaning with the matrices, for $\alpha\xleftarrow{\$}\mathbb{Z}_q^*$

$$D=\begin{pmatrix}\alpha/\rho & 0\\ 1/z_p & 1\end{pmatrix}_{5,6} \qquad D'=\begin{pmatrix}\rho/\alpha & -\rho/\alpha z_p\\ 0 & 1\end{pmatrix}_{5,6}$$

and then $\mathbb{D}=D\cdot\mathbb{V}$, $\mathbb{D}^*=D'\cdot\mathbb{V}^*$. As the four vectors $\mathbf{d}_5,\mathbf{d}_6$ and $\mathbf{d}^*_5,\mathbf{d}^*_6$ are hidden, the modifications will not impact the view of the adversary. This consists in applying successively the matrices :

$$D=\begin{pmatrix}1/z_p & 0\\ 0 & 1\end{pmatrix}_{5,6} \qquad D=\begin{pmatrix}1 & 0\\ 1 & 1\end{pmatrix}_{5,6} \qquad D=\begin{pmatrix}\alpha z_p/\rho & 0\\ 0 & 1\end{pmatrix}_{5,6}$$

Then, working in $(\mathbb{V},\mathbb{V}^*)$ gives, in $(\mathbb{D},\mathbb{D}^*)$:

$$\begin{aligned}
\mathbf{c}_p &= (\sigma_p(1,p),\omega,\tau,\tau,\tau z_p)_{\mathbb{V}} = (\sigma_p(1,p),\omega,\tau,0,\tau z_p)_{\mathbb{D}} \\
\mathbf{c}_t &= (\sigma_t(1,t),\omega,\tau,\tau z_t/z_p,\tau z_t)_{\mathbb{V}} \\
&= (\sigma_t(1,t),\omega,\tau,(\tau z_t/z_p-\tau z_t/z_p)\cdot\rho/\alpha,\tau z_t)_{\mathbb{D}} && t\neq p \\
&= (\sigma_t(1,t),\omega,\tau,0,\tau z_t)_{\mathbb{D}} \\
\mathbf{h}^*_t &= (\pi_t(t,-1),\delta,0,0,\rho/z_t)_{\mathbb{V}^*} = (\pi_t(t,-1),\delta,0,0,\rho/z_t)_{\mathbb{D}^*} && t<p \\
\mathbf{h}^*_p &= (\pi_p(p,-1),\delta,0,\rho,0)_{\mathbb{V}^*} = (\pi_p(p,-1),\delta,0,\alpha,\rho/z_p)_{\mathbb{D}^*} \\
\mathbf{h}^*_t &= (\pi_t(t,-1),\delta,\rho,0,0)_{\mathbb{V}^*} = (\pi_t(t,-1),\delta,\rho,0,0)_{\mathbb{V}^*} && t>p
\end{aligned}$$

We stress again that for all the other keys, as the 5-th component is 0, the writing in basis $\mathbb{V}^*$ is the same in basis $\mathbb{D}^*$. Hence, the perfect indistinguishability between the two games: $\mathsf{Adv}_{2.k.1.p.4}=\mathsf{Adv}_{2.k.1.p.3}$.

Game $\mathbf{G}_{2.k.1.p.5}$: We can now remove the α value in the p-th element of the key: We can consider a triple $(a\cdot G_2,b\cdot G_2,c\cdot G_2)$, where $c=ab+\alpha \bmod q$, with either $\alpha=0$ or $\alpha\xleftarrow{\$}\mathbb{Z}_q^*$, which are indistinguishable situations under the DDH assumption. We define the matrices

$$D'=\begin{pmatrix}1 & a\\ 0 & 1\end{pmatrix}_{2,5} \qquad D=\begin{pmatrix}1 & 0\\ -a & 1\end{pmatrix}_{2,5}$$

and then $\mathbb{D} = D \cdot \mathbb{V}$, $\mathbb{D}^* = D' \cdot \mathbb{V}^*$: we cannot compute $\mathbf{d}_5$, but the components on this vector are all 0:

$$
\begin{aligned}
\mathbf{c}_p &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t)_{\mathbb{D}} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t)_{\mathbb{D}} && t \neq p \\
\mathbf{h}^*_t &= (\pi_t(t,-1), \delta, 0, 0, \rho/z_t)_{\mathbb{D}^*} && t < p \\
\mathbf{h}^*_p &= (-b(p,-1), \delta, 0, c, \rho/z_p)_{\mathbb{V}^*} = (-b(p,-1), \delta, 0, c-ab, \rho/z_p)_{\mathbb{D}^*} \\
&= (-b(p,-1), \delta, 0, \alpha, \rho/z_p)_{\mathbb{D}^*} \\
\mathbf{h}^*_t &= (\pi_t(t,-1), \delta, \rho, 0, 0)_{\mathbb{D}^*} && t > p
\end{aligned}
$$

which is the either the previous game when $\alpha \neq 0$ or the current game with $\alpha = 0$, where $\pi_p = -b$: $\mathsf{Adv}_{2.k.1.p.4} - \mathsf{Adv}_{2.k.1.p.5} \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

5.3 An Attribute-Based Signature (ABS) Construction

In this section, we describe an ABS scheme with perfect anonymity. The construction is inspired by our KP-ABE [DGP22], while the proof is an optimized version (regarding the number of bases) of Okamoto-Takashima's base scheme [OT13]. We also handle unbounded universe of attributes.

This basic scheme is derived from a KP-ABE, where the signature can be seen as a decryption key associated to a policy, and the verification algorithm tries to decrypt a ciphertext on a set of attributes. If decryption works, the signature is valid, otherwise the signature is invalid. In a later section about our contributions, we will add on this basic scheme two kinds of delegations, and the traceability of signers. To anticipate delegations of attributes and policies, we already separate the message and the policy in the signing process, which is a critical difference from the [OT13] construction.

5.3.1 Complementary Properties

Dual-Trees Before going into the construction, we need to introduce a new notion related to access-trees.

Definition 8 We call the *dual-tree* $\mathcal{T}^*$ of $\mathcal{T}$ the access-tree which is the exact same tree as $\mathcal{T}$, except that all OR gates in $\mathcal{T}$ become AND gates in $\mathcal{T}^*$, and conversely all AND gates in $\mathcal{T}$ become OR gates in $\mathcal{T}^*$.

We note that the structure of $\mathcal{T}$ and $\mathcal{T}^*$ is identical, in particular all leaves are present on both trees, thus we will abuse notations and consider $\mathcal{L}_{\mathcal{T}} = \mathcal{L}_{\mathcal{T}^*}$ when there is no ambiguity.

Dual-trees will be crucial in our signature constructions. They will allow the signer to share enough information to the verifier for the verification of the signature, to prove correctness. At the same time, it prevents revealing anything about the validity of the access-tree other than the signer could sign it with its attributes, to ensure the anonymity property that we will define later. This is formalized in the two next propositions.

Proposition 1 *If $(a_\lambda)_\lambda$ is an a_0-labeling of $\mathcal{T}$, and $(b_\lambda)_\lambda$ is a b_0-labeling of its dual tree $\mathcal{T}^*$, then $\sum_{\lambda\in\mathcal{L}} a_\lambda b_\lambda = a_0 b_0$.*

Intuitively, there is always an OR-gate (from either $\mathcal{T}$ or $\mathcal{T}'$) which creates a common factor when recursively evaluating the product at each node on both trees.

Proof We proceed by induction on the depth ℓ of the access-trees $\mathcal{T}$.

When $\ell = 1$, there are only two cases: any tree is either a root node labeled with an AND gate and any number of children, or it is a root node labeled with an OR gate and any number of children, and $\mathcal{T}^*$ is the alternative situation. Hence, by considering $\mathcal{T}$ with an AND-gate at the root and $\mathcal{T}^*$ with an OR-gate at the root, we address both cases at once (we just have to exchange $\mathcal{T}$ and $\mathcal{T}^*$ for the other case): $(a_\lambda)_\lambda$ is an a_0-labeling of $\mathcal{T}$, and $(b_\lambda)_\lambda$ is a b_0-labeling of $\mathcal{T}^*$. Since $\mathcal{L} = \mathsf{children}(\rho)$, because of the AND-gate, $a_0 = \sum_{\kappa\in\mathcal{L}} a_\kappa$, and because of the OR-gate, for all $\kappa \in \mathcal{L}$, $b_\kappa = b_0$: $\sum_{\lambda\in\mathcal{L}} a_\lambda b_\lambda = b_0 \sum_{\lambda\in\mathcal{L}} a_\lambda = a_0 b_0$.

Now we suppose, for the induction step, that this property holds for all $k \leq \ell \in \mathbb{N}$, and prove it holds for $\ell+1$ as well. Again, let us consider $\mathcal{T}$ an access-tree of depth $\ell+1$ with an AND-gate at the root, and $(a_\lambda)_\lambda$ an a_0-labeling of $\mathcal{T}$, then $\mathcal{T}^*$ is an access-tree of depth $\ell+1$ with an OR-gate at the root, $(b_\lambda)_\lambda$ is a b_0-labeling of $\mathcal{T}^*$ (the other case just consists in switching $\mathcal{T}$ and $\mathcal{T}^*$). Now, roots' children are subtrees of depth at most ℓ. We note $\mathcal{T}_\kappa$ the subtree rooted at $\kappa \in \mathsf{children}(\rho)$, and $\mathcal{L}_\kappa$ its leaves, and note that $(\mathcal{L}_\kappa)_\kappa$ is a strict partition of $\mathcal{L}$. As the dual tree is built by just switching AND/OR gates, the dual-tree of $\mathcal{T}_\kappa$ (the subtree rooted at κ) is the subtree of the dual-tree of $\mathcal{T}^*$ rooted at κ, that we can thus denote $\mathcal{T}^*_\kappa$ without any ambiguity.

By definition of the labelings, $a_0 = \sum_{\kappa\in\mathsf{children}(\rho)} a_\kappa$ for $\mathcal{T}$, and in $\mathcal{T}^*$, for all $\kappa \in \mathsf{children}(\rho)$, $b_\kappa = b_0$. Then, as above, $a_0 b_0 = \sum_{\kappa\in\mathsf{children}(\rho)} a_\kappa b_\kappa$. However, we also know from the induction

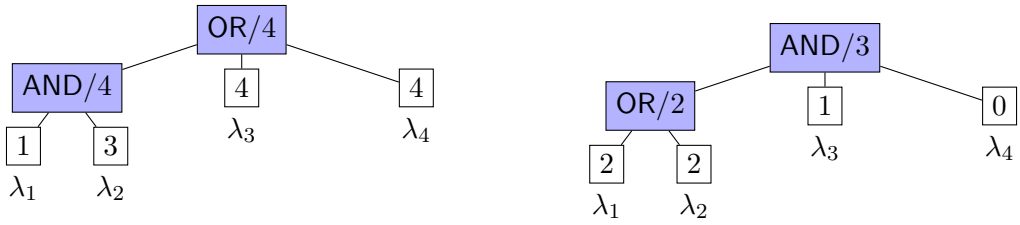

Figure 5.4: An access-tree (left) and its dual tree (right), both with random labelings in $\mathbb{Z}/7\mathbb{Z}$. One can see that $\sum_{\lambda\in\mathcal{L}} a_\lambda b_\lambda = a_0 b_0 = 5 \bmod 7$.

hypothesis on all the subtrees $\mathcal{T}_\kappa$ and $\mathcal{T}_\kappa^*$, rooted at $\kappa \in \mathsf{children}(\rho)$ over the sets of leaves $\mathcal{L}_\kappa$, of depth at most ℓ, $\sum_{\lambda\in\mathcal{L}_\kappa} a_\lambda b_\lambda = a_\kappa b_\kappa$. Because of the partition property of the $\mathcal{L}_\kappa$'s into $\mathcal{L}$, $\sum_{\lambda\in\mathcal{L}} a_\lambda b_\lambda = \sum_{\kappa\in\mathsf{children}(\rho)} \sum_{\lambda\in\mathcal{L}_\kappa} a_\lambda b_\lambda = \sum_{\kappa\in\mathsf{children}(\rho)} a_\kappa b_\kappa = a_0 b_0$.

Proposition 2 *Let $\mathcal{T}$ be an access-tree and Γ a set of attributes so that $\mathcal{T}(\Gamma) = 1$. Then, for any Evaluation Pruned Tree $\mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma)$, there is a 1-labeling $(b_\lambda)_\lambda$ of the dual $\mathcal{T}^*$ which verifies: $b_\lambda = 1$ for all $\lambda \in \mathcal{L}_{\mathcal{T}'}$ and $b_\lambda = 0$ for all $\lambda \notin \mathcal{L}_{\mathcal{T}'}$.*

Intuitively, if a tree $\mathcal{T}$ is satisfied by a set of attributes, we can associate the value 1 to the leaves of the satisfied attributes in the tree (which defines the pruned tree $\mathcal{T}'$), and this association is effectively a 1-labeling of the dual-tree $\mathcal{T}'^* \subset \mathcal{T}^*$, but also a 1-labeling of the dual-tree $\mathcal{T}^*$. This 1-labeling of $\mathcal{T}^*$, which is specific to the choice of $\mathcal{T}'$, can then be randomizez for anonymity with any 0-labeling of $\mathcal{T}^*$, through the linearity of labelings on $\mathcal{T}^*$, while maintaining correctness.

Proof Let $\mathcal{T}'$ be an Evaluation Pruned Tree from $\mathsf{EPT}(\mathcal{T}, \Gamma)$, where $\mathcal{T}(\Gamma) = 1$. By definition of an EPT, $\mathcal{T}'$ has only one child on OR gates that come from $\mathcal{T}$, and all children on AND gates that come from $\mathcal{T}$. This translates to its dual $\mathcal{T}'^*$ having AND gates with only one child, and OR gates having all children. From there, we can easily construct a 1-labeling of $\mathcal{T}'^*$ noted $(b_\lambda)_\lambda$ where $b_\lambda = 1$, for all $\lambda \in \mathcal{L}_{\mathcal{T}'}$. Indeed, since AND gates have a unique child, its label is identical to the one of the parent, and OR gates always have identical labels for the children than the one of the parent, all the 1's then go up to the root. We expand this into a 1-labeling of $\mathcal{T}^*$ by setting $b_\lambda = 0$ for all $\lambda \notin \mathcal{L}_{\mathcal{T}'}$.

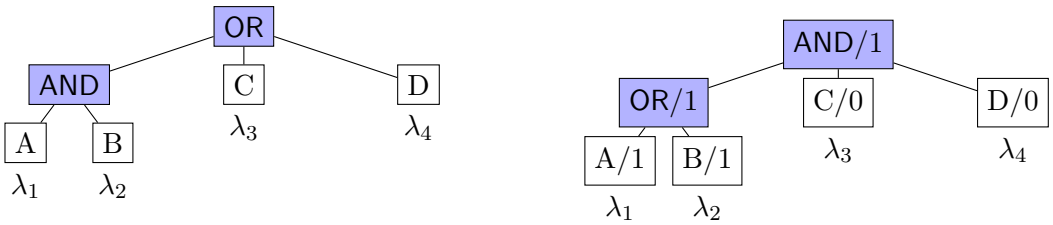

Figure 5.5: An access-tree fulfilled by the set $\{A, B\}$ (left). One can extract a 1-labeling from its dual-tree (right) which has values 1 on leaves $\{A, B\}$ and 0 on all other leaves.

Indexing of Size Three Contrary to our ABE which rely on an indexing on a subspace of size two, our ABS will rely on an indexing of size three. We detail the theorem, and provide a sketch for the proof, but we refer to Theorem 7 for the detailled proof, since both follow the exact same structure.

Theorem 9 (Index-Ind Property) *In $(\mathbb{B}, \mathbb{B}^*)$ of dimension 6, from the view of basis vectors $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_1^*, \mathbf{b}_2^*, \mathbf{b}_3^*, \mathbf{b}_4^*)$, and any vector $\mathbf{u} = (\pi\cdot(x+\rho x', -1, -\rho), \beta, 0, 0)_{\mathbb{B}}$, for chosen $x, x', \beta \in$*

$\mathbb{Z}_q$, but unknown random $\pi \xleftarrow{\$} \mathbb{Z}_q$, and for any chosen $(y, y') \neq (x, x') \in \mathbb{Z}_q^2$, one cannot distinguish the vectors $\mathbf{v}_0^* = (\sigma \cdot (1, y, y'), 0, 0, 0)_{\mathbb{B}^*}$ and $\mathbf{v}_1^* = (\sigma \cdot (1, y, y'), \alpha, 0, 0)_{\mathbb{B}^*}$, for chosen $\alpha \in \mathbb{Z}_q$, but unknown random $\sigma \xleftarrow{\$} \mathbb{Z}_q$, with an advantage better than $4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

We stress that in this theorem, π and σ are unknown and not under control, but $\sigma \cdot G_2$ can be known, as one can note along the games in the proof below.

$\mathbf{G}_1$	Initial situation before indexing							
	$\mathbf{u}$	$= ($	$\pi((x+\rho x'),$	$-1,$	$-\rho)$	β	0	0	$)_{\mathbb{B}}$
	$\mathbf{v}^*$	$= ($	$\sigma(1,$	$y,$	$y')$	0	0	0	$)_{\mathbb{B}^*}$
$\mathbf{G}_2$	SubSpace-Ind on (1,2,5,6)							
	$\mathbf{u}$	$= ($	$\pi((x+\rho x'),$	$-1,$	$-\rho)$	β	$\zeta(x+\rho x',$ | $-1)$	$)_{\mathbb{B}}$
	$\mathbf{v}^*$	$= ($	$\sigma(1,$	$y,$	$y')$	0	0	0	$)_{\mathbb{B}^*}$
$\mathbf{G}_3$	SubSpace-Ind on (1,2,3,5,6)							
	$\mathbf{u}$	$= ($	$\pi((x+\rho x'),$	$-1,$	$-\rho)$	β	$\zeta(x+\rho x',$ | $-1)$	$)_{\mathbb{B}}$
	$\mathbf{v}^*$	$= ($	$\sigma(1,$	$y,$	$y')$	0	$\theta(1,$	$y+\rho y')$	$)_{\mathbb{B}^*}$
$\mathbf{G}_4$	Formal change on (5,6)							
	$\mathbf{u}$	$= ($	$\pi((x+\rho x'),$	$-1,$	$-\rho)$	β	u_1	$u_2)$	$)_{\mathbb{B}}$
	$\mathbf{v}^*$	$= ($	$\sigma(1,$	$y,$	$y')$	0	v_1	v_2	$)_{\mathbb{B}^*}$
$\mathbf{G}_5$	SubSpace-Ind on (5,4)							
	$\mathbf{u}$	$= ($	$\pi((x+\rho x'),$	$-1,$	$-\rho)$	β	u_1	$u_2)$	$)_{\mathbb{B}}$
	$\mathbf{v}^*$	$= ($	$\sigma(1,$	$y,$	$y')$	α	v_1	v_2	$)_{\mathbb{B}^*}$
$\mathbf{G}_6$	Formal change on (5,6)							
	$\mathbf{u}$	$= ($	$\pi((x+\rho x'),$	$-1,$	$-\rho)$	β	$\zeta(x+\rho x',$ | $-1)$	$)_{\mathbb{B}}$
	$\mathbf{v}^*$	$= ($	$\sigma(1,$	$y,$	$y')$	α	$\theta(1,$	$y+\rho y')$	$)_{\mathbb{B}^*}$
$\mathbf{G}_7$	SubSpace-Ind on (1,2,3,5,6)							
	$\mathbf{u}$	$= ($	$\pi((x+\rho x'),$	$-1,$	$-\rho)$	β	$\zeta(x+\rho x',$ | $-1)$	$)_{\mathbb{B}}$
	$\mathbf{v}^*$	$= ($	$\sigma(1,$	$y,$	$y')$	α	0	0	$)_{\mathbb{B}^*}$
$\mathbf{G}_8$	SubSpace-Ind on (1,2,5,6)							
	$\mathbf{u}$	$= ($	$\pi((x+\rho x'),$	$-1,$	$-\rho)$	β	0	0	$)_{\mathbb{B}}$
	$\mathbf{v}^*$	$= ($	$\sigma(1,$	$y,$	$y')$	α	0	0	$)_{\mathbb{B}^*}$

Figure 5.6: Sequence of Games for Index-Ind Property of Size 3.

5.3.2 Description of our ABS Scheme

In this scheme, signatures will only depend on the policies, and will be perfectly independent of the signers for anonymity. We consider attributes t in a large universe $\mathcal{U}$, and policies expressed as access trees $\mathcal{T}$ with leaves λ in $\mathcal{L}$. We will express all the elements as vector in the correctness analysis for more clarity.

Setup(1^κ)**.** The algorithm chooses three random dual orthogonal bases, in a pairing-friendly setting $\mathcal{PG} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$:

$$\mathbb{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_4) \qquad \mathbb{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_{10}) \qquad \mathbb{H} = (\mathbf{h}_1, \ldots, \mathbf{h}_8)$$
$$\mathbb{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_4^*) \qquad \mathbb{D}^* = (\mathbf{d}_1^*, \ldots, \mathbf{d}_{10}^*) \qquad \mathbb{H}^* = (\mathbf{h}_1^*, \ldots, \mathbf{h}_8^*).$$

It picks two hash functions $\mathcal{H}$ and $\mathcal{H}'$ onto $\mathbb{Z}_q$. It then sets public parameters $\mathsf{PK} = \{\mathcal{PG}, \mathcal{H}, \mathcal{H}', (\mathbf{b}_1, \mathbf{b}_3), (\mathbf{b}_2^*), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_5), (\mathbf{d}_1^*, \mathbf{d}_2^*, \mathbf{d}_3^*, \mathbf{d}_4^*), (\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3, \mathbf{h}_5), (\mathbf{h}_4^*)\}$, as well as the master secret key $\mathsf{MK} = \{(\mathbf{b}_1^*), (\mathbf{h}_1^*, \mathbf{h}_2^*, \mathbf{h}_3^*)\} \cup \mathsf{PK}$.

$\mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}, \Gamma)$**.** A random scalar $\delta_{\mathsf{id}} \xleftarrow{\$} \mathbb{Z}_q^*$ is associated to id, to define

$$\mathbf{k}_0^* = \delta_{\mathsf{id}} \cdot \mathbf{b}_1^* + \phi_0 \cdot \mathbf{b}_2^* \qquad \mathbf{k}_t^* = \delta_{\mathsf{id}} \cdot \mathbf{d}_1^* + \pi_t \cdot \mathbf{d}_2^* + t\pi_t \cdot \mathbf{d}_3^* \; \phi_t \cdot \mathbf{d}_4^*$$
$$\mathbf{r}_1^* = \delta_{\mathsf{id}} \cdot \mathbf{h}_1^* + \psi_1 \cdot \mathbf{h}_4^* \qquad \mathbf{r}_2^* = \delta_{\mathsf{id}} \cdot \mathbf{h}_2^* + \psi_2 \cdot \mathbf{h}_4^* \qquad \mathbf{r}_3^* = \delta_{\mathsf{id}} \cdot \mathbf{h}_3^* + \psi_3 \cdot \mathbf{h}_4^*$$

for all attributes $t \in \Gamma$, with $\phi_0, \psi_1, \psi_2, \psi_3, (\phi_t)_t, (\pi_t)_t \xleftarrow{\$} \mathbb{Z}_q^*$ for each t. The signing key $\mathsf{SK}_{\mathsf{id},\Gamma}$ is then set as $(\mathbf{k}_0^*, (\mathbf{k}_t^*)_{t\in\Gamma}, \mathbf{r}_1^*, \mathbf{r}_2^*, \mathbf{r}_3^*)$. It can be completed later for new attributes t, for id, with extra $\mathbf{k}_t^*$, using the same δ_{id}, specific to user id.

$\mathsf{Sig}(\mathsf{SK}_{\mathsf{id},\Gamma}, m, \mathcal{T})$**.** Let $\mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma)$ be an Evaluation Pruned Tree, $\nu, \xi, \zeta, (\omega_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q^*$, and $(\alpha_\lambda)_\lambda$ the 1-labeling of the dual-tree $\mathcal{T}^*$ specific to $\mathcal{T}'$, where $\alpha_\lambda = 1$ if $\lambda \in \mathcal{L}_{\mathcal{T}'}$, else $\alpha_\lambda = 0$. This is possible as $\mathcal{T}(\Gamma) = 1$ (see Proposition 2). Compute $(\beta_\lambda)_\lambda$ to be a random 0-labeling of $\mathcal{T}^*$. Take $(q_\lambda)_\lambda$ random scalars. Eventually, set, for $H = \mathcal{H}(\mathcal{T}), H' = \mathcal{H}'(m)$:

$$U^* = \xi\mathbf{k}_0^* + \zeta\mathbf{b}_2^* \qquad S_\lambda^* = \alpha_\lambda\xi \cdot \mathbf{k}_{t_\lambda}^* + \beta_\lambda \cdot \mathbf{d}_1^* + \omega_\lambda \cdot (\mathbf{d}_2^* + t_\lambda \cdot \mathbf{d}_3^*) + q_\lambda \cdot \mathbf{d}_4^*$$
$$V^* = \xi(\mathbf{r}_1^* + H \cdot \mathbf{r}_2^* + H' \cdot \mathbf{r}_3^*) + \nu \cdot \mathbf{h}_4^*$$

for all the leaves λ, where t_λ is the associated attribute of λ. The signature is $\sigma = (U^*, V^*, (S_\lambda^*)_\lambda)$.

$\mathsf{Verif}(\mathsf{PK}, m, \mathcal{T}, \sigma)$**.** Let $\kappa, \kappa_0, (\kappa_\lambda)_\lambda, s, s_0, \theta, (\theta_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q$. Let $(s_\lambda)_\lambda$ be a random s_0-labeling of $\mathcal{T}$, then set, for $\bar{H} = \mathcal{H}(\mathcal{T}), \bar{H}' = \mathcal{H}'(m)$:

$$u = -(s_0 + s) \cdot \mathbf{b}_1 + \kappa_0 \cdot \mathbf{b}_3 \qquad c_\lambda = s_\lambda \cdot \mathbf{d}_1 + (\theta_\lambda t_\lambda) \cdot \mathbf{d}_2 - \theta_\lambda \cdot \mathbf{d}_3 + \kappa_\lambda \cdot \mathbf{d}_5$$
$$v = (s + \theta\bar{H} + \theta'\bar{H}') \cdot \mathbf{h}_1 - \theta \cdot \mathbf{h}_2 - \theta' \cdot \mathbf{h}_3 + \kappa \cdot \mathbf{h}_5$$

If $e(\mathbf{b}_1, U^*) \neq 1_{\mathbb{G}_t} \wedge e(u, U^*) \cdot e(v, V^*) \cdot \prod e(c_\lambda, S_\lambda^*) = 1_{\mathbb{G}_t}$, accept, else reject.

One can note that, as usual with Dual Pairing Vectors Spaces, some basis vectors are kept hidden to real-life players, as they will only be used in the security proofs: $(\mathbf{b}_2, \mathbf{b}_4)$, $(\mathbf{b}_3^*, \mathbf{b}_4^*)$, $(\mathbf{d}_4, \mathbf{d}_6, \mathbf{d}_7, \mathbf{d}_8, \mathbf{d}_9, \mathbf{d}_{10})$, $(\mathbf{d}_5^*, \mathbf{d}_6^*, \mathbf{d}_7^*, \mathbf{d}_8^*, \mathbf{d}_9^*, \mathbf{d}_{10}^*)$, $(\mathbf{h}_4, \mathbf{h}_6, \mathbf{h}_7, \mathbf{h}_8)$, and $(\mathbf{h}_5^*, \mathbf{h}_6^*, \mathbf{h}_7^*, \mathbf{h}_8^*)$.

Correctness To clarify the content of the vectors, we detail each component in the corresponding bases $(\mathbb{B}, \mathbb{B}^*)$, $(\mathbb{D}, \mathbb{D}^*)$, and $(\mathbb{H}, \mathbb{H}^*)$, where 0^k denotes k zero components:

$\mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}, \Gamma)$**.** For random scalars $\delta_{\mathsf{id}}, \phi_0, \psi_1, \psi_2, (\phi_t)_t, (\pi_t)_t \xleftarrow{\$} \mathbb{Z}_q^*$ for $t \in \Gamma$.

$$\mathbf{k}_0^* = (\delta_{\mathsf{id}}, \phi_0, 0^2)_{\mathbb{B}^*} \qquad \mathbf{k}_t^* = (\delta_{\mathsf{id}}, \pi_t(1, t), \phi_t, 0^6)_{\mathbb{D}^*}$$
$$\mathbf{r}_1^* = (\delta_{\mathsf{id}}, 0, 0, \psi_1, 0^4)_{\mathbb{H}^*} \qquad \mathbf{r}_2^* = (0, \delta_{\mathsf{id}}, 0, \psi_2, 0^4)_{\mathbb{H}^*} \qquad \mathbf{r}_3^* = (0, 0, \delta_{\mathsf{id}}, \psi_3, 0^4)_{\mathbb{H}^*}$$

$\mathsf{Sig}(\mathsf{SK}_{\mathsf{id},\Gamma}, m, \mathcal{T})$**.** For random scalars $\delta_{\mathsf{id}}, \nu, \xi, \zeta, \psi_1, \psi_2, \psi_3, (q_\lambda)_\lambda, (\pi_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q^*$, and $(\alpha_\lambda)_\lambda$ a 1-labeling of $\mathcal{T}^*$, and $(\beta_\lambda)_\lambda$ a random 0-labeling of $\mathcal{T}^*$, for $H = \mathcal{H}(\mathcal{T}), H' = \mathcal{H}'(m)$:

$$U^* = (\xi\delta_{\mathsf{id}}, \xi\phi_0 + \zeta, 0^2)_{\mathbb{B}^*} \qquad S_\lambda^* = (\alpha_\lambda\xi\delta_{\mathsf{id}} + \beta_\lambda, (\alpha_\lambda\xi\pi_\lambda + \omega_\lambda)(1, t_\lambda), \alpha_\lambda\xi\phi_{t_\lambda} + q_\lambda, 0^6)_{\mathbb{D}^*}$$
$$V^* = (\xi\delta_{\mathsf{id}} \cdot (1, H, H'), \xi(\psi_1 + \psi_2 H + \psi_3 H') + \nu, 0^4)_{\mathbb{H}^*}$$

which follows the same distribution as

$$U^* = (\delta', \zeta, 0^2)_{\mathbb{B}^*} \qquad S_\lambda^* = (\alpha_\lambda\delta' + \beta_\lambda, \omega_\lambda(1, t_\lambda), q_\lambda, 0^6)_{\mathbb{D}^*}$$
$$V^* = (\delta' \cdot (1, H, H'), \nu, 0^4)_{\mathbb{H}^*}$$

for random scalars $\delta', \zeta, \nu, (q_\lambda)_\lambda, (\omega_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q^*$, and still $(\alpha_\lambda)_\lambda$ a 1-labeling of $\mathcal{T}^*$ and $(\beta_\lambda)_\lambda$ a random 0-labeling of $\mathcal{T}^*$.

$\mathsf{Verif}(\mathsf{PK}, m, \mathcal{T}, \sigma)$. For random scalars $\kappa, \kappa_0, (\kappa_\lambda)_\lambda, s, s_0, \theta, \theta', (\theta_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q$, and a random s_0-labeling $(s_\lambda)_\lambda$ of $\mathcal{T}$, for $\bar{H} = \mathcal{H}(\mathcal{T}), \bar{H}' = \mathcal{H}'(m)$:

$$u = (-s_0 - s, 0, \kappa_0, 0^1)_{\mathbb{B}} \qquad c_\lambda = (s_\lambda, \theta_\lambda(t_\lambda, -1), 0, \kappa_\lambda, 0^5)_{\mathbb{D}}$$
$$v = (s + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, 0^3)_{\mathbb{H}}$$

Let $(U^*, V^*, (S^*_\lambda)_\lambda)$ be a signature generated by Sig for an access-tree $\mathcal{T}$, with a key $\mathsf{SK}_{\mathsf{id},\Gamma}$ for attributes Γ so that $\mathcal{T}(\Gamma) = 1$, and $(u, v, (c_\lambda)_\lambda)$ the verification vectors generated by Verif for the same access-tree. We note $\mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma)$ the Evaluation Pruned Tree used during signature.

We remind from Proposition 1 that, since $(s_\lambda)_\lambda$ is a s_0-labeling of $\mathcal{T}$ and $(\alpha_\lambda)_\lambda$ (respectively $(\beta_\lambda)_\lambda$ are a 1-labeling (respectively 0-labeling) of $\mathcal{T}^*$, then $\sum_{\lambda\in\mathcal{L}} s_\lambda(\alpha_\lambda + \beta_\lambda) = s_0(1+0) = s_0$. We deduce that $\sum_{\lambda\in\mathcal{L}} s_\lambda(\alpha_\lambda\delta' + \beta_\lambda) = s_0(\delta' + 0) = s_0\delta'$.

The first check $e(\mathbf{b}_1, U^*) = g_t^{\delta'} \neq 1_{\mathbb{G}_t}$ is to make sure $\delta' \neq 0$, and thus that $\xi \neq 0$ during the signing process. For the second verification:

$$\prod e(c_\lambda, S^*_\lambda) = g_t^{\sum_{\lambda\in\mathcal{L}} s_\lambda(\alpha_\lambda\delta'+\beta_\lambda)} = g_t^{s_0\delta'} \tag{5.1}$$
$$e(u, U^*) \cdot e(v, V^*) = g_t^{\delta'\cdot(-s_0-s)} \cdot g_t^{\delta' s} = g_t^{-\delta' s_0} \tag{5.2}$$

This leads to an accept if the signature was properly generated, first by using the same attribute t_λ in the S^*_λ's of the signature and in the c_λ's of the ciphertext, such that the vectors $(1, t_\lambda)$ and $(t_\lambda, -1)$ are orthogonal in equation (5.1). Secondly, the commitment to the message and the access-tree must be the same: $(H, H') = (\bar{H}, \bar{H})$ so that $(1, H)$ and $(\bar{H}, -1)$, as well as $(1, H')$ and $(\bar{H}', -1)$ are orthogonal, to guarantee that for random θ and θ', $(1, H, H')$ and $(\theta\bar{H} + \theta'\bar{H}', -\theta, -\theta')$ are orthogonal in equation (5.2).

5.3.3 Security Analysis of the ABS

About the above ABS scheme described in Section 5.3.2, one can claim the unforgeability and the perfect anonymity.

Theorem 10 (Existential Unforgeability) *The ABS scheme described in Section 5.3.2 is existentially unforgeable under the collision-resistance of the hash functions $\mathcal{H}, \mathcal{H}'$ and the SXDH assumption, according to the Definition 6.*

Theorem 11 (Perfect Anonymity) *The ABS scheme described in Section 5.3.2 is perfectly anonymous, according to the Definition 7.*

Security Proofs We start by the anonymity result, as this will allow to perfectly simulate signing queries in the proof of unforgeability.

Perfect Anonymity. Let us define an alternative signing algorithm AltSig, that uses the master secret key instead of an individual signing key:

$\mathsf{AltSig}(\mathsf{MK}, m, \mathcal{T})$. With random scalars $\delta', \zeta, \nu, (q_\lambda)_\lambda, (\gamma_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q$, and $(\beta'_\lambda)_\lambda$ a random δ'-labeling of $\mathcal{T}^*$, set, for $H = \mathcal{H}(\mathcal{T})$ and $H' = \mathcal{H}'(m)$:

$$U^* = (\delta', \zeta, 0^2)_{\mathbb{B}^*} \qquad S^*_\lambda = (\beta'_\lambda, \gamma_\lambda(1, t_\lambda), q_\lambda, 0^6)_{\mathbb{D}^*} \qquad V^* = (\delta' \cdot (1, H, H'), \nu, 0^4)_{\mathbb{H}^*}$$

As shown, this is the same distribution as a real signature generated by a signing key $\mathsf{SK}_{\mathsf{id},\Gamma}$, except for two elements. First, the random $(\gamma_\lambda)_\lambda$ from the second component of S^*_λ follows the same random uniform distribution as $(\alpha_\lambda\xi\pi_\lambda + \omega_\lambda)_\lambda$. Then, the random δ'-labeling $(\beta'_\lambda)_\lambda$ of $\mathcal{T}^*$ replaces $(\alpha_\lambda\delta' + \beta_\lambda)_\lambda$, where (α_λ) is the 1-labeling of $\mathcal{T}^*$ associated to the Evaluation Pruned Tree specific to Γ, and (β_λ) a random 0-labeling of $\mathcal{T}^*$. As already noted, from the linearity of the labelings, the linear combination is a random $1\cdot\delta' + 0$-labeling of $\mathcal{T}^*$, as $(\beta'_\lambda)_\lambda$ is, which thus makes no difference.

Existential Unforgeability. For this proof, thanks to the above (perfect) indistinguishability of the Sig and AltSig outputs, we will first replace the simulation of the signing oracle by the AltSig procedure.

Then, we will use the index id for all the KeyGen queries/answers, and we assume the number of KeyGen queries bounded by K. We use the index i for all the Sig queries/answers, and we assume the number of Sig queries bounded by S. We will also use t to denote the attributes, and we assume the number of attributes involved in a security game bounded by T.

The verification done by the challenger (on the candidate forgery output by the adversary) uses a pair $(m, \mathcal{T})$ that is different from any pair that appeared in the signing queries, hence with $\bar{H} = \mathcal{H}(\mathcal{T})$ and $\bar{H}' = \mathcal{H}'(m)$, but $(\bar{H}, \bar{H}') \neq (H_i, H'_i)$ for all i, under the collision-resistance of $\mathcal{H}$ and $\mathcal{H}'$, as for any pair $(m_i, \mathcal{T}_i)$ at least $m \neq m_i$ or $\mathcal{T} \neq \mathcal{T}_i$. Then, the proof follows the sequence of games presented on Figure 5.7, to show that

$$\begin{aligned}\mathsf{Adv}_0 - \mathsf{Adv}_5 \leq (6KT + 2S + 2) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (4T^2K + 6K + 4S) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)\\ + S/q + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}}(t) + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}'}(t).\end{aligned}$$

Let us suppose we are at the final game $\mathbf{G}_5$ and consider a signature $(U^*, V^*, (S^*_\lambda)_\lambda)$ generated by the adversary. At that point, the Verif algorithm generates the element u in base $\mathbb{B}$ as :

$$u = (s', 0, \kappa_0, r'_0)$$

First, by definition of Verif, if $e(\mathbf{b}_1, U^*) = 1_{\mathbb{G}_t}$, then the verification fails. Hence, the first component of U^* must be non-zero, in the basis $\mathbb{B}^*$. We can now consider the value $e(u, U^*) \cdot e(v, V^*) \cdot \prod e(c_\lambda, S^*_\lambda)$. Since the coefficient s' of $\mathbf{b}_1$ in u is uniform and independent from all other values, then $e(u, U^*)$ is uniform and independent from all other pairings in the Verif algorithm. This implies $e(u, U^*) \cdot e(v, V^*) \cdot \prod e(c_\lambda, S^*_\lambda) \neq 1_{\mathbb{G}_t}$ except with probability $1/q$: $\mathsf{Adv}_5 \leq 1/q$. As a consequence,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{euf}} = \mathsf{Adv}_0 \leq (6KT + 2S + 2) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (4T^2K + 6K + 4S) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)\\ + (S+1)/q + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}}(t) + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}'}(t).\end{aligned}$$

Detailled Proof of the Existential Unforgeability of the ABS We now delve into the details of the proof. The security proof follows the sequence of games presented on Figure 5.7.

Game $\mathbf{G}_0$: From the correctness of the signature and the perfect anonymity, keys generated by the KeyGen algorithm, for user id, follow the distribution:

$$\begin{aligned}\mathbf{k}^*_{\mathsf{id},0} &= (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0^2)_{\mathbb{B}^*} & \mathbf{k}^*_{\mathsf{id},t} &= (\delta_{\mathsf{id}}, \pi_{\mathsf{id},t}(1,t), \phi_{\mathsf{id},t}, 0^6)_{\mathbb{D}^*} & \forall t\\ \mathbf{r}^*_{\mathsf{id},1} &= (\delta_{\mathsf{id}}, 0, 0, \psi_{\mathsf{id},1}, 0^4)_{\mathbb{H}^*} & \mathbf{r}^*_{\mathsf{id},2} &= (0, \delta_{\mathsf{id}}, 0, \psi_{\mathsf{id},2}, 0^4)_{\mathbb{H}^*}\\ & & \mathbf{r}^*_{\mathsf{id},3} &= (0, 0, \delta_{\mathsf{id}}, \psi_{\mathsf{id},3}, 0^4)_{\mathbb{H}^*}\end{aligned}$$

and the i-th signature generated by the Sig algorithm follows

$$\begin{aligned}U^*_i &= (\delta_i, \zeta_i, 0^2)_{\mathbb{B}^*} & S^*_{i,\lambda} &= (\beta'_{i,\lambda}, \gamma_{i,\lambda}(1, t_\lambda), q_{i,\lambda}, 0^6)_{\mathbb{D}^*}\\ & & V^*_i &= (\delta_i(1, H_i, H'_i), \nu_i, 0^4)_{\mathbb{H}^*}\end{aligned}$$

where $H_i = \mathcal{H}(\mathcal{T}_i), H'_i = \mathcal{H}'(m_i)$.

For the decision of validity of the forgery $S = (U^*, V^*, (S^*_\lambda)_\lambda)$ on message m' and policy $\mathcal{T}'$, one uses

$$\begin{aligned}u &= (-s_0 - s, 0, \kappa_0, 0)_{\mathbb{B}} & v &= (s + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, 0, 0, 0)_{\mathbb{H}}\\ & & c_\lambda &= (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, 0, 0, 0, 0, 0)_{\mathbb{D}}\end{aligned}$$

$\mathbf{G}_0$ Initialization of the EUF security game
For the (at most) K different id's and S different indices i's.

$$
\begin{array}{rcccccccccccc}
\mathbf{k}^*_{\mathsf{id},0} & = (& \delta_{\mathsf{id}} & \phi_{\mathsf{id},0} & 0 & 0 &)_{\mathbb{B}^*} & & & & & & \\
U^*_i & = (& \delta_i & \zeta_i & 0 & 0 &)_{\mathbb{B}^*} & & & & & & \\
u & = (& -s_0 - s & 0 & \kappa_0 & 0 &)_{\mathbb{B}} & & & & & & \\
\hline
\mathbf{r}^*_{\mathsf{id},1} & = (& \delta_{\mathsf{id}} & 0 & 0 & \psi_{\mathsf{id},1} & 0 & 0 & 0 & 0 &)_{\mathbb{H}^*} & & \\
\mathbf{r}^*_{\mathsf{id},2} & = (& 0 & \delta_{\mathsf{id}} & 0 & \psi_{\mathsf{id},2} & 0 & 0 & 0 & 0 &)_{\mathbb{H}^*} & & \\
\mathbf{r}^*_{\mathsf{id},3} & = (& 0 & 0 & \delta_{\mathsf{id}} & \psi_{\mathsf{id},3} & 0 & 0 & 0 & 0 &)_{\mathbb{H}^*} & & \\
V^*_i & = (& \delta_i & \delta_i H_i & \delta_i H'_i & \nu_i & 0 & 0 & 0 & 0 &)_{\mathbb{H}^*} & & \\
v & = (& s+\theta\bar{H}+\theta'\bar{H}' & -\theta & -\theta' & 0 & \kappa & 0 & 0 & 0 &)_{\mathbb{H}} & & \\
\hline
\mathbf{k}^*_{\mathsf{id},t} & = (& \delta_{\mathsf{id}} & \pi_{\mathsf{id},t} & \pi_{\mathsf{id},t}t & \phi_{\mathsf{id},t} & 0 & 0 & 0 & 0 & 0 & 0 &)_{\mathbb{D}^*} \\
S^*_{i,\lambda} & = (& \beta'_{i,\lambda} & \gamma_{i,\lambda} & \gamma_{i,\lambda}t_\lambda & q_{i,\lambda} & 0 & 0 & 0 & 0 & 0 & 0 &)_{\mathbb{D}^*} \\
c_\lambda & = (& s_\lambda & \theta_\lambda t_\lambda, & -\theta_\lambda & 0 & \kappa_\lambda & 0 & 0 & 0 & 0 & 0 &)_{\mathbb{D}}
\end{array}
$$

$\mathbf{G}_1$ r_λ is a r_0-labeling, ω is random
SubSpace-Ind in $(\mathbb{B}, \mathbb{B}^*), (\mathbb{D}, \mathbb{D}^*)$ and $(\mathbb{H}, \mathbb{H}^*)$

$$
\begin{array}{rcccccccccc}
u & = (& -s_0 - s & 0 & \kappa_0 & -r_0 &)_{\mathbb{B}} & & & & & \\
v & = (& s+\theta\bar{H}+\theta'\bar{H}' & -\theta & -\theta' & 0 & \kappa & \omega & 0 & 0 &)_{\mathbb{H}} \\
c_\lambda & = (& s_\lambda & \theta_\lambda t_\lambda & -\theta_\lambda & 0 & \kappa_\lambda & r_\lambda & 0 & 0 & 0 & 0 &)_{\mathbb{D}}
\end{array}
$$

$\mathbf{G}_2$ δ''_{id} all random: Hybrid sub-sequence (see Figure 5.8)

$$
\begin{array}{rcccccccccccc}
\mathbf{k}^*_{\mathsf{id},0} & = (& \delta_{\mathsf{id}} & \phi_{\mathsf{id},0} & 0 & \delta''_{\mathsf{id}} &)_{\mathbb{B}^*} & & & & & & \\
\mathbf{k}^*_{\mathsf{id},t} & = (& \delta_{\mathsf{id}} & \pi_{\mathsf{id},t} & \pi_{\mathsf{id},t}t & \phi_{\mathsf{id},t} & 0 & 0 & 0 & 0 & 0 & 0 &)_{\mathbb{D}^*}
\end{array}
$$

$\mathbf{G}_3$ r'_0 random: Formal change of basis

$$
\begin{array}{rcccccc}
u & = (& -s_0 - s & 0 & \kappa_0 & r'_0 &)_{\mathbb{B}}
\end{array}
$$

$\mathbf{G}_4$ ρ_i, τ_i all random: Hybrid sub-sequence (see Figure 5.9)

$$
\begin{array}{rcccccccccc}
U^*_i & = (& \delta_i & \zeta_i & 0 & \rho_i &)_{\mathbb{B}^*} & & & & \\
V^*_i & = (& \delta_i & \delta_i H_i & \delta_i H'_i & \nu_i & 0 & \tau_i & 0 & 0 &)_{\mathbb{H}}
\end{array}
$$

$\mathbf{G}_5$ s' random

$$
\begin{array}{rcccccc}
u & = (& s' & 0 & \kappa_0 & r'_0 &)_{\mathbb{B}}
\end{array}
$$

Figure 5.7: Sequence of Games for Unforgeability. Grey rectangles indicate the values changed in each game.

where $\bar{H} = \mathcal{H}(\mathcal{T}), \bar{H}' = \mathcal{H}'(m)$, and $(\mathcal{T}, m) \neq (\mathcal{T}_i, m_i)$ for all i. Instead of outputting just the decision, one can consider the challenger outputs $(u, v, (c_\lambda)_\lambda)$, and everybody can make the final verification:

$$e(\mathbf{b}_1, U^*) \neq 1_{\mathbb{G}_t} \qquad e(u, U^*) \cdot e(v, V^*) \cdot \prod e(c_\lambda, S^*_\lambda) = 1_{\mathbb{G}_t}$$

And we denote by Adv_0 the probability of the validity of the forgery. Our goal is to show this is negligible.

Game $\mathbf{G}_1$: We change the verification vectors into

$$u = (-s_0 - s, 0, \kappa_0, -r_0)_{\mathbb{B}} \qquad v = (s + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, \omega, 0, 0)_{\mathbb{H}}$$
$$c_\lambda = (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, r_\lambda, 0, 0, 0, 0)_{\mathbb{D}}$$

where r_0 and ω are random scalars and $(r_\lambda)_\lambda$ is a random r_0-labeling for the tree-policy $\mathcal{T}'$.

The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_1$: one applies the $\mathsf{SubSpace\text{-}Ind}$ property on $(\mathbb{B}, \mathbb{B}^*)_{3,4}, (\mathbb{D}, \mathbb{D}^*)_{4,5}$ and $(\mathbb{H}, \mathbb{H}^*)_{4,5}$. Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \tau \bmod q$ with either $\tau = 0$ or $\tau = 1$, which are indistinguishable under the DDH assumption in $\mathbb{G}_1$.

Let us assume we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$, $(\mathbb{V}, \mathbb{V}^*)$ and $(\mathbb{W}, \mathbb{W}^*)$. Then we define the matrices

$$B = \begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix}_{3,4} \quad B' = \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix}_{3,4} \quad D = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{5,6} \quad D' = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{5,6}$$
$$\mathbb{B} = B \cdot \mathbb{U} \quad \mathbb{B}^* = B' \cdot \mathbb{U}^* \quad \mathbb{D} = D \cdot \mathbb{V} \quad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$
$$H = \begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix}_{4,5} \quad H' = \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix}_{4,5}$$
$$\mathbb{H} = H \cdot \mathbb{W} \quad \mathbb{H}^* = H' \cdot \mathbb{W}^*$$

The vectors $\mathbf{b}^*_4$, $\mathbf{d}^*_5$, and $\mathbf{h}^*_5$ can not be computed, but they are hidden from the adversary's view, and are not used in any vector. We compute the new vectors:

$$\begin{aligned}
u &= (-s_0 - s, 0, \kappa_0, 0)_{\mathbb{B}} - (0, 0, br_0, -cr_0)_{\mathbb{U}} \\
&= (-s_0 - s, 0, \kappa_0, 0)_{\mathbb{B}} - (0, 0, b_1 r_0, -(c - ab)r_0)_{\mathbb{B}} \\
&= (-s_0 - s, 0, \kappa_0 + br_0, -\tau r_0)_{\mathbb{B}} \\
v &= (s + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, 0, 0, 0)_{\mathbb{H}} + (0, 0, 0, 0, -b\omega, c\omega, 0, 0)_{\mathbb{W}} \\
&= (s + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, 0, 0, 0)_{\mathbb{H}} + (0, 0, 0, 0, -b\omega, (c - ab)\omega, 0, 0)_{\mathbb{H}} \\
&= (s + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa - b\omega, \tau\omega, 0, 0)_{\mathbb{H}} \\
c_\lambda &= (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, 0, 0, 0, 0, 0)_{\mathbb{D}} + (0, 0, 0, 0, -br_\lambda, cr_\lambda, 0, 0, 0, 0)_{\mathbb{V}} \\
&= (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, 0, 0, 0, 0, 0)_{\mathbb{D}} + (0, 0, 0, 0, -br_\lambda, (c - ab)r_\lambda, 0, 0, 0, 0)_{\mathbb{D}} \\
&= (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda - br_\lambda, \tau r_\lambda, 0, 0, 0, 0)_{\mathbb{D}}
\end{aligned}$$

One can easily note that when $\tau = 0$, this is the previous game, and when $\tau = 1$, we are in the new game.

On the other side, keys and signatures are unchanged, as their values on the corresponding unknown basis vectors are 0. They can thus be directly defined in $\mathbb{B}^*$, $\mathbb{D}^*$, and $\mathbb{H}^*$. We thus have $\mathsf{Adv}_0 - \mathsf{Adv}_1 \leq 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

Game $\mathbf{G}_2$: We introduce a random value δ''_{id} in every key in basis $\mathbb{B}$, in the component corresponding to the random value r_0 that was just introduced in the verification vector u. In order to do this, we proceed with an hybrid game on the key queries, modifying them one id at a time. We will denote the current key by Δ, and we update the Δ-th key as:

$$\mathbf{k}^*_{\Delta,0} = (\delta_\Delta, \phi_{\mathsf{id},0}, 0, \delta''_\Delta)_{\mathbb{B}^*}$$

When $\Delta = 0$, no key has been modified, this is exactly the game $\mathbf{G}_1$: $\mathbf{G}_1 = \mathbf{G}_{1.0.0}$, whereas for $\Delta = K$, all the keys have been modified, this is exactly the expected game $\mathbf{G}_2$: $\mathbf{G}_2 = \mathbf{G}_{1.K.0}$. We show that for each Δ,

$$\mathsf{Adv}_{1.\Delta.0} - \mathsf{Adv}_{1.\Delta+1.0} \leq 6T \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (4T^2+6) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t).$$

Hence, globally, we have

$$\mathsf{Adv}_1 - \mathsf{Adv}_2 \leq 6KT \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (4T^2+6)K \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t).$$

Game $\mathbf{G}_3$: In this game, we replace r_0 in the verification vector by a random independent $r'_0 \xleftarrow{\$} \mathbb{Z}_q$:

$$u = (-s_0 - s, 0, \kappa_0, r'_0)_{\mathbb{B}}$$

To do this, we proceed with a formal change of basis $\mathbb{B}$. Let us assume we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$. Then we define the matrices, with random $\theta \xleftarrow{\$} \mathbb{Z}_q^*$

$$B = \begin{pmatrix} \theta \end{pmatrix}_4 \qquad B' = \begin{pmatrix} 1/\theta \end{pmatrix}_4$$
$$\mathbb{B} = B \cdot \mathbb{U} \qquad \mathbb{B}^* = B' \cdot \mathbb{U}^*$$

which modifies only the hidden basis vectors $\mathbf{b}_4, \mathbf{b}^*_4$. Since they are not in the adversary's view, the advantage is not modified: $\mathsf{Adv}_3 = \mathsf{Adv}_2$. Furthermore

$$\mathbf{k}^*_{\Delta,0} = (\delta_\Delta, \phi_{id,0}, 0, \delta'_\Delta)_{\mathbb{U}^*} = (\delta_\Delta, \phi_{id,0}, 0, \theta\delta''_\Delta)_{\mathbb{B}^*}$$
$$u = (-s_0 - s, 0, \kappa_0, r_0)_{\mathbb{U}} = (-s_0 - s, 0, \kappa_0, r_0/\theta)_{\mathbb{B}}$$

Which replaces the random value δ''_Δ by another random value $\theta\delta''_\Delta$ that follows the same uniform distribution, and $r'_0 = -r_0/\theta$ follows a uniform independent distribution, also independent from the r_0-labeling $(r_\lambda)_\lambda$.

Game $\mathbf{G}_4$: We now update generated signatures, with random values in coordinates corresponding to the random ω that was introduced in the verification vector v in game $\mathbf{G}_1$. In order to do this, we proceed with an hybrid game on the signature queries, modifying them one i at a time. We will denote the current signature by j, and we update the j-th signature as:

$$U^*_j = (\delta_j, \zeta_j, 0, \rho_j)_{\mathbb{B}^*}$$
$$V^*_j = (\delta_j(1, H_j, H'_j), \nu_j, 0, \tau_j, 0, 0)_{\mathbb{H}^*}$$

with $\rho_j, \tau_j \xleftarrow{\$} \mathbb{Z}_q$.

When $j = 0$, no signature has been modified, this is exactly the game $\mathbf{G}_3$: $\mathbf{G}_3 = \mathbf{G}_{3.0.0}$, whereas for $j = S$, all the signatures have been modified, this is exactly the expected game $\mathbf{G}_4$: $\mathbf{G}_4 = \mathbf{G}_{3.S.0}$. In Section 5.3.3, we show that for each j,

$$\mathsf{Adv}_{3.j.0} - \mathsf{Adv}_{3.j+1.0} \leq 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 1/q,$$

if $(\bar{H}, \bar{H}') \neq (H_j, H'_j)$, which holds under the collision resistance of the two hash functions. Hence, globally, we have

$$\mathsf{Adv}_3 - \mathsf{Adv}_4 \leq S \times (4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 1/q) + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}}(t) + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}'}(t).$$

Game $\mathbf{G}_5$: In this final game, we make the verification vector reject all the signatures by removing the original secret on the first position:

$$u = (s', 0, \kappa_0, r'_0)_{\mathbb{B}}$$

To do this, we define the matrices, with $\Theta \xleftarrow{\$} \mathbb{Z}_q$

$$B' = \begin{pmatrix} 1 & -\Theta \\ 0 & 1 \end{pmatrix}_{1,4} \qquad B = \begin{pmatrix} 1 & 0 \\ \Theta & 1 \end{pmatrix}_{1,4}$$
$$\mathbb{B}^* = B' \cdot \mathbb{U}^* \qquad \mathbb{B} = B \cdot \mathbb{U}$$

which modifies the hidden vectors $\mathbf{b}_4, \mathbf{b}_1^*$. Since they are not in the adversary's view, the advantage is not modified: $\mathsf{Adv}_5 = \mathsf{Adv}_4$. The verification vector is modified as

$$u = (-s_0 - s, 0, \kappa_0, r'_0)_{\mathbb{U}} = (-s_0 - s - \Theta r'_0, 0, \kappa_0, r'_0)_{\mathbb{B}} = (s', 0, \kappa_0, r'_0)_{\mathbb{B}}$$

with $s' := -s_0 - s - \Theta r'_0$ that is uniformly distributed. Meanwhile, the keys and signatures are modified as follows:

$$\mathbf{k}^*_{\mathsf{id},0} = (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, \delta''_{\mathsf{id}})_{\mathbb{U}^*} = (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, \delta''_{\mathsf{id}} + \Theta\delta_{\mathsf{id}})_{\mathbb{B}^*} = (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, \delta'_{\mathsf{id}})_{\mathbb{B}^*}$$
$$U^*_i = (\delta_i, \zeta_i, 0, \rho_i)_{\mathbb{U}^*} = (\delta_i, \zeta_i, 0, \rho_i + \Theta\delta_i)_{\mathbb{B}^*} = (\delta_i, \zeta_i, 0, \rho'_i)_{\mathbb{B}^*}$$

If we combine all the steps:

$$\begin{aligned}
\mathsf{Adv}_0 - \mathsf{Adv}_5 &\leq 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) \\
&\quad + 6KT \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (4T^2 + 6)K \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 0 \\
&\quad + S \times (4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 1/q) + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}}(t) + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}'}(t) \\
&\leq (2S + 2 + 6KT) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + (4T^2K + 6K + 4S) \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) \\
&\quad + S/q + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}}(t) + \mathsf{Adv}^{\mathsf{coll}}_{\mathcal{H}'}(t)
\end{aligned}$$

Existential Unforgeability: Gap between $\mathbf{G}_1$ and $\mathbf{G}_2$. In this sequence of games, detailed on Figure 5.8, we will stop tracking elements on bases $\mathbb{H}, \mathbb{H}^*$ as they are not modified.

Game $\mathbf{G}_{1.\Delta.0}$: The state of the game at that point is the following, for the keys

$$\begin{aligned}
&\mathsf{id} < \Delta && \mathbf{k}^*_{\mathsf{id},0} = (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, \delta''_{\mathsf{id}})_{\mathbb{B}^*} \\
&\mathsf{id} \geq \Delta && \mathbf{k}^*_{\mathsf{id},0} = (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, 0)_{\mathbb{B}^*} \\
&\forall \mathsf{id}, \forall t && \mathbf{k}^*_{\mathsf{id},t} = (\delta_{\mathsf{id}}, \pi_{\mathsf{id},t}(1,t), \phi_{\mathsf{id},t}, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}
\end{aligned}$$

for the signatures,

$$U^*_i = (\delta_i, \zeta_i, 0, 0)_{\mathbb{B}^*} \qquad S^*_{i,\lambda} = (\beta'_{i,\lambda}, \gamma_{i,\lambda}(1, t_\lambda), q_{i,\lambda}, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}$$

and the verification vectors

$$u = (-s_0 - s, 0, \kappa_0, -r_0)_{\mathbb{B}} \qquad c_\lambda = (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, r_\lambda, 0, 0, 0, 0)_{\mathbb{D}}$$

Game $\mathbf{G}_{1.\Delta.1}$: We change the Δ-th key into, for a random $\delta'_\Delta \xleftarrow{\$} \mathbb{Z}_q$

$$\mathbf{k}^*_{\Delta,0} = (\delta_\Delta, \phi_{\Delta,0}, 0, \delta'_\Delta)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\Delta,t} = (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, 0, 0, 0, 0)_{\mathbb{D}^*}$$

Other vectors are not modified.

$\mathbf{G}_{1.\Delta.0}$ Hybrid sequence from $\mathbf{G}_1$ to $\mathbf{G}_2$

$\mathsf{id} \geq \Delta$	$\mathbf{k}^*_{\mathsf{id},0}$	$=($	δ_{id}	$\phi_{\mathsf{id},0}$	0	0	$)_{\mathbb{B}^*}$					
$\mathsf{id} < \Delta$	$\mathbf{k}^*_{\mathsf{id},0}$	$=($	δ_{id}	$\phi_{\mathsf{id},0}$	0	δ''_{id}	$)_{\mathbb{B}^*}$					
	U^*_i	$=($	δ_i	ζ_i	0	0	$)_{\mathbb{B}^*}$					
	u	$=($	$-s_0 - s$	0	κ_0	$-r_0$	$)_{\mathbb{B}}$					
	$\mathbf{r}^*_{\mathsf{id},1}$	$=($	δ_{id}	0	0	$\psi_{\mathsf{id},1}$	0	0	0	0	$)_{\mathbb{H}^*}$	
	$\mathbf{r}^*_{\mathsf{id},2}$	$=($	0	δ_{id}	0	$\psi_{\mathsf{id},2}$	0	0	0	0	$)_{\mathbb{H}^*}$	
	$\mathbf{r}^*_{\mathsf{id},3}$	$=($	0	0	δ_{id}	$\psi_{\mathsf{id},3}$	0	0	0	0	$)_{\mathbb{H}^*}$	
	V^*_i	$=($	δ_i	$\delta_i H_i$	$\delta_i H'_i$	ν_{id}	0	0	0	0	$)_{\mathbb{H}^*}$	
	v	$=($	$s+\theta\bar{H}+\theta\bar{H}'$	$-\theta$	$-\theta'$	0	κ	ω	0	0	$)_{\mathbb{H}}$	
	$\mathbf{k}^*_{\mathsf{id},t}$	$=($	δ_{id}	$\pi_{\mathsf{id},t}$	$\pi_{\mathsf{id},t}t$	$\phi_{\mathsf{id},t}$	0	0	0	0	0	0 $)_{\mathbb{D}^*}$
	$S^*_{i,\lambda}$	$=($	$\beta'_{i,\lambda}$	$\gamma_{i,\lambda}$	$\gamma_{i,\lambda}t_\lambda$	$q_{i,\lambda}$	0	0	0	0	0	0 $)_{\mathbb{D}^*}$
	c_λ	$=($	s_λ	$\theta_\lambda t_\lambda$	$-\theta_\lambda$	0	κ_λ	r_λ	0	0	0	0 $)_{\mathbb{D}}$

$\mathbf{G}_{1.\Delta.1}$ Add random δ'_Δ: SubSpace-Ind in $(\mathbb{B}^*, \mathbb{B})$ and in $(\mathbb{D}^*, \mathbb{D})$

	$\mathbf{k}^*_{\Delta,0}$	$=($	δ_Δ	$\phi_{\Delta,0}$	0	δ'_Δ	$)_{\mathbb{B}^*}$					
	$\mathbf{k}^*_{\Delta,t}$	$=($	δ_Δ	$\pi_{\Delta,t}$	$\pi_{\Delta,t}t$	$\phi_{\Delta,t}$	0	δ'_Δ	0	0	0	0 $)_{\mathbb{D}^*}$

$\mathbf{G}_{1.\Delta.2}$ Add random $\delta'_\Delta z_t$: SubSpace-Ind in $(\mathbb{D}^*, \mathbb{D})$

	$\mathbf{k}^*_{\Delta,t}$	$=($	δ_Δ	π_Δ	$\pi_\Delta t$	$\phi_{\Delta,t}$	0	δ'_Δ	0	$\delta'_\Delta z_t$	0	0 $)_{\mathbb{D}^*}$

$\mathbf{G}_{1.\Delta.3}$ Hybrid game (see Figure 5.10)

	c_λ	$=($	s_λ	$\theta_\lambda t_\lambda$	$-\theta_\lambda$	0	κ_λ	0	0	$\frac{r_\lambda}{z_{t_\lambda}}$	0	0 $)_{\mathbb{D}}$

$\mathbf{G}_{1.\Delta.4}$ Policy argument: r_0 unpredictable, then δ''_Δ random

	$\mathbf{k}^*_{\Delta,0}$	$=($	δ_Δ	$\phi_{\Delta,0}$	0	δ''_Δ	$)_{\mathbb{B}^*}$					

$\mathbf{G}_{1.\Delta.5}$ Undo $\mathbf{G}_{1.\Delta.3}$

	c_λ	$=($	s_λ	$\theta_\lambda t_\lambda$	$-\theta_\lambda$	0	κ_λ	r_λ	0	0	0	0 $)_{\mathbb{D}}$

$\mathbf{G}_{1.\Delta.6}$ Undo $\mathbf{G}_{1.\Delta.2}$

	$\mathbf{k}^*_{\Delta,t}$	$=($	δ_Δ	$\pi_{\Delta,t}$	$\pi_{\Delta,t}t$	$\phi_{\Delta,t}$	0	δ'_Δ	0	0	0	0 $)_{\mathbb{D}^*}$

$\mathbf{G}_{1.\Delta.7}$ Undo $\mathbf{G}_{1.\Delta.1}$

	$\mathbf{k}^*_{\Delta,t}$	$=($	δ_Δ	$\pi_{\Delta,t}$	$\pi_{\Delta,t}t$	$\phi_{\Delta,t}$	0	0	0	0	0	0 $)_{\mathbb{D}^*}$

Figure 5.8: Existential Unforgeability: Gap between $\mathbf{G}_1$ and $\mathbf{G}_2$

The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_2$: one applies the SubSpace-Ind property, on $(\mathbb{B}^*, \mathbb{B})_{2,4}$ and $(\mathbb{D}^*, \mathbb{D})_{1,6}$. Indeed, we can consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \tau \bmod q$ with either $\tau = 0$ or $\tau = \delta'_\Delta \xleftarrow{\$} \mathbb{Z}_q$, which are indistinguishable under the DDH assumption in $\mathbb{G}_2$.

Let us assume we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$. Then we define the matrices

$$B' = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{1,4} \quad B = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{1,4} \quad D' = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{1,6} \quad D = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{1,6}$$
$$\mathbb{B}^* = B' \cdot \mathbb{U}^* \quad \mathbb{B} = B \cdot \mathbb{U} \quad \mathbb{D}^* = D' \cdot \mathbb{V}^* \quad \mathbb{D} = D \cdot \mathbb{V}$$

The vectors $\mathbf{b}_4$ and $\mathbf{d}_6$ can not be computed, but they are hidden from the adversary's view. The Δ-th key is now computed as

$$\begin{aligned} \mathbf{k}^*_{\Delta,0} &= (\delta_\Delta, \phi_{\Delta,0}, 0, 0)_{\mathbb{B}^*} + (b, 0, 0, c)_{\mathbb{U}^*} = (\delta_\Delta + b, \phi_{\Delta,0}, 0, \tau)_{\mathbb{B}^*} \\ \mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_\Delta(1,t), \phi_{\Delta,t}, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*} + (b, 0, 0, 0, 0, c, 0, 0, 0, 0)_{\mathbb{V}^*} \\ &= (\delta_\Delta + b, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \tau, 0, 0, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

When $\tau = 0$, we are in the previous game, meanwhile when $\tau = \delta'_\Delta \xleftarrow{\$} \mathbb{Z}_q$ we are in the new game. In both cases, the random value δ_Δ are replaced by $\delta_\Delta + b$ that follow the same distribution. We can also update $\mathbf{r}^*_{\Delta,1}, \mathbf{r}^*_{\Delta,2}$ and $\mathbf{r}^*_{\Delta,3}$ as $b \cdot G_2$ is given.

Since $\mathbf{b}_4$ and $\mathbf{d}_6$ cannot be computed, one has to generate the verification vectors in the original bases:

$$\begin{aligned} u &= (-s_0 - s, 0, \kappa_0, -r_0)_{\mathbb{U}} \\ &= (-s_0 - ar_0 - s, 0, \kappa_0, -r_0)_{\mathbb{B}} = (-s'_0 - s, 0, \kappa_0, -r_0)_{\mathbb{B}} \\ c_\lambda &= (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, r_\lambda, 0, 0, 0, 0)_{\mathbb{V}} \\ &= (s_\lambda + ar_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, r_\lambda, 0, 0, 0, 0)_{\mathbb{D}} = (s'_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, r_\lambda, 0, 0, 0, 0)_{\mathbb{D}} \end{aligned}$$

where $s'_0 = s_0 + ar_0$ and $s'_\lambda = s_\lambda + ar_\lambda$. Since $(r_\lambda)_\lambda$ and $(s_\lambda)_\lambda$ are random r_0 and s_0-labeling (respectively), then any linear combination $(s'_\lambda = s_\lambda + ar_\lambda)_\lambda$ is a random $s'_0 = s_0 + ar_0$-labeling.

Hence, $\mathsf{Adv}_{1.\Delta.0} - \mathsf{Adv}_{1.\Delta.1} \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{1.\Delta.2}$: We again change the keys into

$$\mathbf{k}^*_{\Delta,t} = (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, 0, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*}$$

for random scalars $z_t \xleftarrow{\$} \mathbb{Z}_q$.

The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_2$: one applies the SubSpace-Ind property on $(\mathbb{D}^*, \mathbb{D})_{4,8}$. Indeed, we can consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \tau \bmod q$ with either $\tau = 0$ or $\tau = 1$, which are indistinguishable under the DSDH assumption in $\mathbb{G}_2$.

One chooses additional scalars $\alpha_t = \delta'_\Delta z_t$ and $\beta_t \xleftarrow{\$} \mathbb{Z}_q$ to virtually set $b_t = \alpha_t \cdot b + \beta_t$ and $c_t = \alpha_t \cdot c + \beta_t \cdot a$, which makes $c_t - ab_t = \alpha_t \cdot \tau = \delta'_\Delta z_t \cdot \tau$.

Let us assume we start from random dual orthogonal bases $(\mathbb{V}, \mathbb{V}^*)$. Then we define the matrices

$$D = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{4,8} \quad D' = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{4,8}$$
$$\mathbb{D} = D \cdot \mathbb{V} \quad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

The vector $\mathbf{d}_7$ can not be computed, but it is hidden from the adversary's view. The Δ-th key is now computed as

$$\begin{aligned}\mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, 0, 0, 0, 0)_{\mathbb{D}^*} + (0,0,0,b_t,0,0,0,c_t,0,0)_{\mathbb{V}^*}\\ &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t} + b_t, 0, \delta'_\Delta, 0, \alpha_t\tau, 0, 0)_{\mathbb{D}^*}\\ &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi'_{\Delta,t}, 0, \delta'_\Delta, 0, \delta'_\Delta z_t\cdot\tau, 0, 0)_{\mathbb{D}^*}\end{aligned}$$

When $\tau = 0$, we are in the previous game, meanwhile when $\tau = 1$ we are in the new game.

Vectors in $\mathbb{D}$ are not modified, and are directly simulated in $\mathbb{D}$, as their components are 0 on the 7-th coordinate. Hence, $\mathsf{Adv}_{1.\Delta.1} - \mathsf{Adv}_{1.\Delta.2} \leq \mathsf{Adv}^{\mathsf{dsdh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{1.\Delta.3}$: We hide the shares r_λ in every verification vectors, in front of the value δ'_Δ that was just introduced in the keys. In order to do this, we proceed with an hybrid game on the attribute indices, modifying them one t at a time, using the first time the attribute t appears in the game. We will denote the current attribute by p, also identified to its index in the order of appearance. We transform the verification vectors for all λ such that $t_\lambda = p$ into

$$c_\lambda = (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, \kappa_\lambda, 0, 0, r_\lambda/z_p)_{\mathbb{D}}$$

When $p = 0$ (for the order of appearance, which means before the first one), this is exactly the game $\mathbf{G}_{1.\Delta.2}$: $\mathbf{G}_{1.\Delta.2} = \mathbf{G}_{1.\Delta.2.0.0}$, whereas for $p = T$ (for the order of appearance, which means the last one) this is exactly the expected game $\mathbf{G}_{1.\Delta.3}$: $\mathbf{G}_{1.\Delta.3} = \mathbf{G}_{1.\Delta.2.T.0}$.

From 5.3.3, for each p, we prove that

$$\mathsf{Adv}_{1.\Delta.2.p.0} - \mathsf{Adv}_{1.\Delta.2.p.5} \leq 3 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 2T \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t).$$

Hence, globally, we have

$$\mathsf{Adv}_{1.\Delta.2} - \mathsf{Adv}_{1.\Delta.3} \leq 3T \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 2T^2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t).$$

Game $\mathbf{G}_{1.\Delta.4}$: First, one can note that the scalars z_t used in the verification vectors and in the Δ-th key mask the r_λ in the verification vectors. Since attributes in the Δ-th key do not satisfy the policy $\mathcal{T}'$ of the forgery, not enough r_λ can be known (others are perfectly private), and then r_0 is perfectly unpredictable, it can be replaced by a random value r'_0, in an intermediate game.

Then, we proceed with a formal change of basis $\mathbb{B}$. Let us assume we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$. Then we define the matrices, with $\theta = r'_0/r_0$, for a random $r'_0 \xleftarrow{\$} \mathbb{Z}_q^*$

$$B = \begin{pmatrix} \theta & \\ & \end{pmatrix}_4 \qquad B' = \begin{pmatrix} 1/\theta & \\ & \end{pmatrix}_4$$
$$\mathbb{B} = B \cdot \mathbb{U} \qquad \mathbb{B}^* = B' \cdot \mathbb{U}^*$$

which modifies only the hidden basis vectors $\mathbf{b}_4, \mathbf{b}^*_4$. Since they are not in the adversary's view, the advantage is not modified: $\mathsf{Adv}_{1.\Delta.4} = \mathsf{Adv}_{1.\Delta.3}$. Then,

$$\begin{aligned}\mathsf{id} < \Delta \qquad \mathbf{k}^*_{\mathsf{id},0} &= (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, \delta''_{\mathsf{id}})_{\mathbb{U}^*} = (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, \theta\delta''_{\mathsf{id}})_{\mathbb{B}^*}\\ \mathbf{k}^*_{\Delta,0} &= (\delta_\Delta, \phi_{\Delta,0}, 0, \delta'_\Delta)_{\mathbb{U}^*} = (\delta_\Delta, \phi_{\Delta,0}, 0, \theta\delta'_\Delta)_{\mathbb{B}^*}\\ \mathsf{id} > \Delta \qquad \mathbf{k}^*_{\mathsf{id},0} &= (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, 0)_{\mathbb{U}^*} = (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, 0)_{\mathbb{B}^*}\\ u &= (-s_0 - s, 0, \kappa_0, -r'_0)_{\mathbb{U}} = (-s_0 - s, 0, \kappa_0, -r'_0/\theta)_{\mathbb{U}}\\ &= (-s_0 - s, 0, \kappa_0, -r_0)_{\mathbb{U}}\end{aligned}$$

Definitions in $(\mathbb{U}, \mathbb{U}^*)$ are the above intermediate game, and definitions in $(\mathbb{D}, \mathbb{D}^*)$ correspond to the new game as for $\mathsf{id} < \Delta$, δ''_{id} are already random values, then $\theta\delta''_{\mathsf{id}}$ is also uniformly random (whatever independent θ is); and since r'_0 is random, $\theta\delta'_\Delta = r'_0\delta'_\Delta/r_0$ is uniformly random, and independent.

Game $\mathbf{G}_{1.\Delta.5}$: In this game, we undo $\mathbf{G}_{1.\Delta.3}$. Then, as above, $\mathsf{Adv}_{1.\Delta.4} - \mathsf{Adv}_{1.\Delta.5} \leq 3T \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 2T^2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{1.\Delta.6}$: In this game, we undo $\mathbf{G}_{1.\Delta.2}$. Then $\mathsf{Adv}_{1.\Delta.5} - \mathsf{Adv}_{1.\Delta.6} \leq \mathsf{Adv}^{\mathsf{dsdh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{1.\Delta.7}$: In this game, we undo $\mathbf{G}_{1.\Delta.1}$. Then $\mathsf{Adv}_{1.\Delta.6} - \mathsf{Adv}_{1.\Delta.7} \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

The hybrid on Δ is over, as one can see: $G_{1.\Delta+1.0} = G_{1.\Delta.7}$. We can now proceed on the hybrid on $\Delta + 1$, until $\Delta = K$.

Existential Unforgeability: Gap between $\mathbf{G}_3$ and $\mathbf{G}_4$. In this sequence, we will stop tracking elements on bases $\mathbb{D}, \mathbb{D}^*$ as they are not modified. See Figure 5.9.

$\mathbf{G}_{3.j.0}$	Hybrid sequence from $\mathbf{G}_3$ to $\mathbf{G}_4$									
$i \geq j$	U_i^*	$= ($	$\xi_i\delta_i$	ζ_i	0	0	$)_{\mathbb{B}^*}$			
$i < j$	U_i^*	$= ($	$\xi_i\delta_i$	ζ_i	0	ρ_i	$)_{\mathbb{B}^*}$			
	u	$= ($	$-s_0 - s$	0	κ_0	$-r_0$	$)_{\mathbb{B}}$			
$i \geq j$	V_i^*	$= ($	$\xi_i\delta_i$	$\xi_i\delta_i H_i$	$\xi_i\delta_i H_i'$	ν_i	0	0	0	$)_{\mathbb{H}^*}$
$i < j$	V_i^*	$= ($	$\xi_i\delta_i$	$\xi_i\delta_i H_i$	$\xi_i\delta_i H_i'$	ν_i	0	τ_i	0	0 $)_{\mathbb{H}^*}$
	v	$= ($	$s+\theta\bar{H}+\theta'\bar{H}'$	$-\theta$	$-\theta'$	0	κ	ω	0	0 $)_{\mathbb{H}}$
$\mathbf{G}_{3.j.1}$	Add random ρ_j: SubSpace-Ind on (1,4) in $(\mathbb{B}^*, \mathbb{B})$ and on (1,6) in $(\mathbb{H}^*, \mathbb{H})$									
	U_j^*	$= ($	$\xi_j\delta_j$	ζ_j	0	ρ_j	$)_{\mathbb{B}^*}$			
	V_j^*	$= ($	$\xi_j\delta_j$	$\xi_j\delta_j H_j$	$\xi_j\delta_j H_j'$	ν_j	0	$\frac{\rho_j r_0}{\omega}$	0	0 $)_{\mathbb{H}^*}$
$\mathbf{G}_{3.j.2}$	Randomize $\rho_j r_0/\omega$: Index-Ind on (1,2,3,6,7,8) in $(\mathbb{H}^*, \mathbb{H})$									
	U_j^*	$= ($	$\xi_j\delta_j$	ζ_j	0	ρ_j	$)_{\mathbb{B}^*}$			
	V_j^*	$= ($	$\xi_j\delta_j$	$\xi_j\delta_j H_j$	$\xi_j\delta_j H_j'$	ν_j	0	τ_j	0	0 $)_{\mathbb{H}^*}$

Figure 5.9: Existential Unforgeability: Gap between $\mathbf{G}_3$ and $\mathbf{G}_4$

Game $\mathbf{G}_{3.j.0}$: The state of the game at that point is the following, for the keys, the signatures, and the verification vectors

$$\mathbf{k}^*_{\mathsf{id},0} = (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, \delta''_{\mathsf{id}})_{\mathbb{B}^*} \qquad \mathbf{r}^*_{\mathsf{id},1} = (\delta_{\mathsf{id}}, 0, 0, \psi_{\mathsf{id},1}, 0, 0, 0, 0)_{\mathbb{H}^*}$$
$$\mathbf{r}^*_{\mathsf{id},2} = (0, \delta_{\mathsf{id}}, 0, \psi_{\mathsf{id},2}, 0, 0, 0, 0)_{\mathbb{H}^*}$$
$$\mathbf{r}^*_{\mathsf{id},3} = (0, 0, \delta_{\mathsf{id}}, \psi_{\mathsf{id},3}, 0, 0, 0, 0)_{\mathbb{H}^*}$$

$i \geq j$: $U_i^* = (\xi_i\delta_i, \zeta_i, 0, 0)_{\mathbb{B}^*}$ $\qquad V_i^* = (\xi_i\delta_i(1, H_i, H_i'), \nu_i, 0, 0, 0, 0)_{\mathbb{H}^*}$

$i < j$: $U_i^* = (\xi_i\delta_i, \zeta_i, 0, \rho_i)_{\mathbb{B}^*}$ $\qquad V_i^* = (\xi_i\delta_i(1, H_i, H_i'), \nu_i, 0, \tau_i, 0, 0)_{\mathbb{H}^*}$

$$u = (-s_0 - s, 0, \kappa_0, -r_0)_{\mathbb{B}} \qquad v = (s + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, \omega, 0, 0)_{\mathbb{H}}$$

Game $\mathbf{G}_{3.j.1}$: We change the j-th signature into:

$$U_j^* = (\xi_j\delta_j, \zeta_j, 0, \rho_j)_{\mathbb{B}^*} \qquad V_j^* = (\xi_j\delta_j(1, H_j, H_j'), \nu_j, 0, \rho_j \cdot r_0/\omega, 0, 0)_{\mathbb{H}^*}$$

with a random $\rho_j \xleftarrow{\$} \mathbb{Z}_q$. The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_2$: one applies the SubSpace-Ind property, on $(\mathbb{B}, \mathbb{B}^*)_{1,4}$ and $(\mathbb{H}, \mathbb{H}^*)_{1,6}$. Indeed, we can consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \tau \bmod q$ with either $\tau = 0$

or random, which are indistinguishable situations under the DDH assumption in $\mathbb{G}_2$. One notes that we can virtually set $a' = r_0/\omega \cdot a$ and $c' = r_0/\omega \cdot c$, which makes $c' - a'b = r_0/\omega \cdot \tau$.

Let us assume we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$, $(\mathbb{W}, \mathbb{W}^*)$. Then we define the matrices

$$B' = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{1,4} \quad B = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{1,4} \quad H' = \begin{pmatrix} 1 & a' \\ 0 & 1 \end{pmatrix}_{1,6} \quad H = \begin{pmatrix} 1 & 0 \\ -a' & 1 \end{pmatrix}_{1,6}$$
$$\mathbb{B}^* = B' \cdot \mathbb{U}^* \qquad \mathbb{B} = B \cdot \mathbb{U} \qquad \mathbb{H}^* = H' \cdot \mathbb{W}^* \qquad \mathbb{H} = H \cdot \mathbb{W}$$

The vectors $\mathbf{b}_4$ and $\mathbf{h}_6$ can not be computed, but they are hidden from the adversary's view. The j-th signature is now computed as:

$$U_j^* = (b, \zeta_j, 0, c)_{\mathbb{U}^*} = (b, \zeta_j, 0, \tau)_{\mathbb{B}^*}$$
$$V_j^* = (b(1, H_j, H_j'), \nu_j, 0, c', 0, 0)_{\mathbb{W}^*} = (b(1, H_j, H_j'), \nu_j, 0, r_0/\omega \cdot \tau, 0, 0)_{\mathbb{H}^*}$$

This is the expected signature, with $\xi_j = b/\delta_j$. Since $b \cdot G_2$ is known, we can use it to simulate $S_{j,\lambda}^*$. When $\tau = 0$, we are in the previous game, meanwhile when $\tau = \rho_j$ is random, we are in the new game.

Since the vectors $\mathbf{b}_4$ and $\mathbf{h}_6$ can not be computed, we cannot define the verification vectors in the new bases:

$$\begin{aligned} u &= (-s_0 - s, 0, \kappa_0, -r_0)_{\mathbb{U}} = (-s_0 - s - ar_0, 0, \kappa_0, -r_0)_{\mathbb{B}} \\ &= (-s_0 - s', 0, \kappa_0, -r_0)_{\mathbb{B}} \\ v &= (s + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, \omega, 0, 0)_{\mathbb{W}} \\ &= (s + a'\omega + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, \omega, 0, 0)_{\mathbb{H}} \\ &= (s + ar_0 + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, \omega, 0, 0)_{\mathbb{H}} \\ &= (s' + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, \omega, 0, 0)_{\mathbb{H}} \end{aligned}$$

They remain consistent, as one can simply replace the random s by $s' = s + ar_0$. Hence, $\mathsf{Adv}_{3.j.0} - \mathsf{Adv}_{3.j.1} \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{3.j.2}$: We change again the j-th signature into:

$$U_j^* = (\xi_j\delta_j, \zeta_j, 0, \rho_j)_{\mathbb{B}^*} \qquad V_j^* = (\xi_j\delta_j(1, H_j, H_j'), \nu_j, 0, \tau_j, 0, 0)_{\mathbb{H}^*}$$

with random and independent $\rho_j, \tau_j \xleftarrow{\$} \mathbb{Z}_q$.

To do this, we use the Index-Ind Property from Theorem 7 on $(\mathbb{H}, \mathbb{H}^*)$ on the 6 coordinates 1,2,3,6,7,8, of dimension 6 from the view $(\mathbf{h}_1^*, \mathbf{h}_2^*, \mathbf{h}_3^*, \mathbf{h}_6^*, \mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3)$ with hidden vectors $(\mathbf{h}_7^*, \mathbf{h}_8^*, \mathbf{h}_6, \mathbf{h}_7, \mathbf{h}_8)$, with

$$\begin{aligned} v &= (s + \theta\bar{H} + \theta'\bar{H}', -\theta, -\theta', 0, \kappa, \omega, 0, 0)_{\mathbb{H}} \\ &= (s + \theta(\bar{H} + \rho\bar{H}'), -\theta, -\rho\theta, 0, \kappa, \omega, 0, 0)_{\mathbb{H}} \\ &= (s, 0, 0, 0, \kappa, 0, 0, 0)_{\mathbb{H}} + (\theta(\bar{H} + \rho\bar{H}', -1, -\rho), 0, 0, \omega, 0, 0)_{\mathbb{H}} \\ V_j^* &= (\xi_j\delta_j(1, H_j, H_j'), \nu_j, 0, \tau_j, 0, 0)_{\mathbb{H}^*} \\ &= (0, 0, 0, \nu_j, 0, 0, 0, 0)_{\mathbb{H}^*} + (\xi_j\delta_j(1, H_j, H_j'), 0, 0, \tau_j, 0, 0)_{\mathbb{H}^*} \end{aligned}$$

with $\rho = \theta'/\theta$, where ρ needs to be decided before the start of the game, as θ, θ' can be too. Under the collision resistance of the hash functions, we can assume that $\bar{H} \neq H_j$ and $\bar{H}' \neq H_j'$. Then, one cannot distinguish between the two following

$$V_j^* = (\xi_j\delta_j(1, H_j, H_j'), \nu_j, 0, r_0/\omega \cdot \tau, 0, 0)_{\mathbb{H}^*}$$
$$V_j^* = (\xi_j\delta_j(1, H_j, H_j'), \nu_j, 0, \tau_j, 0, 0)_{\mathbb{H}^*}$$

with random τ_j, and known random $\xi_j\delta_j \cdot G_2$, as the latter can either be chosen or is the $b \cdot G_2$ from the DDH instances. Hence, $\mathsf{Adv}_{3.j.1} - \mathsf{Adv}_{3.j.2} \leq 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 1/q$.

Existential Unforgeability: Hybrid Sequence. In this sequence, we only follow elements in bases $\mathbb{D}, \mathbb{D}^*$ as other vectors are not modified. See Figure 5.10.

$\mathbf{G}_{1.\Delta.2.p.0}$	Hybrid sequence from $\mathbf{G}_{1.\Delta.3}$ to $\mathbf{G}_{1.\Delta.4}$											
$\mathsf{id} > \Delta$	$\mathbf{k}^*_{\mathsf{id},0}$	δ_{id}	$\phi_{\mathsf{id},0}$	0	0							$)_{\mathbb{B}^*}$
	$\mathbf{k}^*_{\Delta,0}$	δ_Δ	$\phi_{\Delta,0}$	0	δ'_Δ							$)_{\mathbb{B}^*}$
$\mathsf{id} < \Delta$	$\mathbf{k}^*_{\mathsf{id},0}$	δ_{id}	$\phi_{\mathsf{id},0}$	0	δ''_{id}							$)_{\mathbb{B}^*}$
	U^*_i	$\xi_i\delta_i$	ζ_i	0	0							$)_{\mathbb{B}^*}$
	u	$-s_0 - s$	0	κ_0	$-r_0$							$)_{\mathbb{B}}$
	$\mathbf{r}^*_{\mathsf{id},1}$	δ_{id}	0	0	$\psi_{\mathsf{id},1}$	0	0	0	0			$)_{\mathbb{H}^*}$
	$\mathbf{r}^*_{\mathsf{id},2}$	0	δ_{id}	0	$\psi_{\mathsf{id},2}$	0	0	0	0			$)_{\mathbb{H}^*}$
	$\mathbf{r}^*_{\mathsf{id},3}$	0	0	δ_{id}	$\psi_{\mathsf{id},3}$	0	0	0	0			$)_{\mathbb{H}^*}$
	V^*_i	$\xi_i\delta_i$	$\xi_i\delta_i H_i$	$\xi_i\delta_i H_i$	ν_{id}	0	0	0	0			$)_{\mathbb{H}^*}$
	v	$s+\theta\bar{H}+\theta'\bar{H}'$	$-\theta$	$-\theta'$	0	0	ω	0	0			$)_{\mathbb{H}}$
$\mathsf{id} \neq \Delta$	$\mathbf{k}^*_{\mathsf{id},t}$	δ_{id}	$\pi_{\mathsf{id},t}$	$\pi_{\mathsf{id},t}t$	$\phi_{\mathsf{id},t}$	0	0	0	0	0	0	$)_{\mathbb{D}^*}$
	$\mathbf{k}^*_{\Delta,t}$	δ_Δ	$\pi_{\Delta,t}$	$\pi_{\Delta,t}t$	$\phi_{\Delta,t}$	0	δ'_Δ	0	$\delta'_\Delta z_t$	0	0	$)_{\mathbb{D}^*}$
	$S^*_{i,\lambda}$	$\beta'_{i,\lambda}$	$\gamma_{i,\lambda}$	$\gamma_{i,\lambda}t_\lambda$	$q_{i,\lambda}$	0	0	0	0	0	0	$)_{\mathbb{D}^*}$
$t_\lambda \geq p$	c_λ	s_λ	$\theta_\lambda t_\lambda$	$-\theta_\lambda$	0	κ_λ	r_λ	0	0	0	0	$)_{\mathbb{D}}$
$t_\lambda < p$	c_λ	s_λ	$\theta_\lambda t_\lambda$	$-\theta_\lambda$	0	κ_λ	0	0	$\frac{r_\lambda}{z_t}$	0	0	$)_{\mathbb{D}}$
$\mathbf{G}_{1.\Delta.2.p.1}$	Formal change of basis for $(\mathbb{D}^*, \mathbb{D})$: duplication in $\mathbb{D}^*$											
	$\mathbf{k}^*_{\Delta,t}$	δ_Δ	$\pi_{\Delta,t}$	$\pi_{\Delta,t}t$	$\phi_{\Delta,t}$	0	δ'_Δ	δ'_Δ	$\delta'_\Delta z_t$	0	0	$)_{\mathbb{D}^*}$
$\mathbf{G}_{1.\Delta.2.p.2}$	Swap r_λ: $\mathsf{Swap\text{-}Ind}$ on (4,5,6) in $(\mathbb{D}, \mathbb{D}^*)$											
$t_\lambda = p$	c_λ	s_λ	$\theta_\lambda p$	$-\theta_\lambda$	0	κ_λ	0	r_λ	0	0	0	$)_{\mathbb{D}}$
$\mathbf{G}_{1.\Delta.2.p.3}$	$\mathsf{Index\text{-}Ind}$ on all $t \neq p$ in $(\mathbb{D}^*, \mathbb{D})$											
$t \neq p$	$\mathbf{k}^*_{\Delta,t}$	δ_Δ	$\pi_{\Delta,t}$	$\pi_{\Delta,t}t$	$\phi_{\Delta,t}$	0	δ'_Δ	$\frac{\delta'_\Delta z_t}{z_p}$	$\delta'_\Delta z_t$	0	0	$)_{\mathbb{D}^*}$
$\mathbf{G}_{1.\Delta.2.p.4}$	Remove α: Formal change of basis on (6,7) in $(\mathbb{D}^*, \mathbb{D})$											
	$\mathbf{k}^*_{\Delta,t}$	δ_Δ	π_Δ	$\pi_\Delta t$	$\phi_{\Delta,t}$	0	δ'_Δ	0	$\delta'_\Delta z_t$	0	0	$)_{\mathbb{D}^*}$
$t_\lambda = p$	c_λ	s_λ	$\theta_\lambda p$	$-\theta_\lambda$	0	κ_λ	0	α	$\frac{r_\lambda}{z_p}$	0	0	$)_{\mathbb{D}}$
$\mathbf{G}_{1.\Delta.2.p.5}$	$\mathsf{SubSpace\text{-}Ind}$ on (2,6) in $(\mathbb{D}, \mathbb{D}^*)$											
$t_\lambda = p$	c_λ	s_λ	$\theta_\lambda p$	$-\theta_\lambda$	0	κ_λ	0	0	$\frac{r_\lambda}{z_p}$	0	0	$)_{\mathbb{D}}$

Figure 5.10: Existential Unforgeability: Hybrid Sequence

Game $\mathbf{G}_{1.\Delta.2.p.0}$: The state of the game at that point is the following, for the keys

$$\mathbf{k}^*_{\Delta,t} = (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, 0, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*}$$
$$\mathbf{k}^*_{\mathsf{id},t} = (\delta_{\mathsf{id}}, \pi_{\mathsf{id},t}(1,t), \phi_{\mathsf{id},t}, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}$$

the signatures,

$$S^*_{i,\lambda} = (\beta'_{i,\lambda}, \gamma_{i,\lambda}(1,t_\lambda), q_{i,\lambda}, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}$$

and ciphertexts

$$c_\lambda = (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, r_\lambda, 0, 0, 0, 0)_{\mathbb{D}} \qquad t > p$$
$$c_\lambda = (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, 0, 0, r_\lambda/z_t, 0, 0)_{\mathbb{D}} \qquad t \leq p$$

Game $\mathbf{G}_{1.\Delta.2.p.1}$: We change the keys into:

$$\mathbf{k}^*_{\Delta,t} = (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*}$$
$$\mathbf{k}^*_{\mathsf{id},t} = (\delta_{\mathsf{id}}, \pi_{\mathsf{id},t}(1,t), \phi_{\mathsf{id},t}, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}$$

To do this, we define the matrices

$$D' = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}_{6,7} \qquad D = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}_{6,7}$$

which modifies the hidden vectors $\mathbf{d}_7, \mathbf{d}^*_6$. Since they are not in the adversary's view, the advantage is perfect.

The keys are modified in the following way. Note that keys other than Δ and signatures are unmodified as they all have a 0 in the 6-th position.

$$\begin{aligned}\mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, 0, \delta'_\Delta z_t, 0, 0)_{\mathbb{V}^*} \\ &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*}\end{aligned}$$

Meanwhile, the ciphertexts are not modified because they all have a 0 in the 7-th position. The adversary gains no advantage in this game: $\mathsf{Adv}_{1.\Delta.2.p.0} = \mathsf{Adv}_{1.\Delta.2.p.1}$.

Game $\mathbf{G}_{1.\Delta.2.p.2}$: We change only the verification texts linked to the p-th attribute into:

$$t_\lambda = p, \quad c_\lambda = (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, \kappa_\lambda, 0, r_\lambda, 0, 0, 0)_{\mathbb{D}}$$

The previous game and this game are indistinguishable under the DSDH assumption in $\mathbb{G}_1$: one applies the Swap-Ind property, on $(\mathbb{D}, \mathbb{D}^*)_{5,6,7}$. Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, d \cdot G_1)$, where $d = ab + \tau \bmod q$ with either $\tau = 0$ or $\tau = 1$, which are indistinguishable situations under the DSDH assumption.

One chooses additional scalars $\alpha_\lambda = -r_\lambda$ and $\beta_\lambda \xleftarrow{\$} \mathbb{Z}_q$ to virtually set $b_\lambda = \alpha_\lambda \cdot b + \beta_\lambda$ and $d_\lambda = \alpha_\lambda \cdot d + \beta_\lambda \cdot a$, which makes $d_\lambda - ab_\lambda = \alpha_\lambda \cdot \tau = -r_\lambda \cdot \tau$.

$$D = \begin{pmatrix} 1 & a & -a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{5,6,7} \qquad D' = \begin{pmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ a & 0 & 1 \end{pmatrix}_{5,6,7}$$
$$\mathbb{D}^* = D' \cdot \mathbb{V}^* \qquad \mathbb{D} = D \cdot \mathbb{V}$$

The vectors $\mathbf{d}^*_6, \mathbf{d}^*_7$ can not be computed, but they are not in the view of the adversary. The verification texts for the p-th attribute is changed as follows

$$\begin{aligned}c_\lambda &= (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, 0, r_\lambda, 0, 0, 0, 0)_{\mathbb{D}} + (0, 0, 0, 0, b_\lambda, d_\lambda, -d_\lambda, 0, 0, 0)_{\mathbb{V}} \\ &= (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, b_\lambda, r_\lambda + \alpha_\lambda \cdot \tau, -\alpha_\lambda \cdot \tau, 0, 0, 0)_{\mathbb{D}} \\ &= (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, b_\lambda, r_\lambda - r_\lambda \cdot \tau, r_\lambda \cdot \tau, 0, 0, 0)_{\mathbb{D}}\end{aligned}$$

When $\tau = 0$, we are in the previous game, meanwhile when $\tau = 1$, we are in the next game. Other verification texts are generated in $\mathbb{D}$ directly.

Keys and signatures are unchanged because they have the same value on 6th and 7th columns, either 0 or δ'_Δ. Notably, those with 0 on these positions are keys different than Δ, and can be fully generated in $\mathbb{D}^*$.

$$\begin{aligned}\mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_t, 0, 0)_{\mathbb{V}^*} \\ &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, a \cdot \delta'_\Delta - a \cdot \delta'_\Delta, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*} \\ &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*}\end{aligned}$$

The advantage of the adversary is: $\mathsf{Adv}_{1.\Delta.2.p.1} - \mathsf{Adv}_{1.\Delta.2.p.2} \leq \mathsf{Adv}^{\mathsf{dsdh}}_{\mathbb{G}_1}(t)$.

Game $\mathbf{G}_{1.\Delta.2.p.3}$: We keep the δ'_Δ value (at the 7-th hidden position) in the key for the p-th attribute only, and replace all values in other keys by $\delta'_\Delta z_t/z_p$:

$$\begin{aligned}\mathbf{k}^*_{\Delta,p} &= (\delta_\Delta, \pi_{\Delta,p}(1,p), \phi_{\Delta,p}, 0, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_p, 0, 0)_{\mathbb{D}^*}\\ \mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, \delta'_\Delta z_t/z_p, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*}\\ \mathbf{k}^*_{\mathsf{id},t} &= (\delta_{\mathsf{id}}, \pi_{\mathsf{id},t}(1,t), \phi_{\mathsf{id},t}, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}\end{aligned}$$

To show this is possible without impacting the other vectors, we use the Index-Ind property, but in another level of sequence of hybrid games, for $\gamma \in \{1,\dots,T\}\backslash\{p\}$. We will again enumerate γ in their order of appearance in the security game (whether in key queries, or signature queries).

Game $\mathbf{G}_{1.\Delta.2.p.2.\gamma}$: We consider

$$\begin{aligned}\mathbf{k}^*_{\Delta,p} &= (\delta_\Delta, \pi_{\Delta,p}(1,p), \phi_{\Delta,p}, 0, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_p, 0, 0)_{\mathbb{D}^*} && \\ \mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, \delta'_\Delta z_t/z_p, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*} && p \neq t < \gamma\\ \mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*} && t \geq \gamma\\ c_\lambda &= (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, 0, 0, r_\lambda/z_t, 0, 0)_{\mathbb{D}} && t < p\\ c_\lambda &= (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, \kappa_\lambda, 0, r_\lambda, 0, 0, 0)_{\mathbb{D}} && \\ c_\lambda &= (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, r_\lambda, 0, 0, 0, 0)_{\mathbb{D}} && t > p\end{aligned}$$

When $\gamma = 1$, this is the previous game: $\mathbf{G}_{1.\Delta.2.p.2.1} = \mathbf{G}_{1.\Delta.2.p.2}$, whereas with $\gamma = T+1$, this is the current game: $\mathbf{G}_{1.\Delta.2.p.2.T+1} = \mathbf{G}_{1.\Delta.2.p.3}$.

To do this, we use the Index-Ind Property from Section 7 on $(\mathbb{D}, \mathbb{D}^*)$ on the 5 coordinates 2, 3, 6, 9, 10 of dimension 5 from the view $(\mathbf{d}^*_2, \mathbf{d}^*_3, \mathbf{d}^*_7, \mathbf{d}_2, \mathbf{d}_3)$ with hidden vectors $(\mathbf{d}^*_9, \mathbf{d}^*_{10}, \mathbf{d}_7, \mathbf{d}_9, \mathbf{d}_{10})$, with

$$\begin{aligned}c_\lambda &= (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, \kappa_\lambda, 0, r_\lambda, 0, 0, 0)_{\mathbb{D}} && \\ \mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*} && t = \gamma\end{aligned}$$

Hence, we have $\mathsf{Adv}_{1.\Delta.2.p.2} - \mathsf{Adv}_{1.\Delta.2.p.3} \leq 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 2 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$

Game $\mathbf{G}_{1.\Delta.2.p.4}$: We use a theoretic information change to uniformize the keys and verification texts. We change the key into:

$$\begin{aligned}\mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, 0, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*}\\ \mathbf{k}^*_{\Delta,p} &= (\delta_\Delta, \pi_{\Delta,p}(1,p), \phi_{\Delta,p}, 0, \delta'_\Delta, 0, \delta'_\Delta z_p, 0, 0)_{\mathbb{D}^*}\end{aligned}$$

Meanwhile the verification texts associated to attribute p are changed into:

$$c_\lambda = (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, \kappa_\lambda, 0, r_\lambda, r_\lambda/z_p, 0, 0)_{\mathbb{D}} \qquad t = p$$

To do this we use the following matrices:

$$D' = \begin{pmatrix} 1 & 0 \\ 1/z_p & 1 \end{pmatrix}_{7,8} \qquad D = \begin{pmatrix} 1 & -1/z_p \\ 0 & 1 \end{pmatrix}_{7,8}$$

The vectors $\mathbf{d}_7, \mathbf{d}^*_7, \mathbf{d}^*_8$ must not be in the adversary's view, but since they are hidden the advantage is perfect.

Keys for Δ are modified in the following way:

$$\begin{aligned}\mathbf{k}^*_{\Delta,t} &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, \delta'_\Delta z_t/z_p, \delta'_\Delta z_t, 0, 0)_{\mathbb{V}^*}\\ &= (\delta_\Delta, \pi_{\Delta,t}(1,t), \phi_{\Delta,t}, 0, \delta'_\Delta, 0, \delta'_\Delta z_t, 0, 0)_{\mathbb{D}^*}\\ \mathbf{k}^*_{\Delta,p} &= (\delta_\Delta, \pi_{\Delta,p}(1,p), \phi_{\Delta,p}, 0, \delta'_\Delta, \delta'_\Delta, \delta'_\Delta z_p, 0, 0)_{\mathbb{V}^*}\\ &= (\delta_\Delta, \pi_{\Delta,p}(1,p), \phi_{\Delta,p}, 0, \delta'_\Delta, 0, \delta'_\Delta z_p, 0, 0)_{\mathbb{D}^*}\end{aligned}$$

Verification texts associated to p are changed as well:

$$\begin{aligned}c_\lambda &= (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, \kappa_\lambda, 0, r_\lambda, 0, 0, 0)_{\mathbb{V}} && t = p\\ &= (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, \kappa_\lambda, 0, r_\lambda, r_\lambda/z_p, 0, 0)_{\mathbb{D}}\end{aligned}$$

We note that for $t \neq p, c_\lambda$ are unchanged. The same goes for all keys different than Δ and all signatures, as their component on the relevant positions are all 0 (7-th for verification texts, 7-th and 8-th for keys and signatures).

The adversary gains no advantage in this game: $\mathsf{Adv}_{1.\Delta.2.p.3} = \mathsf{Adv}_{1.\Delta.2.p.4}$.

Game $\mathbf{G}_{1.\Delta.2.p.5}$: We remove r_λ from the verification texts associated to the p-th attribute.

$$c_\lambda = (s_\lambda, \theta_\lambda t_\lambda, -\theta_\lambda, 0, \kappa_\lambda, 0, 0, r_\lambda/z_p, 0, 0)_{\mathbb{D}} \qquad t_\lambda = p$$

The previous game and this game are indistinguishable under the DSDH assumption in $\mathbb{G}_1$: one applies the SubSpace-Ind property, on $(\mathbb{D}, \mathbb{D}^*)_{5,7}$. Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, d \cdot G_1)$, where $d = ab + \tau \bmod q$ with either $\tau = 0$ or $\tau = 1$, which are indistinguishable situations under the DSDH assumption.

One chooses additional scalar $\beta_\lambda \xleftarrow{\$} \mathbb{Z}_q$ to virtually set $b_\lambda = r_\lambda \cdot b + \beta_\lambda$ and $d_\lambda = r_\lambda \cdot d + \beta_\lambda \cdot a$, which makes $d_\lambda - ab_\lambda = r_\lambda \cdot \tau$.

$$D' = \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix}_{5,7} \qquad D = \begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix}_{5,7}$$
$$\mathbb{D}^* = D' \cdot \mathbb{V}^* \qquad \mathbb{D} = D \cdot \mathbb{V}$$

The vector $\mathbf{d}^*_7$ cannot be computed, but they are not in the adversary's view. The verification texts for each λ so that $t_\lambda = p$ are changed in the following way:

$$\begin{aligned}c_\lambda &= (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, 0, 0, 0, r_\lambda, r_\lambda/z_p, 0, 0)_{\mathbb{D}} + (0, 0, 0, 0, b_\lambda, 0, d_\lambda, 0, 0, 0)_{\mathbb{V}}\\ &= (s_\lambda, \theta_\lambda p, -\theta_\lambda, 0, b_\lambda, 0, r_\lambda \cdot \tau, r_\lambda/z_p, 0, 0)_{\mathbb{D}}\end{aligned}$$

When $\tau = 1$, we are in the previous game, meanwhile when $\tau = 0$, we are in the next game. Other verification texts are unchanged and can be fully generated in $\mathbb{V}$. Keys and signatures are unchanged and can be fully generated in $\mathbb{V}^*$ because they all have a value of 0 on the 7-th position at that point in the hybrid game. The advantage of the adversary is: $\mathsf{Adv}_{1.\Delta.2.p.4} - \mathsf{Adv}_{1.\Delta.2.p.5} \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

5.4 Discussion

We discuss here the features and differences between these schemes, those of Okamoto and Takashima [OT12b, OT13], and our contribution schemes presented in the two next section, to explain some of our design choices.

5.4.1 ABE

One of our target application regarding ABE is tracing. We decided to use codewords for that application. Full details are given on Section 6.5, but we explain here at a high level some of the shortcomings of the traditional approach. The main difficulty for traitor-tracing is that the traitor *can never know he is being traced*, else he will adopt a strategy to make the tracing mode ineffective (for example by not answering). However, for ABE, one must access freely the attributes given to him in the ciphertext in order to decrypt. This is a huge advantage for the traitor, because if tracing mode relies on the attributes in the ciphertext, since the attributes must be freely accessible for decryption, the adversary will detect tracing mode. Based on this observation, we arrived at the following conclusion: the attributes used for tracing must be freely available for decryption, but at the same time they must be able to contain some *trapdoor* that is invisible for the traitor, so he doesn't notice whether tracing mode is on. We call this property of trapdoor being unnoticeable inside attributes as *Attribute-Indistinguishability*. These notions will be fully discussed in the next section.

5.4.2 ABS

Our ABS with delegation, presented in Section 7, proposes two kind of delegations: one where the attributes are delegated, and one where a pre-signed policy is delegated, where it can sign any message afterwards. This last method of delegation is the reason we have decorrelated the hash of the access-tree and the message in 5.3.2, contrary to the traditional approaches seen in [OT11, EGK14]. This decorrelation relies on an indexing with an orthogonal subspace of size 3 (see Theorem 9). While Okamoto-Takashima presented a generic indexing for subspace of any size, concrete instanciation always relied on size two. Our scheme instantiates a concrete application for an indexing of larger size. We show that these delegations are compatible with delegation, using a new approach relying on linearly-homomorphic signatures [HPP20] instead of the traditional NIZK approach to encrypt the identity of the signer.

Chapter

6 ABE with Switchable Attributes

Chapter content

We introduce our main contribution regarding KP-ABE. The ideal original idea came from Waters' Dual System Encryption [Wat09] that was developped to make attribute-based (as well as identity-based) schemes adaptive, with the DSE (see Section 5.1).

The main idea of this new construction is to take Waters' proof technique and look for applications in the case where the semi-functionality introduced for proof only could be used concretely as a scheme functionality. For that, we introduce new states for the keys and ciphertexts that the authority can switch to simulate the semi-functionality, hence the name of our Switchable KP-ABE. This completely new primitive of ABE is then proven compatible with a well-known feature of broadcast-like encryption: Traitor-Tracing.

6.1 Independent Leaves

Before going on the Switchable Attributes, we need to identify a property called *independent leaves* to guarantee the Attribute-Indistinguishability notion we introduce in the next section. This property essentially describes leaves for which the secret share leaks no information about any of the other leaves in the access-tree. As a toy example for non-independent leaves, consider a trivial tree with only an OR root node with two children λ_1 and λ_2. Then any information the adversary gains on a_{λ_1} can be used for a_{λ_2} as $a_{\lambda_1} = a_{\lambda_2}$. This is problematic in the case where we would want to trick the adversary with a "secretly incorrect" attribute for λ_1 (in a sense that will be explained later), because he could compare it to the information he received for λ_2 and detect any incoherence between the two.

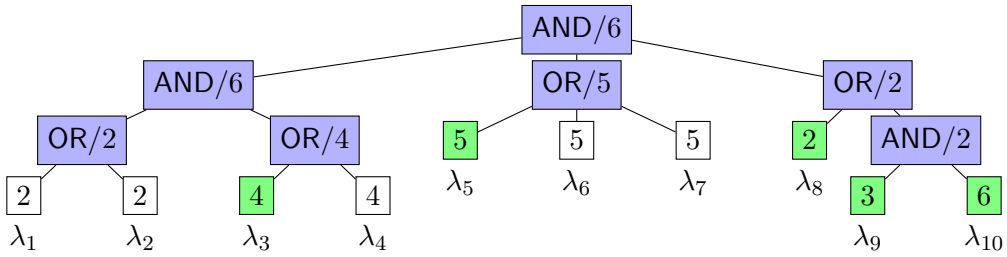

Figure 6.1: A 6-labeling in $\mathbb{Z}/7\mathbb{Z}$, with a non-satisfying set of (colored) leaves: leaves λ_8, λ_9 and λ_{10} are not independent as $a_{\lambda_8} = a_{\lambda_9} + a_{\lambda_{10}} \bmod 7$. However, λ_3 and λ_5 are independent among the green leaves.

Definition 9 (Independent Leaves) Given an access-tree $\mathcal{T}$ and a set Γ so that $\mathcal{T}(\Gamma) = 0$, we call independent leaves in $\mathcal{L}_\Gamma$, the leaves μ such that, given only $(a_\lambda)_{\lambda \in \mathcal{L}_\Gamma \setminus \{\mu\}}$, a_μ is unpredictable: for any y, the two distributions $\mathcal{D}_y^{\$}(\Gamma) = \{(a_\lambda)_{\lambda \in \mathcal{L}_\Gamma}\}$ and $\mathcal{D}_y(\Gamma, \mu) = \{(b_\mu) \cup (a_\lambda)_{\lambda \in \mathcal{L}_\Gamma \setminus \{\mu\}}\}$ are perfectly indistinguishable, where $(a_\lambda)_\lambda \xleftarrow{\$} \Lambda_y(\mathcal{T})$ and $b_\mu \xleftarrow{\$} \mathbb{Z}_p$.

Intuitively, given $(a_\lambda)_{\lambda \in \mathcal{L}_\Gamma \setminus \{\mu\}}$, one can complete it into a valid labeling by setting the right value for a_μ. Note that the root label can be any value y as $\mathcal{T}(\Gamma) = 0$, thus y is unpredictable. With the illustration on Figure 6.1, with a non-satisfied tree, when only colored leaves are set to True, leaves λ_3 and λ_5 are independent among the True leaves $\{\lambda_3, \lambda_5, \lambda_8, \lambda_9, \lambda_{10}\}$. But leaves λ_8, λ_9 and λ_{10} are not independent as $a_{\lambda_8} = a_{\lambda_9} + a_{\lambda_{10}} \bmod 7$ for any random labeling.

6.2 Switchable Leaves and Attributes

For a Key-Policy ABE with Switchable Attributes (SA-KP-ABE), leaves in the access-tree can be made active or passive, and attributes in the ciphertext can be made valid or invalid. We thus enhance the access-tree $\mathcal{T}$ into $\tilde{\mathcal{T}} = (\mathcal{T}, \mathcal{L}_a, \mathcal{L}_p)$, where the implicit set of leaves $\mathcal{L} = \mathcal{L}_a \cup \mathcal{L}_p$ is now the explicit disjoint union of the active-leaf and passive-leaf sets. Similarly, a ciphertext will be associated to the pair (Γ_v, Γ_i), also referred as a disjoint union $\Gamma = \Gamma_v \cup \Gamma_i$, of the valid-attribute and invalid-attribute sets.

We note $\tilde{\mathcal{T}}(\Gamma_v, \Gamma_i) = 1$ if there is an evaluation pruned tree $\mathcal{T}'$ of $\mathcal{T}$ that is satisfied by $\Gamma = \Gamma_v \cup \Gamma_i$ (*i.e.*, $\mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma)$), with the additional condition that all the active leaves in $\mathcal{T}'$ correspond only to valid attributes in Γ_v: $\exists \mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma), \forall \lambda \in \mathcal{T}' \cap \mathcal{L}_a$, $A(\lambda) \in \Gamma_v$. In other words, this means that an invalid attribute in the ciphertext should be considered non-existant for active leaves only.

We also have to enhance the partial order on $\mathcal{T}$ to $\tilde{\mathcal{T}}$, so that we can deal with delegation: $\tilde{\mathcal{T}}' = (\mathcal{T}', \mathcal{L}'_a, \mathcal{L}'_p) \leq \tilde{\mathcal{T}} = (\mathcal{T}, \mathcal{L}_a, \mathcal{L}_p)$ if and only if $\mathcal{T}' \leq \mathcal{T}$, $\mathcal{L}'_a \cap \mathcal{L}_p = \mathcal{L}'_p \cap \mathcal{L}_a = \emptyset$ and $\mathcal{L}'_a \subseteq \mathcal{L}_a$. More concretely, $\mathcal{T}'$ must be more restrictive, existing leaves cannot change their passive or active status, and new leaves can only be passive.

6.3 KP-ABE with Switchable Attributes (SA-KP-ABE)

6.3.1 Definition of SA-KP-ABE

We can now define the algorithms of an SA-KP-ABE, with the usual description of Key Encapsulation Mechanism. In our definitions, there are two secret keys: the master secret key MK for the generation of users' keys, and the secret key SK for running the advanced encapsulation with invalid attributes:

$\mathsf{Setup}(1^\kappa)$**.** From the security parameter κ, the algorithm defines all the global parameters PK, the secret key SK and the master secret key MK;

$\mathsf{KeyGen}(\mathsf{MK}, \tilde{\mathcal{T}})$**.** The algorithm outputs a key $\mathsf{dk}_{\tilde{\mathcal{T}}}$ which enables the user to decapsulate keys generated under a set of attributes $\Gamma = \Gamma_v \uplus \Gamma_i$ if and only if $\tilde{\mathcal{T}}(\Gamma_v, \Gamma_i) = 1$;

$\mathsf{Delegate}(\mathsf{dk}_{\tilde{\mathcal{T}}}, \tilde{\mathcal{T}}')$**.** Given a key $\mathsf{dk}_{\tilde{\mathcal{T}}}$, generated from either the KeyGen or the $\mathsf{Delegate}$ algorithms, for a policy $\tilde{\mathcal{T}}$ and a more restrictive policy $\tilde{\mathcal{T}}' \leq \tilde{\mathcal{T}}$, the algorithm outputs a decryption key $\mathsf{dk}_{\mathcal{T}'}$;

$\mathsf{Encaps}(\mathsf{PK}, \Gamma)$**.** For a set Γ of (valid only) attributes, the algorithm generates the ciphertext C and an encapsulated key K;

$\mathsf{Encaps}^*(\mathsf{SK}, \Gamma_v, \Gamma_i)$**.** For a pair (Γ_v, Γ_i) of disjoint sets of valid/invalid attributes, the algorithm generates the ciphertext C and an encapsulated key K;

$\mathsf{Decaps}(\mathsf{dk}_{\tilde{\mathcal{T}}}, C)$**.** Given the key $\mathsf{dk}_{\tilde{\mathcal{T}}}$ from either KeyGen or $\mathsf{Delegate}$, and the ciphertext C, the algorithm outputs the encapsulated key K.

We stress that fresh keys (from the KeyGen algorithm) and delegated keys (from the $\mathsf{Delegate}$ algorithm) can both be used for decryption and can both be delegated. This allows multi-hop delegation.

On the other hand, one can note the difference between Encaps with PK and Encaps^* with SK, where the former runs the latter on the pair $(\Gamma, \emptyset)$. And as $\Gamma_i = \emptyset$, the public key is enough. This is thus still a public-key encryption scheme when only valid attributes are in the ciphertext, but the invalidation of some attributes require the secret key SK. As explained later in the section, this will lead to secret-key traceability, as only the owner of SK will be able to invalidate attributes for the tracing procedure (see Section 6.5). For correctness, the Decaps algorithm should output the encapsulated key K if and only if C has been generated for a pair (Γ_v, Γ_i) that satisfies the policy $\tilde{\mathcal{T}}$ of the decryption key $\mathsf{dk}_{\tilde{\mathcal{T}}}$: $\tilde{\mathcal{T}}(\Gamma_v, \Gamma_i) = 1$. The following security notion enforces this property. But some other indistinguishability notions need to be defined in order to be able to exploit these switchable attributes in more complex protocols.

6.3.2 Security Model for SA-KP-ABE

For the sake of simplicity, we focus on one-challenge definitions (one encapsulation with Real-or-Random encapsulated key, one user key with Real-or-All-Passive leaves, and one encapsulation with Real-or-All-Valid attributes), in the same vein as the Find-then-Guess security game. But the adversary could generate additional values, as they can either be publicly generated or an oracle is available. Then, the definitions can be turned into multi-challenge security games, with an hybrid proof, as explained in [BDJR97].

Definition 10 (Delegation-Indistinguishability for SA-KP-ABE) Del-IND security for SA-KP-ABE is defined by the following game:

Initialize: The challenger runs the Setup algorithm of SA-KP-ABE and gives the public parameters PK to the adversary;

Oracles: The following oracles can be called in any order and any number of times, except for $\mathsf{RoREncaps}$ which can be called only once.

- $\mathsf{OKeyGen}(\tilde{\mathcal{T}})$**:** this models a KeyGen-query for any access-tree $\tilde{\mathcal{T}} = (\mathcal{T}, \mathcal{L}_a, \mathcal{L}_p)$. It generates the decryption key but only outputs the index k of the key;
- $\mathsf{ODelegate}(k, \tilde{\mathcal{T}}')$**:** this models a $\mathsf{Delegate}$-query for any more restrictive access-tree $\tilde{\mathcal{T}}' \leq \tilde{\mathcal{T}}$, for the k-indexed generated decryption key for $\tilde{\mathcal{T}}$. It generates the decryption key but only outputs the index k' of the new key;

$\mathsf{OGetKey}(k)$**:** the adversary gets back the k-indexed decryption key generated by $\mathsf{OKeyGen}$ or $\mathsf{ODelegate}$ oracles;

$\mathsf{OEncaps}(\Gamma_v, \Gamma_i)$**:** The adversary may be allowed to issue Encaps^*-queries, with $(K, C) \leftarrow \mathsf{Encaps}^*(\mathsf{SK}, \Gamma_v, \Gamma_i)$, and C is returned;

$\mathsf{RoREncaps}(\Gamma_v, \Gamma_i)$**:** The adversary submits a unique real-or-random encapsulation query on a set of attributes $\Gamma = \Gamma_v \uplus \Gamma_i$. The challenger asks for an encapsulation query on (Γ_v, Γ_i) and receives (K_0, C). It also generates a random key K_1. It eventually flips a random coin b, and outputs (K_b, C) to the adversary;

Finalize(b'): The adversary outputs a guess b' for b. If for some access-tree $\tilde{\mathcal{T}}'$ corresponding to a key asked to the $\mathsf{OGetKey}$-oracle, $\tilde{\mathcal{T}}'(\Gamma_v, \Gamma_i) = 1$, on the challenge set (Γ_v, Γ_i), $\beta \xleftarrow{\$} \{0, 1\}$, otherwise one sets $\beta = b'$. One outputs β.

$\mathsf{Adv}^{\mathsf{del\text{-}ind}}(\mathcal{A})$ denotes the advantage of an adversary $\mathcal{A}$ in this game.

In the basic form of $\mathsf{Del\text{-}IND}$-security, where Encaps^* encapsulations are not available, the $\mathsf{RoREncaps}$-oracle only allows $\Gamma_i = \emptyset$, and no $\mathsf{OEncaps}$-oracle is available. But as Encaps (with $\Gamma_i = \emptyset$) is a public-key algorithm, the adversary can generate valid ciphertexts by himself. We will call it "$\mathsf{Del\text{-}IND}$-security for Encaps". For the more advanced security level, $\mathsf{RoREncaps}$-query will be allowed on any pair (Γ_v, Γ_i), with the additional $\mathsf{OEncaps}$-oracle. We will call it "$\mathsf{Del\text{-}IND}$-security for Encaps^*".

With these disjoint unions of $\mathcal{L} = \mathcal{L}_a \uplus \mathcal{L}_p$ and $\Gamma = \Gamma_v \uplus \Gamma_i$, we will also consider some indistinguishability notions on $(\mathcal{L}_a, \mathcal{L}_p)$ and (Γ_v, Γ_i), about which leaves are active or passive in $\mathcal{L} = \mathcal{L}_a \uplus \mathcal{L}_p$ for a given key, and which attributes are valid or invalid in $\Gamma = \Gamma_v \uplus \Gamma_i$ for a given ciphertext. The former will be the key-indistinguishability, whereas the latter will be attribute-indistinguishability. Again, as Encaps is public-key, the adversary can generate valid encapsulations by himself. However, we may provide access to an $\mathsf{OEncaps}$-oracle to allow Encaps^* queries, but with constraints in the final step, to exclude trivial attacks against key-indistinguishability. Similarly there will be constraints in the final step on the $\mathsf{OKeyGen}/\mathsf{ODelegate}$-queries for the attribute-indistinguishability.

Definition 11 (Key-Indistinguishability) $\mathsf{Key\text{-}IND}$ security for $\mathsf{SA\text{-}KP\text{-}ABE}$ is defined by the following game:

Initialize: The challenger runs the Setup algorithm of $\mathsf{SA\text{-}KP\text{-}ABE}$ and gives the public parameters PK to the adversary;

Oracles: $\mathsf{OKeyGen}(\tilde{\mathcal{T}})$, $\mathsf{ODelegate}(k, \tilde{\mathcal{T}}')$, $\mathsf{OGetKey}(k)$, $\mathsf{OEncaps}(\Gamma_v, \Gamma_i)$, and

$\mathsf{RoAPKeyGen}(\tilde{\mathcal{T}})$**:** The adversary submits one Real or All-Passive KeyGen-query for any access structure $\tilde{\mathcal{T}}$ of its choice, with a list $\mathcal{L} = \mathcal{L}_a \uplus \mathcal{L}_p$ of active and passive leaves, and gets $\mathsf{dk}_0 = \mathsf{KeyGen}(\mathsf{MK}, (\mathcal{T}, \mathcal{L}_a, \mathcal{L}_p))$ or $\mathsf{dk}_1 = \mathsf{KeyGen}(\mathsf{MK}, (\mathcal{T}, \emptyset, \mathcal{L}))$. It eventually flips a random coin b, and outputs dk_b to the adversary;

Finalize(b'): The adversary outputs a guess b' for b. If for some (Γ_v, Γ_i) asked to the $\mathsf{OEncaps}$-oracle, $\mathcal{T}(\Gamma_v \uplus \Gamma_i) = 1$, for the challenge access-tree $\mathcal{T}$ where $\mathcal{L} = \mathcal{L}_a \uplus \mathcal{L}_p$, $\beta \xleftarrow{\$} \{0, 1\}$, otherwise one sets $\beta = b'$. One outputs β.

$\mathsf{Adv}^{\mathsf{key\text{-}ind}}(\mathcal{A})$ denotes the advantage of an adversary $\mathcal{A}$ in this game.

In this first definition, the constraints in the finalize step require the adversary not to ask for an encapsulation on attributes that would be accepted by the policy with all-passive attributes in the leaves.

A second version deals with accepting policies: it allows encapsulations on attributes that would be accepted by the policy with all-passive leaves in the challenge key, as long as attributes associated to the active leaves in the challenge key and invalid attributes in the ciphertexts are **distinct**. Hence, the **Distinct Key-Indistinguishability** (dKey-IND) where **Finalize**(b') reads: *The adversary outputs a guess b' for b. If some active leaf $\lambda \in \mathcal{L}_a$ from the challenge key corresponds to some invalid attribute $t \in \Gamma_i$ in an* OEncaps*-query, then set $\beta \xleftarrow{\$} \{0,1\}$, otherwise set $\beta = b'$. One outputs β.*

Definition 12 (Attribute-Indistinguishability) Att-IND security for SA-KP-ABE is defined by the following game:

Initialize: The challenger runs the Setup algorithm of SA-KP-ABE and gives the public parameters PK to the adversary;

Oracles: $\mathsf{OKeyGen}(\tilde{\mathcal{T}})$, $\mathsf{ODelegate}(k, \tilde{\mathcal{T}}')$, $\mathsf{OGetKey}(k)$, $\mathsf{OEncaps}(\Gamma_v, \Gamma_i)$, and

$\mathsf{RoAVEncaps}(\Gamma_v, \Gamma_i)$**:** The adversary submits one Real-or-All-Valid encapsulation query on the distinct sets of attributes (Γ_v, Γ_i). The challenger then generates $(K, C) \leftarrow \mathsf{Encaps}^*(\mathsf{SK}, \Gamma_v, \Gamma_i)$ as the real case, if $b = 0$, or $(K, C) \leftarrow \mathsf{Encaps}(\mathsf{PK}, \Gamma_v \cup \Gamma_i)$ as the all-valid case, if $b = 1$, and outputs C to the adversary;

Finalize(b')**:** The adversary outputs a guess b' for b. If for some access-tree $\tilde{\mathcal{T}}'$ corresponding to a key asked to the OGetKey-oracle, $\tilde{\mathcal{T}}'(\Gamma_v \cup \Gamma_i, \emptyset) = 1$, on the challenge set (Γ_v, Γ_i), $\beta \xleftarrow{\$} \{0,1\}$, else one sets $\beta = b'$. One outputs β.

$\mathsf{Adv}^{\mathsf{att\text{-}ind}}(\mathcal{A})$ denotes the advantage of an adversary $\mathcal{A}$ in this game.

This definition ensures that a user with keys for access-trees that are not satisfied by $\Gamma = \Gamma_v \cup \Gamma_i$ cannot distinguish valid from invalid attributes in the ciphertext.

As above on key-indistinguishability, this first definition excludes accepting policies on the challenge ciphertext. However, for tracing, one also needs to deal with ciphertexts on accepting policies. More precisely, we must allow keys and a challenge ciphertext that would be accepted in the all-valid case, and still have indistinguishability, until attributes associated to the active leaves in the keys and invalid attributes in the challenge ciphertext are **distinct**. Hence, the **Distinct Attribute-Indistinguishability** (dAtt-IND) where **Finalize**(b') reads: *The adversary outputs a guess b' for b. If some attribute $t \in \Gamma_i$ from the challenge query corresponds to some active leaf $\lambda \in \mathcal{L}'_a$ in a* OGetKey*-query, then set $\beta \xleftarrow{\$} \{0,1\}$, otherwise set $\beta = b'$. One outputs β.*

6.4 Our SA-KP-ABE

6.4.1 Construction

We extend the basic KP-ABE scheme proven in 5.2, with leaves that can be made active or passive in a decryption key, and some attributes can be made valid or invalid in a ciphertext, and prove that it still achieves the Del-IND-security. For our construction, we will use two DPVS, of dimensions 3 and 9 respectively, in a pairing-friendly setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$, using the notations introduced in the DPVS section (4). Essentially, we introduce a 7-th component to deal with switchable attributes. The two new basis-vectors $\mathbf{d}_7$ and $\mathbf{d}_7^*$ are in the secret key SK and the master secret key MK respectively. The two additional 8-th and 9-th components are to deal with the unbounded universe of attributes, to be able to use the adaptive Index-Ind property (see Theorem 7), instead of the static one. These additional components are hidden, and for the proof only:

$\mathsf{Setup}(1^\kappa)$**.** The algorithm chooses two random dual orthogonal bases

$$\mathbb{B} = (\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3) \qquad \mathbb{B}^* = (\mathbf{b}_1^*, \mathbf{b}_2^*, \mathbf{b}_3^*) \qquad \mathbb{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_9) \qquad \mathbb{D}^* = (\mathbf{d}_1^*, \ldots, \mathbf{d}_9^*).$$

It sets the public parameters $\mathsf{PK} = \{(\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}_1^*), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_1^*, \mathbf{d}_2^*, \mathbf{d}_3^*)\}$, whereas the master secret key is $\mathsf{MK} = \{\mathbf{b}_3^*, \mathbf{d}_7^*\}$ and the secret key is $\mathsf{SK} = \{\mathbf{d}_7\}$. Other basis vectors are kept hidden.

$\mathsf{KeyGen}(\mathsf{MK}, \tilde{\mathcal{T}})$**.** For an extended access-tree $\tilde{\mathcal{T}} = (\mathcal{T}, \mathcal{L}_a, \mathcal{L}_p)$, the algorithm first chooses a random $a_0 \xleftarrow{\$} \mathbb{Z}_q$, and a random a_0-labeling $(a_\lambda)_\lambda$ of the access-tree $\mathcal{T}$, and builds the key:

$$\mathbf{k}_0^* = (a_0, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}_\lambda^* = (\pi_\lambda(1, t_\lambda), a_\lambda, 0, 0, 0, r_\lambda, 0, 0)_{\mathbb{D}^*}$$

for all the leaves λ, where $t_\lambda = A(\lambda)$, $\pi_\lambda \xleftarrow{\$} \mathbb{Z}_q$, and $r_\lambda \xleftarrow{\$} \mathbb{Z}_q^*$ if λ is an active leaf in the key ($\lambda \in \mathcal{L}_a$) or else $r_\lambda = 0$ for a passive leaf ($\lambda \in \mathcal{L}_p$). The decryption key $\mathsf{dk}_{\tilde{\mathcal{T}}}$ is then $(\mathbf{k}_0^*, (\mathbf{k}_\lambda^*)_\lambda)$.

$\mathsf{Delegate}(\mathsf{dk}_{\tilde{\mathcal{T}}}, \tilde{\mathcal{T}}')$**.** Given a private key for a tree $\tilde{\mathcal{T}}$ and a more restrictive subtree $\tilde{\mathcal{T}}' \leq \tilde{\mathcal{T}}$, the algorithm creates a delegated key $\mathsf{dk}_{\tilde{\mathcal{T}}'}$. It chooses a random $a_0' \xleftarrow{\$} \mathbb{Z}_q$ and a random a_0'-labeling $(a_\lambda')_\lambda$ of $\mathcal{T}'$; Then, it updates $\mathbf{k}_0^* \leftarrow \mathbf{k}_0^* + (a_0', 0, 0)_{\mathbb{B}^*}$; It sets $\mathbf{k}_\lambda^* \leftarrow (\pi_\lambda' \cdot (1, t_\lambda), a_\lambda', 0, 0, 0, 0, 0, 0)_{\mathbb{B}^*}$ for a new leaf, or updates $\mathbf{k}_\lambda^* \leftarrow \mathbf{k}_\lambda^* + (\pi_\lambda' \cdot (1, t_\lambda), a_\lambda', 0, 0, 0, 0, 0, 0)_{\mathbb{B}^*}$ for an old leaf, with $\pi_\lambda' \xleftarrow{\$} \mathbb{Z}_q$.

$\mathsf{Encaps}(\mathsf{PK}, \Gamma)$**.** For a set Γ of attributes, the algorithm first chooses random scalars $\omega, \xi \xleftarrow{\$} \mathbb{Z}_q$. It then sets $K = g_t^\xi$ and generates the ciphertext $C = (\mathbf{c}_0, (\mathbf{c}_t)_{t \in \Gamma})$ where

$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(t, -1), \omega, 0, 0, 0, 0, 0, 0)_{\mathbb{D}}$$

for all the attributes $t \in \Gamma$, with $\sigma_t \xleftarrow{\$} \mathbb{Z}_q$.

$\mathsf{Encaps}^*(\mathsf{SK}, (\Gamma_v, \Gamma_i))$**.** For a disjoint union $\Gamma = \Gamma_v \uplus \Gamma_i$ of sets of attributes (Γ_v is the set of valid attributes and Γ_i is the set of invalid attributes), the algorithm first chooses random scalars $\omega, \xi \xleftarrow{\$} \mathbb{Z}_q$. It then sets $K = g_t^\xi$ and generates the ciphertext $C = (\mathbf{c}_0, (\mathbf{c}_t)_{t \in (\Gamma_v \cup \Gamma_i)})$ where

$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(t, -1), \omega, 0, 0, 0, u_t, 0, 0)_{\mathbb{D}}$$

for all the attributes $t \in \Gamma_v \uplus \Gamma_i$, $\sigma_t \xleftarrow{\$} \mathbb{Z}_q$ and $u_t \xleftarrow{\$} \mathbb{Z}_q^*$ if $t \in \Gamma_i$ or $u_t = 0$ if $t \in \Gamma_v$.

$\mathsf{Decaps}(\mathsf{dk}_{\tilde{\mathcal{T}}}, C)$**.** The algorithm first selects an evaluation pruned tree $\mathcal{T}'$ of $\mathcal{T}$ that is satisfied by $\Gamma = \Gamma_v \cup \Gamma_i$, such that any leaf λ in $\mathcal{T}'$ is either passive in the key ($\lambda \in \mathcal{L}_p$) or associated to a valid attribute in the ciphertext ($t_\lambda \in \Gamma_v$). This means that the labels a_λ for all the leaves λ in $\mathcal{T}'$ allow to reconstruct a_0 by simple additions, where $t = t_\lambda$:

$$\mathbf{c}_t \times \mathbf{k}_\lambda^* = g_t^{\sigma_t \cdot \pi_\lambda \cdot \langle (t,-1),(1,t_\lambda) \rangle + \omega \cdot a_\lambda + u_t \cdot r_\lambda} = g_t^{\omega \cdot a_\lambda},$$

as $u_t = 0$ or $r_\lambda = 0$. Hence, the algorithm can derive $g_t^{\omega \cdot a_0}$. From $\mathbf{c}_0$ and $\mathbf{k}_0^*$, it can also compute $\mathbf{c}_0 \times \mathbf{k}_0^* = g_t^{\omega \cdot a_0 + \xi}$, which then easily leads to $K = g_t^\xi$.

First, note that the delegation works as $\mathbf{b}_1^*$, $\mathbf{d}_1^*, \mathbf{d}_2^*, \mathbf{d}_3^*$ are public. This allows to create a new key for $\tilde{\mathcal{T}}' \leq \tilde{\mathcal{T}}$. But as $\mathbf{d}_7^*$ is not known, any new leaf is necessarily passive, and an active existing leaf in the original key cannot be converted to passive, and vice-versa. Indeed, all the randomnesses are fresh, except for the last components r_λ that remain unchanged: this is perfectly consistent with the definition of $\tilde{\mathcal{T}}' \leq \tilde{\mathcal{T}}$.

Second, in encapsulation with Encaps^*, to invalidate a contribution $\mathbf{c}_t$ in the ciphertext with a non-zero u_t, for $t \in \Gamma_i$, one needs to know $\mathbf{d}_7$, hence SK is required. On the contrary, when using Encaps with $\Gamma_i = \emptyset$, one just needs PK.

Eventually, we stress that in the above decryption, one can recover $g_t^{\omega \cdot a_0}$ if and only if there is an evaluation pruned tree $\mathcal{T}'$ of $\mathcal{T}$ that is satisfied by Γ and the active leaves in $\tilde{\mathcal{T}}'$ correspond to valid attributes in Γ_v (used during the encapsulation). And this holds if and only if $\tilde{\mathcal{T}}(\Gamma_v, \Gamma_i) = 1$.

6.4.2 Security Results

Del-IND-Security of our SA-KP-ABE for Encaps For this security notion, we first consider only valid contributions in the challenge ciphertext, with indistinguishability of the Encaps algorithm. Which means that $\Gamma_i = \emptyset$ in the challenge pair. And the security result holds even if the vector $\mathbf{d}_7$ is made public:

Theorem 12 *Our SA-KP-ABE scheme is Del-IND for* Encaps *(with only valid attributes in the challenge ciphertext), even if* $\mathbf{d}_7$ *is public.*

The proof essentially reduces to the IND-security result of the KP-ABE scheme, and is presented thereafter in 6.4.3.

Although the structure of the proof is the same as the one for the KP-ABE in Section 5.2, we present again an overview of the proof for the reader's convenience. We follow the traditional Dual Encryption System, where the simulator first makes the ciphertext *semi-functional* in games $\mathbf{G}_1$ and $\mathbf{G}_2$. Then, the simulator changes the keys into *semi-functional* keys one by one in a hybrid game, up until all keys are semi-functional in game $\mathbf{G}_3$. Then the sequence of games conclude as the adversary has no information on the encapsulated key in game $\mathbf{G}_4$.

The only difference with the former proof is the existence of vectors $(\mathbf{d}_7, \mathbf{d}_7^*)$. However, we can still reduce to the classic Del-IND proof for KP-ABE as $\mathbf{d}_7$ is public, hence the adversary can run by himself both Encaps and Encaps*. Likewise, since $\mathbf{d}_7^*$ is known to the simulator, delegation can just be done by using the key generation algorithm, making sure we use the same randomness for all the keys delegated from the same one.

The global sequence of games is described on Figure 6.2, where $(\mathbf{c}_0, (\mathbf{c}_t))$ is the challenge ciphertext for all the attributes $t \in \Gamma$, and $(\mathbf{k}^*_{\ell,0}, (\mathbf{k}^*_{\ell,\lambda}))$ are the keys, for $1 \leq \ell \leq K$, and $\lambda \in \mathcal{L}_\ell$ for each ℓ-query, with active and passive leaves.

We stress that our construction makes more basis vectors public than in the schemes from Okamoto and Takashima [OT12b], as only $\mathbf{b}_3^*$ is for the key issuer. This makes the proof more tricky, but this is the reason why we can deal with delegation for any user.

Del-IND-Security of our SA-KP-ABE for Encaps* We now study the full indistinguishability of the ciphertext generated by an Encaps* challenge, with delegated keys. The intuition is that when $u_t \cdot r_{\ell,\lambda} \neq 0$, the share $a_{\ell,\lambda}$ in $g_t^{\omega \cdot a_{\ell,\lambda} + u_t \cdot r_{\ell,\lambda}}$ is hidden, but we have to formally prove it.

The main issue in this proof is the need to anticipate whether $u_t \cdot r_{\ell,\lambda} = 0$ or not when simulating the keys, and the challenge ciphertext as well (even before knowing the exact query (Γ_v, Γ_i)). Intuitively, if the adversary is able to switch an attribute in the ciphertext between valid and invalid, it would leak too much information. Thus the possible sets of valid attributes and invalid attributes must be decided before the game start. Concretely, without being in the selective-set setting where both Γ_v and Γ_i would have to be specified before generating the public parameters PK, we ask to know disjoint super-sets $A_v, A_i \subseteq \mathcal{U}$ of attributes. Then, in the challenge ciphertext query, we will ask that $\Gamma_v \subseteq A_v$ and $\Gamma_i \subseteq A_i$. We will call this setting the *semi-adaptive super-set* setting, where the super-sets have to be specified before the first decryption keys are issued. Furthermore, the set of attributes $\Gamma = \Gamma_v \cup \Gamma_i$ used in the real challenge query is only specified at the moment of the challenge, as in the adaptive setting. We stress that the semi-adaptive super-set setting is much stronger than the selective-set setting where the adversary would have to specify both Γ_v and Γ_i before the setup. Here, only super-sets have to be specified, and just before the first key-query. The adversary is thus given much more power.

For this proof, $\mathbf{d}_7$ must be kept secret (cannot be provided to the adversary). We will thus give access to an Encaps* oracle. We then need to simulate it.

Theorem 13 *Our SA-KP-ABE scheme is Del-IND for* Encaps**, in the semi-adaptive super-set setting (where* $A_v, A_i \subseteq \mathcal{U}$ *so that* $\Gamma_v \subseteq A_v$ *and* $\Gamma_i \subseteq A_i$ *are specified before asking for keys).*

The full proof can be found in 6.4.3. We still give some details here: we only consider keys that are really provided to the adversary, and thus delegated keys. They can be generated as fresh keys except for the r_λ's in the last position that have to be the same for leaves in keys delegated from the same initial key. However, in order to randomize $s_{\ell,0}$ once all of the shares have been masked, one cannot directly conclude that $s_{\ell,0}$ is independent from the view of the adversary: we only know $\tilde{\mathcal{T}}_\ell(\Gamma_v, \Gamma_i) = 0$, but not necessarily $\mathcal{T}_\ell(\Gamma_v \cup \Gamma_i) = 0$, as in the previous proof.

To this aim, we revisit this gap with an additional sequence where we focus on the k-th key and the challenge ciphertext. In that sequence, we first prepare with additional random values $y_{\ell,\lambda}$ in all the keys, with the same repetition properties as the $r_{\ell,\lambda}$. Thereafter, in another subsequence of games on the attributes, we can use the Swap-Ind property to completely randomize $s_{k,\lambda}$ when $u_{t_{k,\lambda}} \cdot r_{k,\lambda} \neq 0$. Hence, the $s_{k,\lambda}$ are unknown either when $z_{t_{k,\lambda}}$ is not known (the corresponding element is not provided in the challenge ciphertext) or this is a random $s'_{k,\lambda}$ when there is enough noise, namely when $u_{t_{k,\lambda}} \cdot r_{k,\lambda} \neq 0$. The property of the access-tree then makes $s_{k,0}$ perfectly unpredictable, which can be replaced by a random independent $r_{k,0}$.

Distinct Indistinguishability Properties We first claim easy results, for which the proofs are symmetrical:

Theorem 14 *Our SA-KP-ABE scheme is dKey-IND, even if* $\mathbf{d}_7^*$ *is public.*

Theorem 15 *Our SA-KP-ABE scheme is dAtt-IND, even if* $\mathbf{d}_7$ *is public.*

Both proofs can be found in 6.4.3. In these alternative variants, all the invalid attributes in all the queried ciphertexts do not correspond to any active leaf in the challenge keys (for dKey-IND) or all active leaves in all the queried keys do not correspond to any invalid attribute in the challenge ciphertext (for dAtt-IND). Then, we can gradually replace all the real keys by all-passive in the former proof or all the real ciphertexts by all-valid in the latter proof.

Attribute-Indistinguishability

Theorem 16 *Our SA-KP-ABE scheme is Att-IND, even if* $\mathbf{d}_7$ *is public, if all the active keys correspond to independent leaves with respect to the set of attributes* $\Gamma = \Gamma_v \cup \Gamma_i$ *in the challenge ciphertext.*

The proof can be found in 6.4.3. This is an important result with respect to our target application of tracing, combined with possible revocation. Indeed, with such a result, if a user is excluded independently of the tracing procedure (the policy would reject him even if all his passive leaves match valid attributes in the ciphertext), he will not be able to detect whether there are invalid attributes in the ciphertext and thus that the ciphertext is from a tracing procedure. This gives us a strong resistance to collusion for tracing.

6.4.3 Security Proofs

Del-IND-Security for Encaps – Proof of Theorem 12

PROOF We will proceed to prove this by a succession of games. At some point, our game will be in the same state as Game $\mathbf{G}_0$ in the proof of IND for the KP-ABE scheme, in Section 5.2, which allows us to conclude. We stress that we use the Adaptive Index-Ind instead of the static version, but this just impacts the way we enumerate the attributes: instead of enumerating all the universe that was polynomial-size, we enumerate them in the order the appear in the security game (either in a policy or in a ciphertext). This will be important for hybrid sequences of games on attributes: t or p will actually be the attributes but also associated to their order number when they appear for the first time in the security game.

Game $\mathbf{G}_0$: The first game is the real game, where the simulator honestly runs the setup, with $\mathsf{PK} = \{(\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}_1^*), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_1^*, \mathbf{d}_2^*, \mathbf{d}_3^*)\}$, $\mathsf{SK} = \{\mathbf{d}_7\}$, and $\mathsf{MK} = \{\mathbf{b}_3^*, \mathbf{d}_7^*\}$, from random dual orthogonal bases.

$\mathsf{OKeyGen}(\tilde{\mathcal{T}}_\ell)$**:** The adversary is allowed to issue KeyGen-queries on an access-tree $\tilde{\mathcal{T}}_\ell$ (for the ℓ-th query), for which the simulator chooses a random scalar $a_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$ and a random $a_{\ell,0}$-labeling $(a_{\ell,\lambda})_\lambda$ of the access-tree $\tilde{\mathcal{T}}_\ell$, and builds the key:

$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}$$

for all the leaves λ, where $t_{\ell,\lambda} = A(\lambda)$ in $\mathcal{T}_\ell$, $\pi_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q$ and $r_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q^*$ if λ is an active leave or $r_{\ell,\lambda} = 0$ if it is passive. The decryption key $\mathsf{dk}_\ell = (\mathbf{k}^*_{\ell,0}, (\mathbf{k}^*_{\ell,\lambda})_\lambda)$ is kept private, and will be used for delegation queries;

$\mathsf{ODelegate}(\tilde{\mathcal{T}}, \tilde{\mathcal{T}}')$**:** The adversary is allowed to issue Delegate-queries for an access-tree $\tilde{\mathcal{T}}'$, of an already queried decryption key with access-tree $\tilde{\mathcal{T}} = \tilde{\mathcal{T}}_\ell$, with the only condition that $\tilde{\mathcal{T}}' \leq \tilde{\mathcal{T}}$. From $\mathsf{dk}_\ell = (k^*_0, (k^*_\lambda)_\lambda)$, for $\lambda \in \mathcal{L}$, then the simulator computes the delegated key as, $\forall \lambda \in \mathcal{L}'$:

$$\mathbf{k}'^*_0 = \mathbf{k}^*_0 + (a'_0, 0, 0)_{\mathbb{B}^*} \qquad \mathbf{k}'^*_\lambda = \mathbf{k}^*_\lambda + (\pi'_\lambda(t_\lambda, -1), a'_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*},$$

where $\mathbf{k}^*_\lambda = (0, 0, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}$ if λ was not in $\mathcal{L}$, and $a'_0 \xleftarrow{\$} \mathbb{Z}_q$ and $(a'_\lambda)_\lambda$ is an a'_0-labeling of $\mathcal{T}'$.

$\mathsf{RoREncaps}(\Gamma_v, \Gamma_i = \emptyset)$**:** On the unique query on a set of attributes $\Gamma = \Gamma_v$, the simulator chooses random scalars $\omega, \xi, \xi' \xleftarrow{\$} \mathbb{Z}_q$. It then sets $K_0 = g_t^{\xi}$ and $K_1 = g_t^{\xi'}$. It generates the ciphertext $C = (\mathbf{c}_0, (\mathbf{c}_t)_{t\in\Gamma})$ where

$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1, t), \omega, 0, 0, 0, 0, 0, 0)_{\mathbb{D}}$$

for all the attributes $t \in \Gamma$ and $\sigma_t \xleftarrow{\$} \mathbb{Z}_q$. According to the real or random game (bit $b \xleftarrow{\$} \{0, 1\}$), one outputs (K_b, C).

From the adversary's guess b' for b, if for some $\tilde{\mathcal{T}}'$ asked as a delegation-query, $\tilde{\mathcal{T}}'(\Gamma_v, \Gamma_i) = 1$, then $\beta \xleftarrow{\$} \{0, 1\}$, otherwise $\beta = b'$. We denote $\mathsf{Adv}_0 = \Pr[\beta = 1|b = 1] - \Pr[\beta = 1|b = 0]$.

We stress that in this game, we deal with delegation queries, but only want to show they do not help to break indistinguishability of the encapsulated keys with the official Encaps algorithm, and not the private Encaps^* one. Hence, $\Gamma_i = \emptyset$ in the challenge ciphertext.

Game $\mathbf{G}_1$: We now show it can be reduced to Game $\mathbf{G}_0$ from the IND security game on the KP-ABE, in the proof provided in Section 5.2. The challenge ciphertext is already exactly the same, as we only consider Encaps. However, we have to simulate the key-generation and key-delegation oracles OKeyGen and ODelegate using only the key-generation oracle from Game $\mathbf{G}_0$

$\mathbf{G}_0$ Real Del-IND-Security game

$$\mathbf{c}_0 = (\omega \quad 0 \quad \xi) \quad \mathbf{c}_t = (\sigma_t(1,t) \quad \omega \mid 0 \quad 0 \quad 0 \quad u_t \mid 0\ 0)$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0} \quad 0 \quad 1) \quad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad 0 \quad r_{\ell,\lambda} \mid 0\ 0)$$

$\mathbf{G}_1$ SubSpace-Ind Property, on $(\mathbb{B},\mathbb{B}^*)_{1,2}$ and $(\mathbb{D},\mathbb{D}^*)_{3,4}$, between 0 and $\tau \xleftarrow{\$} \mathbb{Z}_q$

$$\mathbf{c}_0 = (\omega \quad \tau \quad \xi) \quad \mathbf{c}_t = (\sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad 0 \quad u_t \mid 0\ 0)$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0} \quad 0 \quad 1) \quad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad 0 \quad r_{\ell,\lambda} \mid 0\ 0)$$

$\mathbf{G}_2$ SubSpace-Ind Property, on $(\mathbb{D},\mathbb{D}^*)_{1,2,6}$, between 0 and τz_t

$$\mathbf{c}_0 = (\omega \quad \tau \quad \xi) \quad \mathbf{c}_t = (\sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad \tau z_t \quad u_t \mid 0\ 0)$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0} \quad 0 \quad 1) \quad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad 0 \quad r_{\ell,\lambda} \mid 0\ 0)$$

$\mathbf{G}_3$ Introduction of an additional random-labeling. See Figure 6.3

$$\mathbf{c}_0 = (\omega \quad \tau \quad \xi) \quad \mathbf{c}_t = (\sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad \tau z_t \quad u_t \mid 0\ 0)$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0} \quad r_{\ell,0} \quad 1) \quad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad \frac{s_{\ell,\lambda}}{z_{t_{\ell,\lambda}}} \quad r_{\ell,\lambda} \mid 0\ 0)$$

$\mathbf{G}_4$ Formal basis change, on $(\mathbb{B},\mathbb{B}^*)_{2,3}$, to randomize ξ

$$\mathbf{c}_0 = (\omega \quad \tau \quad \xi'') \quad \mathbf{c}_t = (\sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad \tau z_t \quad u_t \mid 0\ 0)$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0} \quad r_{\ell,0} \quad 1) \quad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad \frac{s_{\ell,\lambda}}{z_{t_{\ell,\lambda}}} \quad r_{\ell,\lambda} \mid 0\ 0)$$

Figure 6.2: Sequence of games on the keys for the Del-IND-security proof of our SA-KP-ABE

$\mathbf{G}_{2.k.0}$ Hybrid game for $\mathbf{G}_2$, with $1 \le k \le K+1$ (from Figure 6.2)

$$\mathbf{c}_0 = (\omega \quad \tau \quad \xi) \quad \mathbf{c}_t = (\sigma_t(1,t) \quad \omega \mid \tau \quad 0 \quad \tau z_t \quad u_t \mid 0\ 0)$$
$\ell < k$ $$\mathbf{k}^*_{\ell,0} = (a_{\ell,0} \quad r_{\ell,0} \quad 1) \quad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad \frac{s_{\ell,\lambda}}{z_{t_{\ell,\lambda}}} \quad r_{\ell,\lambda} \mid 0\ 0)$$
$\ell \ge k$ $$\mathbf{k}^*_{\ell,0} = (a_{\ell,0} \quad 0 \quad 1) \quad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \mid 0 \quad 0 \quad 0 \quad r_{\ell,\lambda} \mid 0\ 0)$$

$\mathbf{G}_{2.k.1}$ SubSpace-Ind Property, on $(\mathbb{B}^*,\mathbb{B})_{1,2}$ and $(\mathbb{D}^*,\mathbb{D})_{3,4}$, between 0 and $s_{k,*}$

$$\mathbf{k}^*_{k,0} = (a_{k,0} \quad s_{k,0} \quad 1) \quad \mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \mid s_{k,\lambda} \quad 0 \quad 0 \quad r_{k,\lambda} \mid 0\ 0)$$

$\mathbf{G}_{2.k.2}$ Masking of the labeling. See Figure 5.3 for Encaps, or Figure 6.4 for Encaps*

$$\mathbf{k}^*_{k,0} = (a_{k,0} \quad s_{k,0} \quad 1) \quad \mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \mid 0 \quad 0 \quad \frac{s_{k,\lambda}}{z_{t_{k,\lambda}}} \quad r_{k,\lambda} \mid 0\ 0)$$

$\mathbf{G}_{2.k.3}$ Limitations on KeyGen-queries: $s_{k,0}$ unpredictable, replaced by a random $r_{k,0}$

$$\mathbf{k}^*_{k,0} = (a_{k,0} \quad r_{k,0} \quad 1) \quad \mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \mid 0 \quad 0 \quad \frac{s_{k,\lambda}}{z_{t_{k,\lambda}}} \quad r_{k,\lambda} \mid 0\ 0)$$

Figure 6.3: Sequence of games on the keys for the Del-IND-security proof of our SA-KP-ABE

in the proof provided in Section 5.2. We call $\mathsf{OKeyGen}'$ the key generation simulator from the classic KP-ABE proof in the SA-KP-ABE game, and note that it only partially generates our new keys, without a 7-th coordinate $r_{\ell,\lambda}$. To keep track of the $r_{\ell,\lambda}$ used in the game, we instantiate a list Λ.

$\mathsf{OKeyGen}(\tilde{\mathcal{T}}_\ell)$**.** The simulator calls the oracle $\mathsf{OKeyGen}'(\mathcal{T}_\ell)$, and chooses $r_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q^*$ or sets $r_{\ell,\lambda} \leftarrow 0$ according to whether $\lambda \in \mathcal{L}_a$ or $\lambda \in \mathcal{L}_p$. It then adds the last component $r_{\ell,\lambda}$ on every $\mathbf{k}^*_{\ell,\lambda}$ using $\mathbf{d}^*_7$ which is known to the simulator. Finally, it updates Λ with a new entry $\Lambda_\ell = (r_{\ell,\lambda})_\lambda$;

$\mathsf{ODelegate}(\tilde{\mathcal{T}}_\ell, \tilde{\mathcal{T}}')$**.** The simulator calls the oracle $\mathsf{OKeyGen}'(\mathcal{T}')$ to get the decryption key dk. As already noted, in the KP-ABE, a delegated key is indistinguishable from a fresh key. Then, we pick the entry $r_{\ell,\lambda}$ from Λ_ℓ, to the last component $r_{\ell,\lambda}$ on every $\mathbf{k}^*_\lambda$ using $\mathbf{d}^*_7$ which is known to the simulator. We stress that for any new leaf, not present in $\tilde{\mathcal{T}}_\ell$ is necessarily passive in the delegated tree $\tilde{\mathcal{T}}'$. So, if a leaf is not in Λ_ℓ, $r_{\ell,\lambda} = 0$.

In this new game, we are exactly using the security game from the IND security on the KP-ABE, and simulating the 7-th component using $\mathbf{d}^*_7$. As this component does not change nor intervene at all in any of the games from the proof in Section 5.2, and this is exactly the same situation as in Game $\mathbf{G}_0$ in that proof, we conclude by following those security games, which leads to the adversary having the same advantage in the last game.

We stress that this simulation of ODelegate will be used in all the following proofs: a delegated key is identical to a fresh key, except the $r_{\ell,\lambda}$ for keys delegated, which is the same value that original key it comes from.

Del-IND-Security for Encaps* – Proof of Theorem 13

Proof The proof will proceed by games, with the first two games following exactly the same sequence as in the previous IND-security proof of the KP-ABE in 5.2, except the RoREncaps-challenge that allows non-empty Γ_i. We also have to consider how to simulate OEncaps-queries on pairs (Γ_v, Γ_i), with $\Gamma_i \neq \emptyset$, meaning there are invalid attributes in the ciphertext. In order to do that, we remind that everything on the 7-th component can be done independently, using both $\mathbf{d}_7$ and $\mathbf{d}^*_7$, as these vectors will be known to the simulator almost all the time, except in some specific gaps. In theses cases, we will have to make sure how to simulate the OEncaps ciphertexts. As explained in the proof, Section 6.4.3, we can simulate ODelegate-queries as OKeyGen-queries, since a delegated key is identical to a fresh key, except the common $r_{\ell,\lambda}$ for keys delegated from the same original key. We thus just have to take care about the way we choose $r_{\ell,\lambda}$.

As in the IND-security proof of the KP-ABE, the idea of the sequence is to introduce an additional labeling $(s_{\ell,0}, (s_{\ell,\lambda})_\lambda)$ in each ℓ-th key (in $\mathbf{G}_{2.k.1}$, from Figure 5.2), where each label is masked by a random z_t for each attribute t (in $\mathbf{G}_{2.k.2}$).

However, in order to go to game $\mathbf{G}_{2.k.3}$, one cannot directly conclude that $s_{k,0}$ is independent from the view of the adversary: we only know $\tilde{\mathcal{T}}_k(\Gamma_v, \Gamma_i) = 0$, but not necessarily $\mathcal{T}_k(\Gamma_v \cup \Gamma_i) = 0$, as in the previous proof. Hence, we revisit this gap with an additional sequence presented in the Figure 6.4 where we focus on the k-th key and the ciphertext, with random $\omega, \tau, \xi, \xi', (\sigma_t), (z_t) \xleftarrow{\$} \mathbb{Z}_q$, but for all the OKeyGen-query, random $a_{\ell,0}, (\pi_{\ell,\lambda}) \xleftarrow{\$} \mathbb{Z}_q$, as well as a random $a_{\ell,0}$-labeling $(a_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_k$, but also $s_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$ and a second independent random $s_{\ell,0}$-labeling $(s_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_k$, and an independent random $r_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$. The goal is to replace each label $s_{k,\lambda}$ by a random independent value $s'_{k,\lambda}$ when $u_{t_{k,\lambda}} \cdot r_{k,\lambda} \neq 0$. As a consequence, we will consider below that $s'_{k,\lambda}$ denotes either the label $s_{k,\lambda}$ when $u_{t_{k,\lambda}} \cdot r_{k,\lambda} = 0$ or a random scalar:

Game $\mathbf{G}_{2.k.2.0}$: The first game is exactly $\mathbf{G}_{2.k.2}$, where the simulator honestly runs the setup, with $\mathsf{PK} = \{(\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}^*_1), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}^*_1, \mathbf{d}^*_2, \mathbf{d}^*_3)\}$, $\mathsf{SK} = \{\mathbf{d}_7\}$, and $\mathsf{MK} = \{\mathbf{b}^*_3, \mathbf{d}^*_7\}$, from random dual orthogonal bases.

$\mathbf{G}_{2.k.2.0}$ Intermediate sequence from $\mathbf{G}_{2.k.2}$ (from Figure 6.3)

$\mathbf{c}_t = (\quad \sigma_t(1,t) \quad \omega \quad | \quad \tau \quad 0 \quad \tau z_t \quad u_t \quad | 0\ 0\ )$

$\ell < k$ $\mathbf{k}^*_{\ell,0} = (\quad a_{\ell,0} \quad r_{\ell,0} \quad 1\ )$

$\mathbf{k}^*_{\ell,\lambda} = (\quad \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \quad | \quad 0 \quad 0 \quad s'_{\ell,\lambda}/z_{t_{\ell,\lambda}} \quad r_{\ell,\lambda} \quad | 0\ 0\ )$

$\ell = k$ $\mathbf{k}^*_{k,0} = (\quad a_{k,0} \quad s_{k,0} \quad 1\ )$

$\mathbf{k}^*_{k,\lambda} = (\quad \pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \quad | \quad 0 \quad 0 \quad s_{k,\lambda}/z_{t_{k,\lambda}} \quad r_{k,\lambda} \quad | 0\ 0\ )$

$\ell > k$ $\mathbf{k}^*_{\ell,0} = (\quad a_{\ell,0} \quad 0 \quad 1\ )$

$\mathbf{k}^*_{\ell,\lambda} = (\quad \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \quad | \quad 0 \quad 0 \quad 0 \quad r_{\ell,\lambda} \quad | 0\ 0\ )$

$s'_{\ell,\lambda}$ is either the label $s_{\ell,\lambda}$ when $r_{\ell,\lambda} \cdot u_{t_{\ell,\lambda}} = 0$, or a random scalar in $\mathbb{Z}_q$ otherwise

$\mathbf{G}_{2.k.2.1}$ SubSpace-Ind Property, on $(\mathbb{D},\mathbb{D}^*)_{4,5}$, between τ and 0

$\mathbf{c}_t = (\quad \sigma_t(1,t) \quad \omega \quad | \quad 0 \quad 0 \quad \tau z_t \quad u_t \quad | 0\ 0\ )$

$\mathbf{G}_{2.k.2.2}$ SubSpace-Ind Property, on $(\mathbb{D}^*,\mathbb{D})_{2,4}$, between 0 and $y_{\ell,\lambda}$

$\mathbf{k}^*_{k,0} = (\quad a_{k,0} \quad s_{k,0} \quad 1\ )$

$\mathbf{k}^*_{k,\lambda} = (\quad \pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \quad | \quad y_{k,\lambda} \quad 0 \quad s_{k,\lambda}/z_{t_{k,\lambda}} \quad r_{k,\lambda} \quad | 0\ 0\ )$

$\mathbf{G}_{2.k.2.3}$ Formal basis change, on $(\mathbb{D},\mathbb{D}^*)_{5,7}$, to duplicate $r_{\ell,\lambda}$

$\ell < k$ $\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \ r_{\ell,0} \ 1 \ ) \ \mathbf{k}^*_{\ell,\lambda} = (\dots | \ y_{\ell,\lambda} \ r_{\ell,\lambda} \ s'_{\ell,\lambda}/z_{t_{\ell,\lambda}} \ r_{\ell,\lambda} \ | 0\ 0\ )$

$\ell = k$ $\mathbf{k}^*_{k,0} = (\ a_{k,0} \ s_{k,0} \ 1 \ ) \ \mathbf{k}^*_{k,\lambda} = (\dots | \ y_{k,\lambda} \ r_{k,\lambda} \ s_{k,\lambda}/z_{t_{k,\lambda}} \ r_{k,\lambda} \ | 0\ 0\ )$

$\ell > k$ $\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \ 0 \ 1 \ ) \ \mathbf{k}^*_{\ell,\lambda} = (\dots | \ y_{\ell,\lambda} \ r_{\ell,\lambda} \ 0 \ r_{\ell,\lambda} \ | 0\ 0\ )$

$\mathbf{G}_{2.k.2.4}$ Alteration of the labeling. See Figure 6.5

$\ell < k$ $\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \ r_{\ell,0} \ 1 \ ) \ \mathbf{k}^*_{\ell,\lambda} = (\dots | \ y_{\ell,\lambda} \ 0 \ s'_{\ell,\lambda}/z_{t_{\ell,\lambda}} \ r_{l,\lambda} \ | 0\ 0\ )$

$\ell = k$ $\mathbf{k}^*_{k,0} = (\ a_{k,0} \ s_{k,0} \ 1 \ ) \ \mathbf{k}^*_{k,\lambda} = (\dots | \ y_{k,\lambda} \ 0 \ s'_{k,\lambda}/z_{t_{k,\lambda}} \ r_{k,\lambda} \ | 0\ 0\ )$

$\ell > k$ $\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \ 0 \ 1 \ ) \ \mathbf{k}^*_{\ell,\lambda} = (\dots | \ y_{\ell,\lambda} \ 0 \ 0 \ r_{\ell,\lambda} \ | 0\ 0\ )$

$\mathbf{G}_{2.k.2.5}$ Limitations on KeyGen-queries: $s_{k,0}$ unpredictable, replaced by a random $r_{k,0}$

$\ell = k$ $\mathbf{k}^*_{k,0} = (\ a_{k,0} \ r_{k,0} \ 1 \ ) \ \mathbf{k}^*_{k,\lambda} = (\dots | \ y_{k,\lambda} \ 0 \ s'_{k,\lambda}/z_{t_{k,\lambda}} \ r_{k,\lambda} \ | 0\ 0\ )$

$\mathbf{G}_{2.k.2.6}$ SubSpace-Ind Property, on $(\mathbb{D}^*,\mathbb{D})_{2,4}$, between $y_{\ell,\lambda}$ and 0

$\mathbf{k}^*_{k,0} = (\quad a_{k,0} \quad r_{k,0} \quad 1\ )$

$\mathbf{k}^*_{k,\lambda} = (\quad \pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \quad | \quad 0 \quad 0 \quad s'_{k,\lambda}/z_{t_{k,\lambda}} \quad r_{k,\lambda} \quad | 0\ 0\ )$

$\mathbf{G}_{2.k.2.7}$ SubSpace-Ind Property, on $(\mathbb{D},\mathbb{D}^*)_{4,5}$, between 0 and τ

$\mathbf{c}_t = (\quad \sigma_t(1,t) \quad \omega \quad | \quad \tau \quad 0 \quad \tau z_t \quad u_t \quad | 0\ 0\ )$

Figure 6.4: Sequence of games on the keys for the Del-IND-security proof of our SA-KP-ABE

$\mathbf{G}_{2.k.2.3.p.0}$ Hybrid game for $\mathbf{G}_{2.k.2.3}$, with $1 \leq p \leq P+1$ (from Figure 6.4)

$$\begin{array}{lll}
 & \mathbf{c}_0 = & (\ \omega \quad \tau \quad \xi\ ) \\
 & \mathbf{c}_t = & (\ \sigma_t(1,t) \quad \omega \quad | \quad 0 \quad 0 \quad \tau z_t \quad u_t \quad |\, 0\ 0\ ) \\
\ell < k & \mathbf{k}^*_{\ell,0} = & (\ a_{\ell,0} \quad r_{\ell,0} \quad 1\ ) \\
 & \mathbf{k}^*_{\ell,\lambda} = & (\ \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \quad | \quad y_{\ell,\lambda} \quad r_{\ell,\lambda} \quad s'_{\ell,\lambda}/z_{t_{\ell,\lambda}} \quad r_{\ell,\lambda} \quad |\, 0\ 0\ ) \\
\ell = k & \mathbf{k}^*_{k,0} = & (\ a_{k,0} \quad s_{k,0} \quad 1\ ) \\
t_{k,\lambda} < p & \mathbf{k}^*_{k,\lambda} = & (\ \pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \quad | \quad y_{k,\lambda} \quad r_{k,\lambda} \quad s'_{k,\lambda}/z_{t_{k,\lambda}} \quad r_{k,\lambda} \quad |\, 0\ 0\ ) \\
t_{k,\lambda} \geq p & \mathbf{k}^*_{k,\lambda} = & (\ \pi_{k,\lambda}(t_{k,\lambda},-1) \quad a_{k,\lambda} \quad | \quad y_{k,\lambda} \quad r_{k,\lambda} \quad s_{k,\lambda}/z_{t_{k,\lambda}} \quad r_{k,\lambda} \quad |\, 0\ 0\ ) \\
\ell > k & \mathbf{k}^*_{\ell,0} = & (\ a_{\ell,0} \quad 0 \quad 1\ ) \\
 & \mathbf{k}^*_{\ell,\lambda} = & (\ \pi_{\ell,\lambda}(t_{\ell,\lambda},-1) \quad a_{\ell,\lambda} \quad | \quad y_{\ell,\lambda} \quad r_{\ell,\lambda} \quad 0 \quad r_{\ell,\lambda} \quad |\, 0\ 0\ )
\end{array}$$

$s'_{\ell,\lambda}$ is either the label $s_{\ell,\lambda}$ when $r_{\ell,\lambda} \cdot u_{t_{\ell,\lambda}} = 0$, or a random scalar in $\mathbb{Z}_q$ otherwise

$\mathbf{G}_{2.k.2.3.p.1}$ Swap-Ind Property, on $(\mathbb{D}, \mathbb{D}^*)_{5,7}$, for 0 and u_p in $\mathbf{c}_p$ only

$$\begin{array}{lll}
\mathbf{c}_p = (\ldots | & 0 \quad u_p \quad \tau z_p \quad 0 \quad |0\ 0) & \mathbf{c}_t = (\ldots | \quad 0 \quad 0 \quad \tau z_t \quad u_t \quad |\, 0\ 0\ )
\end{array}$$

$\mathbf{G}_{2.k.2.3.p.2}$ Index-Ind Property, on $(\mathbb{D}^*, \mathbb{D})_{1,2,5}$, between $r_{\ell,\lambda}$ and 0, for all $t_{\ell,\lambda} \neq p$

$$\begin{array}{lll}
\mathbf{c}_p = (\ldots | & 0 \quad u_p \quad \tau z_p \quad 0 \quad |0\ 0) & \mathbf{c}_t = (\ldots | \quad 0 \quad 0 \quad \tau z_t \quad u_t \quad |\, 0\ 0\ ) \\
 & t_{\ell,\lambda} \neq p, \ell < k & \mathbf{k}^*_{\ell,\lambda} = (\ldots | \quad y_{\ell,\lambda} \quad 0 \quad s'_{\ell,\lambda}/z_{t_{\ell,\lambda}} \quad r_{\ell,\lambda} \quad |\, 0\ 0\ ) \\
\mathbf{k}^*_{\ell,\lambda} = (\ldots | & y_{\ell,\lambda} \quad r_{\ell,\lambda} \quad s'_{\ell,\lambda}/z_p \quad r_{\ell,\lambda} \quad |0\ 0) & t_{\ell,\lambda} = p, \ell < k \\
 & t_{k,\lambda} < p, \ell = k & \mathbf{k}^*_{k,\lambda} = (\ldots | \quad y_{k,\lambda} \quad 0 \quad s'_{k,\lambda}/z_{t_{k,\lambda}} \quad r_{k,\lambda} \quad |\, 0\ 0) \\
\mathbf{k}^*_{k,\lambda} = (\ldots | & y_{k,\lambda} \quad r_{k,\lambda} \quad s_{k,\lambda}/z_p \quad r_{k,\lambda} \quad |0\ 0) & t_{k,\lambda} = p, \ell = k \\
 & t_{k,\lambda} > p, \ell = k & \mathbf{k}^*_{k,\lambda} = (\ldots | \quad y_{k,\lambda} \quad 0 \quad s_{k,\lambda}/z_{t_{k,\lambda}} \quad r_{k,\lambda} \quad |\, 0\ 0\ ) \\
 & t_{\ell,\lambda} \neq p, \ell > k & \mathbf{k}^*_{\ell,\lambda} = (\ldots | \quad y_{\ell,\lambda} \quad 0 \quad 0 \quad r_{\ell,\lambda} \quad |\, 0\ 0\ ) \\
\mathbf{k}^*_{\ell,\lambda} = (\ldots | & y_{\ell,\lambda} \quad r_{\ell,\lambda} \quad 0 \quad r_{\ell,\lambda} \quad |0\ 0) & t_{\ell,\lambda} = p, \ell > k
\end{array}$$

$\mathbf{G}_{2.k.2.3.p.3}$ Formal change of basis on column 5, multiplying ciphertext by $\tau z_p / u_p$

$$\begin{array}{lll}
\mathbf{c}_p = (\ldots | & 0 \quad \tau z_p \quad \tau z_p \quad 0 \quad |0\ 0) & \mathbf{c}_t = (\ldots | \quad 0 \quad 0 \quad \tau z_t \quad u_t \quad |\, 0\ 0\ ) \\
\mathbf{k}^*_{\ell,\lambda} = (\ldots | & y_{\ell,\lambda} \quad r_{\ell,\lambda} u_p/\tau z_p \quad s'_{\ell,\lambda}/z_p \quad r_{\ell,\lambda} \quad |0\ 0) & t_{\ell,\lambda} = p, \ell < k \\
\mathbf{k}^*_{k,\lambda} = (\ldots | & y_{k,\lambda} \quad r_{k,\lambda} u_p/\tau z_p \quad s_{k,\lambda}/z_p \quad r_{k,\lambda} \quad |0\ 0) & t_{k,\lambda} = p, \ell = k \\
\mathbf{k}^*_{\ell,\lambda} = (\ldots | & y_{\ell,\lambda} \quad r_{\ell,\lambda} u_p/\tau z_p \quad 0 \quad r_{\ell,\lambda} \quad |0\ 0) & t_{\ell,\lambda} = p, \ell > k
\end{array}$$

$\mathbf{G}_{2.k.2.3.p.4}$ Index-Ind Property, on $(\mathbb{D}, \mathbb{D}^*)_{1,2,5}$, between 0 and τz_t, for $t \neq p$

$$\begin{array}{lll}
\mathbf{c}_p = (\ldots | & 0 \quad \tau z_p \quad \tau z_p \quad 0 \quad |0\ 0) & \mathbf{c}_t = (\ldots | \quad 0 \quad \tau z_t \quad \tau z_t \quad u_t \quad |\, 0\ 0\ )
\end{array}$$

$\mathbf{G}_{2.k.2.3.p.5}$ Swap-Ind Property, on $(\mathbb{D}^*, \mathbb{D})_{4,5,6}$, between $r_{k,\lambda} u_p / \tau z_p$ and 0, for $t_{k,\lambda} = p$ only

$$\begin{array}{lll}
\mathbf{c}_p = (\ldots | & 0 \quad \tau z_p \quad \tau z_p \quad 0 \quad |0\ 0) & \mathbf{c}_t = (\ldots | \quad 0 \quad \tau z_t \quad \tau z_t \quad u_t \quad |\, 0\ 0\ ) \\
\mathbf{k}^*_{\ell,\lambda} = (\ldots | & y_{\ell,\lambda} \quad r_{\ell,\lambda} u_p/\tau z_p \quad s'_{\ell,\lambda}/z_p \quad r_{\ell,\lambda} \quad |0\ 0) & t_{\ell,\lambda} = p, \ell < k \\
\mathbf{k}^*_{k,\lambda} = (\ldots | & y_{k,\lambda} \quad 0 \quad \frac{s_{k,\lambda} + r_{k,\lambda} u_p/\tau}{z_p} \quad r_{k,\lambda} \quad |0\ 0) & t_{k,\lambda} = p, \ell = k \\
\mathbf{k}^*_{\ell,\lambda} = (\ldots | & y_{\ell,\lambda} \quad r_{\ell,\lambda} u_p/\tau z_p \quad 0 \quad r_{\ell,\lambda} \quad |0\ 0) & t_{\ell,\lambda} = p, \ell > k
\end{array}$$

Figure 6.5: Sequence of sub-games on the attributes for the Del-IND-security proof of our SA-KP-ABE (Cont'ed on Figure 6.5)

$\mathbf{G}_{2.k.2.3.p.6}$ SubSpace-Ind Property, on $(\mathbb{D}^*, \mathbb{D})_{4,7}$, to randomize $r_{k,\lambda}$ for $t_{k,\lambda} = p$

$\mathbf{c}_p = (\dots \mid$	0	τz_p	τz_p	0	$\mid 0\ 0)$	$\mathbf{c}_t = (\dots \mid$	0	τz_t	τz_t	u_t	$\mid 0\ 0)$
$t_{\ell,\lambda} \neq p, \ell < k$						$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	0	$s'_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	$\mid 0\ 0)$
$t_{k,\lambda} < p, \ell = k$						$\mathbf{k}^*_{k,\lambda} = (\dots \mid$	$y_{k,\lambda}$	0	$s'_{k,\lambda}/z_{t_{k,\lambda}}$	$r_{k,\lambda}$	$\mid 0\ 0)$
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}u_p/\tau z_p$	$s'_{\ell,\lambda}/z_p$	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell < k$					
$\mathbf{k}^*_{k,\lambda} = (\dots \mid$	$y_{k,\lambda}$	0	$s'_{k,\lambda}/z_p$	$r'_{k,\lambda}$	$\mid 0\ 0)$	$t_{k,\lambda} = p, \ell = k$					
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}u_p/\tau z_p$	0	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell > k$					
$t_{k,\lambda} > p, \ell = k$						$\mathbf{k}^*_{k,\lambda} = (\dots \mid$	$y_{k,\lambda}$	0	$s_{k,\lambda}/z_{t_{k,\lambda}}$	$r_{k,\lambda}$	$\mid 0\ 0)$
$t_{\ell,\lambda} \neq p, \ell > k$						$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	0	0	$r_{\ell,\lambda}$	$\mid 0\ 0)$

$\mathbf{G}_{2.k.2.3.p.7}$ Swap-Ind Property, on $(\mathbb{D}^*, \mathbb{D})_{4,5,6}$, between 0 and $r'_{k,\lambda}u_p/\tau z_p$, for $t_{k,\lambda} = p$ only

$\mathbf{c}_p = (\dots \mid$	0	τz_p	τz_p	0	$\mid 0\ 0)$	$\mathbf{c}_t = (\dots \mid$	0	τz_t	τz_t	u_t	$\mid 0\ 0)$
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}u_p/\tau z_p$	$s'_{\ell,\lambda}/z_p$	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell < k$					
$\mathbf{k}^*_{k,\lambda} = (\dots \mid$	$y_{k,\lambda}$	$r'_{k,\lambda}u_p/\tau z_p$	$\frac{s'_{k,\lambda} - r'_{k,\lambda}u_p/\tau}{z_p}$	$r'_{k,\lambda}$	$\mid 0\ 0)$	$t_{k,\lambda} = p, \ell = k$					
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}u_p/\tau z_p$	0	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell > k$					

$\mathbf{G}_{2.k.2.3.p.8}$ Index-Ind Property, on $(\mathbb{D}, \mathbb{D}^*)_{1,2,5}$, between τz_t and 0

$\mathbf{c}_p = (\dots \mid$	0	τz_p	τz_p	0	$\mid 0\ 0)$	$\mathbf{c}_t = (\dots \mid$	0	0	τz_t	u_t	$\mid 0\ 0)$
$t_{\ell,\lambda} \neq p, \ell < k$						$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	0	$s'_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	$\mid 0\ 0)$
$t_{k,\lambda} < p, \ell = k$						$\mathbf{k}^*_{k,\lambda} = (\dots \mid$	$y_{k,\lambda}$	0	$s'_{k,\lambda}/z_{t_{k,\lambda}}$	$r_{k,\lambda}$	$\mid 0\ 0)$
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}u_p/\tau z_p$	$s'_{\ell,\lambda}/z_p$	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell \leq k$					
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}u_p/\tau z_p$	0	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell > k$					
$t_{k,\lambda} > p, \ell = k$						$\mathbf{k}^*_{k,\lambda} = (\dots \mid$	$y_{k,\lambda}$	0	$s_{k,\lambda}/z_{t_{k,\lambda}}$	$r_{k,\lambda}$	$\mid 0\ 0)$
$t_{\ell,\lambda} \neq p, \ell > k$						$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	0	0	$r_{\ell,\lambda}$	$\mid 0\ 0)$

$\mathbf{G}_{2.k.2.3.p.9}$ Formal change of basis

$\mathbf{c}_p = (\dots \mid$	0	u_p	τz_p	0	$\mid 0\ 0)$	$\mathbf{c}_t = (\dots \mid$	0	0	τz_t	u_t	$\mid 0\ 0)$
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}$	$s'_{\ell,\lambda}/z_p$	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell \leq k$					
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}$	0	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell > k$					

$\mathbf{G}_{2.k.2.3.p.10}$ Index-Ind Property, on $(\mathbb{D}^*, \mathbb{D})_{1,2,5}$, between 0 and $r_{\ell,\lambda}$, for all $t_{\ell,\lambda} \neq p$

$\mathbf{c}_p = (\dots \mid$	0	u_p	τz_p	0	$\mid 0\ 0)$	$\mathbf{c}_t = (\dots \mid$	0	0	τz_t	u_t	$\mid 0\ 0)$
$t_{\ell,\lambda} \neq p, \ell < k$						$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}$	$s'_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	$\mid 0\ 0)$
$t_{k,\lambda} < p, \ell = k$						$\mathbf{k}^*_{k,\lambda} = (\dots \mid$	$y_{k,\lambda}$	$r_{k,\lambda}$	$s'_{k,\lambda}/z_{t_{k,\lambda}}$	$r_{k,\lambda}$	$\mid 0\ 0)$
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}$	$s'_{\ell,\lambda}/z_p$	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell \leq k$					
$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	0	$r_{\ell,\lambda}$	0	$r_{\ell,\lambda}$	$\mid 0\ 0)$	$t_{\ell,\lambda} = p, \ell > k$					
$t_{k,\lambda} > p, \ell = k$						$\mathbf{k}^*_{k,\lambda} = (\dots \mid$	$y_{k,\lambda}$	$r_{k,\lambda}$	$s_{k,\lambda}/z_{t_{k,\lambda}}$	$r_{k,\lambda}$	$\mid 0\ 0)$
$t_{\ell,\lambda} \neq p, \ell > k$						$\mathbf{k}^*_{\ell,\lambda} = (\dots \mid$	$y_{\ell,\lambda}$	$r_{\ell,\lambda}$	0	$r_{\ell,\lambda}$	$\mid 0\ 0)$

$\mathbf{G}_{2.k.2.3.p.11}$ Swap-Ind Property, on $(\mathbb{D}, \mathbb{D}^*)_{5,7}$, for 0 and u_p

$\mathbf{c}_p = (\dots \mid$	0	0	τz_p	u_p	$\mid 0\ 0)$	$\mathbf{c}_t = (\dots \mid$	0	0	τz_t	u_t	$\mid 0\ 0)$

Figure 6.5: Sequence of sub-games on the attributes for the Del-IND-security proof of our SA-KP-ABE (Cont'ed)

$\mathsf{OKeyGen}(\mathcal{T}_\ell)$ (or $\mathsf{ODelegate}$-queries): The simulator builds the ℓ-th key:

$$\begin{array}{ll} \ell < k & \mathbf{k}^*_{\ell,0} = (a_{\ell,0}, r_{\ell,0}, 1)_{\mathbb{B}^*} \\ & \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, s'_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \\ \ell = k & \mathbf{k}^*_{k,0} = (a_{k,0}, s_{k,0}, 1)_{\mathbb{B}^*} \\ & \mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda}, -1), a_{k,\lambda}, 0, 0, s_{k,\lambda}/z_{t_{k,\lambda}}, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*} \\ \ell > k & \mathbf{k}^*_{\ell,0} = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} \\ & \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \end{array}$$

with $r_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q$ if $\lambda \in \mathcal{L}_a$ or $r_{\ell,\lambda} = 0$ if $\lambda \in \mathcal{L}_p$. The decryption key dk_ℓ is then $(\mathbf{k}^*_{\ell,0}, (\mathbf{k}^*_{\ell,\lambda})_\lambda)$.

$\mathsf{OEncaps}(\Gamma_{m,v}, \Gamma_{m,i})$**:** The simulator builds the m-th ciphertext using all the known vectors of the basis:

$$\mathbf{c}_{m,0} = (\omega_m, 0, \xi_m)_{\mathbb{B}} \qquad \mathbf{c}_{m,t} = (\sigma_{m,t}(t, -1), \omega_m, 0, 0, 0, u_{m,t}, 0, 0)_{\mathbb{D}}$$

with $\omega_m, \xi_m \xleftarrow{\$} \mathbb{Z}_q$, $\sigma_{m,t} \xleftarrow{\$} \mathbb{Z}_q$ and $u_{m,t} \xleftarrow{\$} \mathbb{Z}_q^*$ if $t \in \Gamma_{m,i}$ or $u_{m,t} \leftarrow 0$ if $t \in \Gamma_{m,v}$. The ciphertext C_m is then $(\mathbf{c}_{m,0}, (\mathbf{c}_{m,t})_t)$;

$\mathsf{RoREncaps}(\Gamma_v, \Gamma_i)$**:** On the unique query on a set of attributes $(\Gamma_v \cup \Gamma_i)$, the simulator generates the ciphertext $C = (\mathbf{c}_0, (\mathbf{c}_t)_{t \in (\Gamma_v \cup \Gamma_i)})$ where

$$\mathbf{c}_0 = (\omega, \tau, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1, t), \omega, \tau, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}}$$

for all the attributes $t \in (\Gamma_v \cup \Gamma_i)$, with $u_t \xleftarrow{\$} \mathbb{Z}_q$ if $t \in \Gamma_i$ or $u_t = 0$ if $t \in \Gamma_v$. According to the real or random game (bit $b \xleftarrow{\$} \{0, 1\}$), one outputs (K_b, C).

From the adversary's guess b' for b, if for some $\tilde{\mathcal{T}}$, $\tilde{\mathcal{T}}(\Gamma_v, \Gamma_i) = 1$, then $\beta \xleftarrow{\$} \{0, 1\}$, otherwise $\beta = b'$. We denote $\mathsf{Adv}_{2.k.2.0} = \Pr[\beta = 1 | b = 1] - \Pr[\beta = 1 | b = 0]$. The goal of this sequence of games is to replace $s_{k,0}$, that can be derived by an acceptable set of $s_{k,\lambda}$, by a random and independent value $r_{k,0}$, in the key generated during the k-th $\mathsf{OKeyGen}$-query.

Indeed, to be a legitimate attack (that does not randomize the adversary's guess b'), for all the key queries $\tilde{\mathcal{T}}_\ell$, one must have $\tilde{\mathcal{T}}_\ell(\Gamma_v, \Gamma_i) = 0$. In particular, $\tilde{\mathcal{T}}_k(\Gamma_v, \Gamma_i) = 0$: this means that

- either the regular access-tree policy is not met, *i.e.*, $\mathcal{T}_k(\Gamma_v \cup \Gamma_i) = 0$.
- or the regular access-tree policy is met, but one active key leaf matches one invalid ciphertext attribute: $\forall \mathcal{T}' \in \mathsf{EPT}(\mathcal{T}_k, \Gamma_v \cup \Gamma_i), \exists \lambda \in \mathcal{T}' \cap \mathcal{L}_a, A(\lambda) \in \Gamma_i$, and from the assumptions, for any such tree $\mathcal{T}'$, the active leave is an independent leave.

In both cases, we will use the same technique to show $s_{k,0}$ is independent from any other value. But first, we will replace all the active leaves associated to invalid ciphertexts in the challenge ciphertext by inactive leaves.

Of course, in the following sequence, we will have to take care of the simulation of the challenge ciphertext, but also of the $\mathsf{OEncaps}$-oracle. For the latter, we will have to clarify how we do the simulation when public vectors $(\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3)$ or the private vector $\mathbf{d}_7$ are impacted.

Game $\mathbf{G}_{2.k.2.1}$: In this game, we first clean the 4-th column of the ciphertext from the τ. To this aim, we are given a tuple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$ in $\mathbb{G}_1$, where $c = ab + \mu \bmod q$ with either

$\mu = 0$ or $\mu = \tau$ (fixed from $\mathbf{c}_0$). When we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 7 respectively, one considers the matrices:

$$D = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{3,4} \qquad D' = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{3,4} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^* \qquad \mathbb{D} = D \cdot \mathbb{V}$$

We can calculate all vectors but $\mathbf{d}_3^*$. Hence, there is no problem for simulating the $\mathsf{OEncaps}$-queries. For the challenge ciphertext, we exploit the DSDH assumption:

$$\begin{aligned}\mathbf{c}_t &= (\sigma_t(1,t), b, c, 0, \tau z_t, u_t, 0, 0)_{\mathbb{V}} = (\sigma_t(1,t), b, c - ab, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} \\ &= (\sigma_t(1,t), b, \mu, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}}\end{aligned}$$

which is correct, with $\omega = b$ and according to μ, this is either τ, as in the previous game or 0 as in this game. For the keys, one notes that the 4-th component is 0, and so the change of basis has no impact on the 3-rd component, when using basis $\mathbb{V}^*$:

$$\mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, \ldots)_{\mathbb{V}^*} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, \ldots)_{\mathbb{D}^*}$$

Then, we have $\mathsf{Adv}_{2.k.2.0} - \mathsf{Adv}_{2.k.2.1} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

Game $\mathbf{G}_{2.k.2.2}$: In this game, we can now introduce noise in the 4-th column the keys. In order to properly deal with delegated keys, as for $r_{\ell,\lambda}$ that have to be the same values for all the leaves delegated from the same initial key, we will also set the same random $y_{\ell,\lambda}$. To this aim, we are given a tuple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$ in $\mathbb{G}_2$, where $c = ab + \zeta \bmod q$ with either $\zeta = 0$ or $\zeta \xleftarrow{\$} \mathbb{Z}_q^*$. We choose additional random scalars $\alpha_{\ell,\lambda}, \beta_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q$ (but the same $\alpha_{\ell,\lambda}$ for all the leaves delegated from the same initial key), to virtually set $b_{\ell,\lambda} = \alpha_{\ell,\lambda} \cdot b + \beta_{\ell,\lambda}$ and $c_{\ell,\lambda} = \alpha_{\ell,\lambda} \cdot c + \beta_{\ell,\lambda} \cdot a$, then $c_{\ell,\lambda} - ab_{\ell,\lambda} = \zeta \cdot \alpha_{\ell,\lambda}$, which are either 0 or independent random values. When we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 7 respectively, one considers the matrices:

$$D = \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix}_{2,4} \qquad D' = \begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix}_{2,4} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^* \qquad \mathbb{D} = D \cdot \mathbb{V}$$

We can calculate all vectors but $\mathbf{d}_4$, which is not used anywhere. Then, for the keys, we exploit the DDH assumption:

$$\begin{aligned}\mathbf{k}^*_{\ell,\lambda} &= (b_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, c_{\ell,\lambda}, \ldots)_{\mathbb{V}^*} = (b_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, c_{\ell,\lambda} - ab_{\ell,\lambda}, \ldots)_{\mathbb{D}^*} \\ &= (b_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, \zeta \cdot \alpha_{\ell,\lambda}, \ldots)_{\mathbb{D}^*}\end{aligned}$$

Which is either the previous game, with $\pi_{\ell,\lambda} = b_{\ell,\lambda}$, when $\zeta = 0$, or the current game with $y_{\ell,\lambda} = \zeta \cdot \alpha_{\ell,\lambda}$ (the same random $y_{\ell,\lambda}$ for all the leaves delegated from the same initial key): $\mathsf{Adv}_{2.k.2.1} - \mathsf{Adv}_{2.k.2.2} \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{2.k.2.3}$: In this game, we duplicate every $r_{\ell,\lambda}$ into the 5-th column of the key. To this aim, one defines the matrices

$$D = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}_{5,7} \qquad D' = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}_{5,7} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^* \qquad \mathbb{D} = D \cdot \mathbb{V}$$

which only modifies $\mathbf{d}_5$, which is hidden, and $\mathbf{d}_7^*$, which is secret, so the change is indistinguishable for the adversary. One can compute the keys and ciphertexts as follows, for all leaves λ of each query ℓ of the adversary:

$$\begin{aligned}\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, 0, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{V}^*} \\ &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{V}} = (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}}\end{aligned}$$

As the 5-th component in the ciphertext is 0, the change of basis makes no change. And this is the same for the ciphertexts generated by the OEncaps-simulation. Hence, the perfect indistinguishability between the two games: $\mathsf{Adv}_{2.k.2.3} = \mathsf{Adv}_{2.k.2.2}$.

Game $\mathbf{G}_{2.k.2.4}$: In this game, we target the k-th OKeyGen-query, and replace $s_{k,\lambda}$ by an independent $s'_{k,\lambda}$ for all the active leaves that correspond to an invalid attribute in the challenge ciphertext. For the sake of simplicity, $s'_{\ell,\lambda}$ is either the label $s_{\ell,\lambda}$ when $r_{\ell,\lambda} \cdot u_{t_{\ell,\lambda}} = 0$, or a random independent scalar in $\mathbb{Z}_q$:

$$\mathbf{k}^*_{k,0} = (a_{k,0}, s_{k,0}, 1)_{\mathbb{B}^*}$$
$$\mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda}, -1), a_{k,\lambda}, y_{k,\lambda}, 0, s'_{k,\lambda}/z_{t_{k,\lambda}}, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*}$$

But to this aim, we will need an additional sequence of sub-games $\mathbf{G}_{2.k.2.3.p.*}$, that will operate iteratively on each attribute p, to convert $\mathbf{G}_{2.k.2.3}$ into $\mathbf{G}_{2.k.2.4}$, as presented in the Figure 6.5. But we first complete the first sequence, and details the sub-sequence afterwards.

Game $\mathbf{G}_{2.k.2.5}$: For the k-th key query, one can now replace $s_{k,0}$ by $r_{k,0}$. Indeed, as explained in the Remark 1, for missing ciphertexts in the challenge ciphertext, the associated leaves in the key have unpredictable $s_{k,\lambda}$. In addition, for active leaves that correspond to invalid attributes in the challenge ciphertext, $s_{k,\lambda}$ have been transformed into $s'_{k,\lambda}$, random independent values. Then, we can consider that all the leaves associated to attributes not in Γ are false, but also active leaves associated to attributes in Γ_i are false. As $\widetilde{\mathcal{T}}_k(\Gamma_v, \Gamma_i) = 0$, the root label is unpredictable. One thus generates the k-th key query as:

$$\mathbf{k}^*_{k,0} = (a_{k,0}, r_{k,0}, 1)_{\mathbb{B}^*}$$
$$\mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda}, -1), a_{k,\lambda}, y_{k,\lambda}, 0, s'_{k,\lambda}/z_{t_{k,\lambda}}, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*}$$

Game $\mathbf{G}_{2.k.2.6}$: We can now invert the above step, when we added $y_{\ell,\lambda}$: $\mathsf{Adv}_{2.k.2.5} - \mathsf{Adv}_{2.k.2.6} \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{2.k.2.7}$: We can now invert the above step, when we removed τ from the ciphertext: $\mathsf{Adv}_{2.k.2.6} - \mathsf{Adv}_{2.k.2.7} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

We now detail the sub-sequence starting from $\mathbf{G}_{2.k.2.3.p.0}$ to prove the indistinguishability between $\mathbf{G}_{2.k.2.3}$ and $\mathbf{G}_{2.k.2.4}$. In the new hybrid game $\mathbf{G}_{2.k.2.3.p.0}$, the critical point will be the p-th ciphertext, where, when $p = 1$, this is exactly the above Game $\mathbf{G}_{2.k.2.3}$, and when $p = P + 1$, this is the above Game $\mathbf{G}_{2.k.2.4}$. And it will be clear, for any p, that $\mathbf{G}_{2.k.2.3.p.11} = \mathbf{G}_{2.k.2.3.p+1.0}$.

With random $\omega, \tau, \xi, \xi', (\sigma_t), (z_t) \xleftarrow{\$} \mathbb{Z}_q$, but for all the OKeyGen-query, random $a_{\ell,0}, (y_\lambda)$, $(\pi_{\ell,\lambda}) \xleftarrow{\$} \mathbb{Z}_q$, as well as a random $a_{\ell,0}$-labeling $(a_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_k$, but also $s_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$ and a second independent random $s_{\ell,0}$-labeling $(s_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_k$, and an independent random $r_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$:

Game $\mathbf{G}_{2.k.2.3.p.0}$: One defines the hybrid game for p:

$$\mathbf{k}^*_{k,0} = (a_{k,0}, s_{k,0}, 1)_{\mathbb{B}^*}$$
$$t_{k,\lambda} < p \qquad \mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda}, -1), a_{k,\lambda}, y_{k,\lambda}, r_{k,\lambda}, s'_{k,\lambda}/z_{t_{k,\lambda}}, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*}$$
$$t_{k,\lambda} \geq p \qquad \mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(t_{k,\lambda}, -1), a_{k,\lambda}, y_{k,\lambda}, r_{k,\lambda}, s_{k,\lambda}/z_{t_{k,\lambda}}, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*}$$

where $s'_{\ell,\lambda}$ is either the label $s_{\ell,\lambda}$ when $r_{\ell,\lambda} \cdot u_{t_{\ell,\lambda}} = 0$, or a random independent scalar in $\mathbb{Z}_q$ (when this is an active leaf that corresponds to an invalid ciphertext).

So one can note that if at the challenge query $p \in \Gamma_v$, then $u_p = 0$, and so we can jump to $\mathbf{G}_{2.k.2.3.p.11}$, but we do not know it before the challenge-query is asked, whereas we have to simulate the keys. This is the reason why we need to know the super sets A_v and A_i: the challenge ciphertext is anticipated with $u_p = 0$ if $p \in A_v$ or with $u_p \xleftarrow{\$} \mathbb{Z}_q^*$ if $p \in A_i$.

Game $\mathbf{G}_{2.k.2.3.p.1}$: The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_1$: one essentially uses theorem 4. Given a tuple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$ in $\mathbb{G}_1$, where $c = ab + \mu \bmod q$ with either $\mu = 0$ or $\mu = u_p$, the 7-th component of the p-th ciphertext. When we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 7 respectively, one considers the matrices:

$$D = \begin{pmatrix} 1 & a & -a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{1,5,7} \qquad D' = \begin{pmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ a & 0 & 1 \end{pmatrix}_{1,5,7} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

We can calculate all vectors but $\mathbf{d}^*_5$ and $\mathbf{d}^*_7$, which are not in the public key. Through $\mathbb{V}$, we calculate the challenge ciphertext for the attribute of the p-th ciphertext

$$\begin{aligned} \mathbf{c}_p &= (0,0,\omega,0,0,\tau z_p, u_p, 0, 0)_{\mathbb{D}} + (b(1,p), 0, 0, c, 0, -c, 0, 0)_{\mathbb{V}} \\ &= (0,0,\omega,0,0,\tau z_p, u_p, 0, 0)_{\mathbb{D}} + (b(1,p), 0, 0, c - ab, 0, ab - c, 0, 0)_{\mathbb{D}} \\ &= (b(1,p), \omega, 0, \mu, \tau z_p, u_p - \mu, 0, 0)_{\mathbb{D}} \end{aligned}$$

If $\mu = 0$, we are in the previous game. If $\mu = u_p$, then we are in the current game. Then, every other ciphertext is computed directly in $\mathbb{D}$:

$$\forall t \neq p, \mathbf{c}_t = (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}}$$

as well as the answers to OEncaps-queries. The keys are calculated through $\mathbb{V}^*$ but are unchanged by the change of basis because the 5-th and 7-th components are exactly the same for every key query ℓ, and thus cancel themselves in the 1st component. We thus have $\mathsf{Adv}_{2.k.2.3.p.0} - \mathsf{Adv}_{2.k.2.3.p.1} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

Game $\mathbf{G}_{2.k.2.3.p.2}$: We keep the $r_{\ell,\lambda}$ value (at the 5-th hidden position) in the keys such that $t_{\ell,\lambda} = p$, and replace it in all other keys by 0, in order to prepare the possibility to later modify the ciphertexts on this component. To show this is possible without impacting the other vectors, we use the Index-Ind property from Theorem 7, but in another level of sequence of hybrid games, for $\gamma \in \{1, \ldots, P\} \backslash \{p\}$. We will enumerate γ in their order of appearance in the security game (wether in key queries, or ciphertexts), therefore we can treat an unbounded number of γ.

Game $\mathbf{G}_{2.k.2.3.p.1.\gamma}$: We consider the following hybrid game, where the first satisfied condition on the indices is applied:

$$\begin{aligned} \mathbf{c}_p &= (\sigma_p(1,p), \omega, 0, u_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && t \neq p \end{aligned}$$

$$\begin{aligned} \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && t_{\ell,\lambda} = p \\ \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, 0, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && p \neq t_{\ell,\lambda} < \gamma \\ \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && p \neq t_{\ell,\lambda} \geq \gamma \end{aligned}$$

where $s^*_{\ell,\lambda}$ is either $s'_{\ell,\lambda}$, $s_{\ell,\lambda}$, or 0:

$$\begin{aligned} s^*_{\ell,\lambda} &= s'_{\ell,\lambda} && \text{if } \ell < k, \text{ or } \ell = k, t_{k,\lambda} < p \\ s^*_{\ell,\lambda} &= s_{\ell,\lambda} && \text{if } \ell = k, t_{k,\lambda} \geq p \\ s^*_{\ell,\lambda} &= 0 && \text{if } \ell > k \end{aligned}$$

When $\gamma = 1$, this is the previous game: $\mathbf{G}_{2.k.2.3.p.1.1} = \mathbf{G}_{2.k.2.3.p.1}$, whereas with $\gamma = P+1$, this is the current game: $\mathbf{G}_{2.k.2.3.p.1.P+1} = \mathbf{G}_{2.k.2.3.p.2}$.
We will gradually replace the $r_{\ell,\lambda}$ values, at the 5-th hidden position, by 0 (when $t_{\ell,\lambda} \neq p$): in this game, we deal with the case $t_{\ell,\lambda} = \gamma$, for all the ℓ-th keys.
For this, we use the Adaptive Index-Ind property on $(\mathbb{D}, \mathbb{D}^*)_{1,2,5,8,9}$, with:

$$\begin{aligned} \mathbf{c}_p &= (\sigma_p(1,p), \omega, 0, u_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\ \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \qquad t_{\ell,\lambda} = \gamma \end{aligned}$$

With all this sequence, we have $\mathsf{Adv}_{2.k.2.3.p.1} - \mathsf{Adv}_{2.k.2.3.p.2} \leq 2P \cdot (8 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t))$.

Game $\mathbf{G}_{2.k.2.3.p.3}$: The last game (in bases $(\mathbb{U}, \mathbb{U}^*, \mathbb{V}, \mathbb{V}^*)$) and this game (in bases $(\mathbb{B}, \mathbb{B}^*, \mathbb{D}, \mathbb{D}^*)$) are perfectly indistinguishable by using a formal change of basis, on hidden vectors, with

$$D = \begin{pmatrix} \frac{\tau z_p}{u_p} \end{pmatrix}_5 \qquad D' = \begin{pmatrix} \frac{u_p}{\tau z_p} \end{pmatrix}_5 \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

The challenge ciphertext and keys that are impacted become:

$$\begin{aligned} \mathbf{c}_p &= (\sigma_p(1,p), \omega, 0, u_p, \tau z_p, 0, 0, 0)_{\mathbb{V}} \\ &= (\sigma_p(1,p), \omega, 0, \tau z_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\ \forall \ell, t_{\ell,\lambda} = p, \quad \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(p, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s^*_{\ell,\lambda}/z_p, r_{\ell,\lambda}, 0, 0)_{\mathbb{V}^*} \\ &= (\pi_{\ell,\lambda}(p, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda} u_p/\tau z_p, s^*_{\ell,\lambda}/z_p, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

All the other vectors have a zero in these components (included the OEncaps-ciphertexts). Hence, $\mathsf{Adv}_{2.k.2.3.p.3} = \mathsf{Adv}_{2.k.2.3.p.2}$. Note however this is because of this game the security result requires the semi-adaptive super-set setting: the change of basis needs to know that $u_p \neq 0$.

Game $\mathbf{G}_{2.k.2.3.p.4}$: We keep the τz_p value (at the 5-th hidden position) in the ciphertext for the p-th attribute only, and replace all the other values from 0 to τz_t, which is the same value as in the 6-th component of each ciphertext, to allow a later swap of the key elements from the 6-th component to the 5-th:

$$\begin{aligned} \mathbf{c}_p &= (\sigma_t(1,t), \omega, 0, \tau z_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, \tau z_t, \tau z_t, u_t, 0, 0)_{\mathbb{D}} \qquad t \neq p \end{aligned}$$

To show this is possible without impacting the other vectors, we use the Index-Ind property from Theorem 7, but in another level of sequence of hybrid games, for $\gamma \in \{1, \ldots, P\} \backslash \{p\}$. We will enumerate γ in their order of appearance in the security game (wether in key queries, or in ciphertexts), therefore we can treat an unbounded number of γ.

Game $\mathbf{G}_{2.k.2.3.p.4.\gamma}$: We consider

$$\begin{aligned} \mathbf{c}_p &= (\sigma_p(1,p), \omega, 0, \tau z_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, \tau z_t, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && p \neq t < \gamma \\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && p \neq t \geq \gamma \\ \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && t_{\ell,\lambda} = p \\ \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, 0, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && t_{\ell,\lambda} \neq p \end{aligned}$$

When $\gamma = 1$, this is the previous game: $\mathbf{G}_{2.k.2.3.p.4.1} = \mathbf{G}_{2.k.2.3.p.3}$, whereas with $\gamma = P+1$, this is the current game: $\mathbf{G}_{2.k.2.3.p.4.P+1} = \mathbf{G}_{2.k.2.3.p.4}$.
For this, we use the Adaptive Index-Ind property on $(\mathbb{D}^*, \mathbb{D})_{1,2,5,8,9}$, with:

$$\begin{aligned} \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && t_{\ell,\lambda} = p \\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && t = \gamma \end{aligned}$$

With all this sequence, we have $\mathsf{Adv}_{2.k.2.3.p.3} - \mathsf{Adv}_{2.k.2.3.p.4} \leq 2P \cdot (8 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t))$.

Game $\mathbf{G}_{2.k.2.3.p.5}$: All ciphertexts now have exactly the same value in 5-th and 6-th positions. We will thus use $r_{\ell,\lambda}$ in the 5-th position, for keys with $t_{\ell,\lambda} = p$, to modify the 6-th position of said keys with a swap. The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_2$: one essentially uses theorem 4. We consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \zeta \bmod q$ with either $\zeta = 0$ or $\zeta = u_p/\tau z_p$, which are indistinguishable under the DSDH assumption. When we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 7 respectively, one considers the matrices:

$$D = \begin{pmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ a & 0 & 1 \end{pmatrix}_{1,5,6} \qquad D' = \begin{pmatrix} 1 & a & -a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{1,5,6} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

We can calculate all vectors but $\mathbf{d}_5$ and $\mathbf{d}_6$, which are not in the public key: However the challenge ciphertext computation through $\mathbb{V}$ is trivial since the 5-th and 6-th components cancel each other out. We can thus simulate them in $\mathbb{D}$.

For challenge ciphertexts, we set

$$\begin{aligned} \mathbf{c}_p &= (\sigma_t(1,t), \omega, 0, \tau z_p, \tau z_p, 0, 0, 0)_{\mathbb{V}} = (\sigma_t(1,t), \omega, 0, \tau z_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, \tau z_t, \tau z_t, u_t, 0, 0)_{\mathbb{V}} = (\sigma_t(1,t), \omega, 0, \tau z_t, \tau z_t, u_t, 0, 0)_{\mathbb{D}} \qquad t \neq p \end{aligned}$$

The only keys that are calculated through $\mathbb{V}^*$ are the ones from the k-th query so that $t_{k,\lambda} = p$. We choose additional random scalars $\beta_{k,\lambda} \xleftarrow{\$} \mathbb{Z}_q$, to virtually set $b_{k,\lambda} = r_{k,\lambda} \cdot b + \beta_{k,\lambda}$ and $c_{k,\lambda} = r_{k,\lambda} \cdot c + \beta_{k,\lambda} \cdot a$, then $c_{k,\lambda} - ab_{k,\lambda} = \zeta \cdot r_{k,\lambda}$, which is either 0 or $r_{k,\lambda} \cdot u_p/\tau z_p$.

$$\begin{aligned} \mathbf{k}^*_{k,\lambda} &= (0, 0, a_{k,\lambda}, y_{k,\lambda}, 0, 0, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*} \\ &\quad + (b(p,-1), 0, 0, 0, r_{k,\lambda} \cdot u_p/\tau z_p - c_{k,\lambda}, c_{k,\lambda} + s_{k,\lambda}/z_p, 0, 0, 0)_{\mathbb{V}^*} \\ \mathbf{k}^*_{k,\lambda} &= (0, 0, a_{k,\lambda}, y_{k,\lambda}, 0, 0, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*} \\ &\quad + (b(p,-1), 0, 0, b_{k,\lambda}, r_{k,\lambda} \cdot u_p/\tau z_p - (c_{k,\lambda} - ab_{k,\lambda}), \\ &\qquad (c_{k,\lambda} - ab_{k,\lambda}) + s_{k,\lambda}/z_p, 0, 0, 0)_{\mathbb{D}^*} \\ \mathbf{k}^*_{k,\lambda} &= (b(p,-1), a_{k,\lambda}, y_{k,\lambda}, r_{k,\lambda} \cdot u_p/\tau z_p - \zeta \cdot r_{k,\lambda}, \zeta \cdot r_{k,\lambda} + \frac{s_{k,\lambda}}{z_p}, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

If $\zeta = 0$, we are in the previous game. If $\zeta = u_p/\tau z_p$, then $\zeta \cdot r_{k,\lambda} = r_{k,\lambda} \cdot u_p/\tau z_p$ and we are in the current game. All other keys are unchanged and calculated through $\mathbb{D}^*$ directly, without any change. And, $\mathsf{Adv}_{2.k.2.3.p.4} - \mathsf{Adv}_{2.k.2.3.p.5} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{2.k.2.3.p.6}$: In this game, we want to replace $r_{k,\lambda}$ when $t_{k,\lambda} = p$ by a random value in the 7-th column, independently of the value in the 6-th column, so that this 6-th column value can be really random and independent from other values. We will exploit the random $y_{k,\lambda}$ in the 4-th column: We consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \zeta \bmod q$ with either $\zeta = 0$ or $\zeta \xleftarrow{\$} \mathbb{Z}_q^*$, which are indistinguishable under the DDH assumption. We choose additional random scalars $\alpha_\lambda, \beta_\lambda \xleftarrow{\$} \mathbb{Z}_q$, to virtually set $b_\lambda = \alpha_\lambda \cdot b + \beta_\lambda$ and $c_\lambda = \alpha_\lambda \cdot c + \beta_\lambda \cdot a$, then $c_\lambda - ab_\lambda = \zeta \cdot \alpha_\lambda$, which are either 0 or independent random values. When we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 7 respectively, one considers the matrices:

$$D = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{4,7} \qquad D' = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{4,7} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

We can calculate all vectors but $\mathbf{d}_7$, which is not in the public key. Through $\mathbb{V}$, we calculate the challenge ciphertext, and the OEncaps-answers, when the 7-th component is non-zero, as the 0 value of the 4-th component does not impact the 7-th during the change of basis.

On the other hand, all the keys can be directly generated in $\mathbb{D}^*$, except $\mathbf{k}_{k,\lambda}$ when $t_{k,\lambda} = p$, for which we use the DDH assumption:

$$\begin{aligned}\mathbf{k}^*_{k,\lambda} &= (\pi_{k,\lambda}(p,-1), a_{k,\lambda}, 0, 0, \frac{s_{k,\lambda} + r_{k,\lambda}\cdot u_p/\tau}{z_p}, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*}\\ &\quad + (0,0,0,b_\lambda,0,0,c_\lambda,0,0)_{\mathbb{V}^*}\\ &= (\pi_{k,\lambda}(p,-1), a_{k,\lambda}, 0, 0, \frac{s_{k,\lambda} + r_{k,\lambda}\cdot u_p/\tau}{z_p}, r_{k,\lambda}, 0, 0)_{\mathbb{D}^*}\\ &\quad + (0,0,0,b_\lambda,0,0,c_\lambda - ab_\lambda,0,0)_{\mathbb{D}^*}\\ &= (\pi_{k,\lambda}(p,-1), a_{k,\lambda}, b_\lambda, 0, \frac{s_{k,\lambda} + r_{k,\lambda}\cdot u_p/\tau}{z_p}, r_{k,\lambda} + \zeta\cdot\alpha_\lambda, 0, 0)_{\mathbb{V}^*}\end{aligned}$$

When $\zeta = 0$, this is the previous game, with $y_{k,\lambda} = b_\lambda$, when $t_{k,\lambda} = p$. Whereas when $\zeta \xleftarrow{\$} \mathbb{Z}_q^*$, $r'_{k,\lambda} = r_{k,\lambda} + \zeta\cdot\alpha_\lambda$ is independent of $r_{k,\lambda}$, which makes $s'_{k,\lambda} = (s_{k,\lambda} + r_{k,\lambda}\cdot u_p/\tau)/z_p$ independent of $s_{k,\lambda}$ when $r_{k,\lambda}\cdot u_p \neq 0$. Then, $\mathsf{Adv}_{2.k.2.3.p.5} - \mathsf{Adv}_{2.k.2.3.p.6} \leq \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

In order to keep the same $r_{\ell,\lambda}$ for all the leaves delegated from the same initial key, we also apply this additional vector $(0,0,0,b_\lambda,0,0,c_\lambda)_{\mathbb{V}^*}$. This will also keep the same $y_{\ell,\lambda}$ for all these leaves.

Game $\mathbf{G}_{2.k.2.3.p.7}$: All ciphertexts have exactly the same value in 5-th and 6-th positions. We will thus use the Swap-Ind property to revert the change made in game $\mathbf{G}_{2.k.2.3.p.5}$, with the notable difference we are now working with $r'_{k,\lambda}$ (which has just been randomized) instead of $r_{k,\lambda}$, for keys with $t_{k,\lambda} = p$. We are thus not restoring the initial $s_{k,\lambda}$ but we get a truly random value $s'_{k,\lambda}$. The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_2$: one essentially uses theorem 4. We consider a triple $(a\cdot G_2, b\cdot G_2, c\cdot G_2)$, where $c = ab + \zeta \bmod q$ with either $\zeta = 0$ or $\zeta = u_p/\tau z_p$, which are indistinguishable under the DSDH assumption. When we start from random dual orthogonal bases $(\mathbb{U},\mathbb{U}^*)$ and $(\mathbb{V},\mathbb{V}^*)$ of size 3 and 7 respectively, one considers the matrices:

$$D = \begin{pmatrix} 1 & 0 & 0\\ -a & 1 & 0\\ a & 0 & 1\end{pmatrix}_{4,5,6} \qquad D' = \begin{pmatrix} 1 & a & -a\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}_{4,5,6} \qquad \mathbb{D} = D\cdot\mathbb{V} \qquad \mathbb{D}^* = D'\cdot\mathbb{V}^*$$

We can calculate all vectors but $\mathbf{d}_5$ and $\mathbf{d}_6$, which are not in the public key: However, the challenge ciphertext computation through $\mathbb{V}$ is trivial since the 5-th and 6-th component cancel each other out. We can thus simulate them through $\mathbb{V}$. We can revert as above by setting in $\mathbb{V}^*$ the keys from the k-th query so that $t_{k,\lambda} = p$. And, $\mathsf{Adv}_{2.k.2.3.p.6} - \mathsf{Adv}_{2.k.2.3.p.7} \leq 2\cdot\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$. We stress that after the swap, we get, for $t_{k,\lambda} = p$

$$\mathbf{k}^*_{k,\lambda} = (\pi_{k,\lambda}(p,-1), a_{k,\lambda}, y_{k,\lambda}, r'_{k,\lambda}u_p/\tau z_p, (s'_{k,\lambda} - r'_{k,\lambda}u_p/\tau)/z_p, r'_{k,\lambda}, 0, 0)_{\mathbb{D}^*}$$

where $s'_{k,\lambda}$ is a truly random value independent of $r'_{k,\lambda}$. So we are not back to game $\mathbf{G}_{2.k.2.3.p.4}$, but still with a random value in the 6-th component of the key.

Game $\mathbf{G}_{2.k.2.3.p.8}$: We keep the τz_p value (at the 5-th hidden position) in the ciphertext for the p-th attribute only, and replace all the other values from τz_t to 0

$$\begin{aligned}\mathbf{c}_p &= (\sigma_t(1,t), \omega, 0, \tau z_p, \tau z_p, 0, 0, 0)_{\mathbb{D}}\\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} \qquad\qquad t \neq p\end{aligned}$$

To show this is possible without impacting the other vectors, we use the Index-Ind property from Theorem 7, but in another level of sequence of hybrid games, for $\gamma \in \{1,\dots,P\}\backslash\{p\}$. We will enumerate γ in their order of appearance in the security game (wether in key queries, or in ciphertexts), therefore we can treat an unbounded number of γ.

Game $\mathbf{G}_{2.k.2.3.p.8.\gamma}$: We consider

$$\begin{aligned}
\mathbf{c}_p &= (\sigma_p(1,p), \omega, 0, \tau z_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && p \neq t < \gamma \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, \tau z_t, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && p \neq t \geq \gamma \\
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r'_{\ell,\lambda}, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r'_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && t_{\ell,\lambda} = p \\
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, 0, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && t_{\ell,\lambda} \neq p
\end{aligned}$$

When $\gamma = 1$, this is the previous game: $\mathbf{G}_{2.k.2.3.p.8.1} = \mathbf{G}_{2.k.2.3.p.7}$, whereas with $\gamma = P+1$, this is the current game: $\mathbf{G}_{2.k.2.3.p.8.P+1} = \mathbf{G}_{2.k.2.3.p.8}$.
For this, we use the Adaptive Index-Ind property on $(\mathbb{D}^*, \mathbb{D})_{1,2,5,8,9}$, with:

$$\begin{aligned}
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r'_{\ell,\lambda}, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r'_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && t_{\ell,\lambda} = p \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, \tau z_t, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && t = \gamma
\end{aligned}$$

With all this sequence, we have $\mathsf{Adv}_{2.k.2.3.p.7} - \mathsf{Adv}_{2.k.2.3.p.8} \leq 2P(8 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t))$.

Game $\mathbf{G}_{2.k.2.3.p.9}$: The last game (in bases $(\mathbb{U}, \mathbb{U}^*, \mathbb{V}, \mathbb{V}^*)$) and this game (in bases $(\mathbb{B}, \mathbb{B}^*, \mathbb{D}, \mathbb{D}^*)$) are perfectly indistinguishable by using a formal change of basis, on hidden vectors, with

$$D = \begin{pmatrix} \frac{u_p}{\tau z_p} \end{pmatrix}_5 \qquad D' = \begin{pmatrix} \frac{\tau z_p}{u_p} \end{pmatrix}_5 \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

The challenge ciphertext and keys that are impacted become:

$$\begin{aligned}
\mathbf{c}_p &= (\sigma_p(1,p), \omega, 0, \tau z_p, \tau z_p, 0, 0, 0)_{\mathbb{V}} \\
&= (\sigma_p(1,p), \omega, 0, u_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\
\forall \ell, t_{\ell,\lambda} = p, \quad \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(p, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda} \cdot u_p/\tau z_p, s^*_{\ell,\lambda}/z_p, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \\
&= (\pi_{\ell,\lambda}(p, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s^*_{\ell,\lambda}/z_p, r_{\ell,\lambda}, 0, 0)_{\mathbb{V}^*}
\end{aligned}$$

All the other vectors have a zero in these components (included the OEncaps-ciphertexts). Hence, $\mathsf{Adv}_{2.k.2.3.p.9} = \mathsf{Adv}_{2.k.2.3.p.8}$.

Game $\mathbf{G}_{2.k.2.3.p.10}$: We keep the $r'_{\ell,\lambda}$ value (at the 5-th hidden position) in the keys such that $t_{\ell,\lambda} = p$, and replace back the 0 in all other keys by $r_{\ell,\lambda}$, in order to prepare the possibility to later modify the ciphertexts on this component. To show this is possible without impacting the other vectors, we use the Index-Ind property from Theorem 7, but in another level of sequence of hybrid games, for $\gamma \in \{1,\dots,P\}\backslash\{p\}$. We will enumerate γ in their order of appearance in the security game (wether in key queries, or in ciphertexts), therefore we can treat an unbounded number of γ.

Game $\mathbf{G}_{2.k.2.3.p.9.\gamma}$: We consider the following hybrid game, where the first satisfied condition on the indices is applied:

$$\mathbf{c}_p = (\sigma_p(1,p), \omega, 0, u_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \qquad \mathbf{c}_t = (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} \qquad t \neq p$$

$$\begin{aligned}
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && t_{\ell,\lambda} = p \\
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, r_{\ell,\lambda}, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && p \neq t_{\ell,\lambda} < \gamma \\
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, 0, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && p \neq t_{\ell,\lambda} \geq \gamma
\end{aligned}$$

where $s^*_{\ell,\lambda}$ is either $s'_{\ell,\lambda}$, $s_{\ell,\lambda}$, or 0

When $\gamma = 1$, this is the previous game: $\mathbf{G}_{2.k.2.3.p.9.1} = \mathbf{G}_{2.k.2.3.p.9}$, whereas with $\gamma = P+1$, this is the current game: $\mathbf{G}_{2.k.2.3.p.9.P+1} = \mathbf{G}_{2.k.2.3.p.10}$.

We will gradually replace the 0 values, at the 5-th hidden position, by $r_{\ell,\lambda}$ (when $t_{\ell,\lambda} \neq p$): in this game, we deal with the case $t_{\ell,\lambda} = \gamma$, for all the ℓ-th keys.

For this, we use the Adaptive Index-Ind property on $(\mathbb{D}, \mathbb{D}^*)_{1,2,5,8,9}$, with:

$$\begin{aligned} \mathbf{c}_p &= (\sigma_p(1,p), \omega, 0, u_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\ \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, y_{\ell,\lambda}, 0, s^*_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \qquad t_{\ell,\lambda} = \gamma \end{aligned}$$

With all this sequence, we have $\mathsf{Adv}_{2.k.2.3.p.9} - \mathsf{Adv}_{2.k.2.3.p.10} \leq 2P \cdot (8 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t))$.

Game $\mathbf{G}_{2.k.2.3.p.11}$: The previous game and this game are indistinguishable under the DDH assumption in $\mathbb{G}_1$: one essentially uses theorem 4. Given a tuple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$ in $\mathbb{G}_1$, where $c = ab + \mu \bmod q$ with either $\mu = 0$ or $\mu = u_p$, the 5-th component of the p-th ciphertext. When we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 7 respectively, one considers the matrices:

$$D = \begin{pmatrix} 1 & a & -a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{1,5,7} \qquad D' = \begin{pmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ a & 0 & 1 \end{pmatrix}_{1,5,7} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

We can calculate all vectors but $\mathbf{d}^*_5$ and $\mathbf{d}^*_7$, which are not in the public key. Through $\mathbb{V}$, we calculate the challenge ciphertext for the attribute of the p-th ciphertext

$$\begin{aligned} \mathbf{c}_p &= (0,0,\omega,0,0,\tau z_p, u_p)_{\mathbb{D}} + (b(1,p),0,0,c,0,-c,0,0)_{\mathbb{V}} \\ &= (0,0,\omega,0,0,\tau z_p, u_p)_{\mathbb{D}} + (b(1,p),0,0,c-ab,0,ab-c,0,0)_{\mathbb{D}} \\ &= (b(1,p),\omega,0,\mu,\tau z_p, u_p - \mu, 0, 0)_{\mathbb{D}} \end{aligned}$$

If $\mu = u_p$, we are in the previous game. If $\mu = 0$, then we are in the current game. Then, every other ciphertext is computed directly in $\mathbb{D}$:

$$\forall t \neq p, \mathbf{c}_t = (\sigma_t(1,t), \omega, 0, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}}$$

as well as the answers to OEncaps-queries. The keys are calculated through $\mathbb{V}^*$ but are unchanged by the change of basis because the 5-th and 7-th components are exactly the same for every key query ℓ, and thus cancel themselves in the 1st component. We thus have $\mathsf{Adv}_{2.k.2.3.p.10} - \mathsf{Adv}_{2.k.2.3.p.11} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

dKey-IND-Security – Proof of Theorem 14

Proof In this security game, the adversary has access to the OEncaps-oracle, but only for distinct key-indistinguishability: all the invalid attributes $t \in \Gamma_{m,i}$ in a OEncaps-query correspond to passive leaves $\lambda \in \mathcal{L}_p$ from the challenge key. We will prove it as usual with a sequence of games:

Game $\mathbf{G}_0$: The first game is the real game where the simulator plays the role of the challenger, with $\mathsf{PK} = \{(\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}^*_1), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}^*_1, \mathbf{d}^*_2, \mathbf{d}^*_3, \mathbf{d}^*_7)\}$, $\mathsf{SK} = \{\mathbf{d}_7\}$, and $\mathsf{MK} = \{\mathbf{b}^*_3\}$, from random dual orthogonal bases. We note that $\mathbf{d}^*_7$ can be public.

$\mathsf{OKeyGen}(\tilde{\mathcal{T}}_\ell)$ (or ODelegate-queries): The adversary is allowed to issue KeyGen-queries on an access-tree $\tilde{\mathcal{T}}_\ell = (\mathcal{T}_\ell, \mathcal{L}_{\ell,a}, \mathcal{L}_{\ell,p})$ (for the ℓ-th query), for which the simulator chooses a

random scalar $a_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$ and a random $a_{\ell,0}$-labeling $(a_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_\ell$, and builds the key:

$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}$$

for all the leaves λ, where $t_{\ell,\lambda} = A(\lambda)$, $\pi_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q$ and $r_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q^*$ if $\lambda \in \mathcal{L}_{\ell,a}$, or else $r_{\ell,\lambda} \leftarrow 0$ if $\lambda \in \mathcal{L}_{\ell,p}$. The decryption key dk_ℓ is then $(\mathbf{k}^*_{\ell,0}, (\mathbf{k}^*_{\ell,\lambda})_\lambda)$;

$\mathsf{OEncaps}(\Gamma_{m,v}, \Gamma_{m,i})$**:** The adversary is allowed to issue Encaps^*-queries on disjoint unions $\Gamma_m = \Gamma_{m,v} \uplus \Gamma_{m,i}$ of sets of attributes, for which the simulator chooses random scalars $\omega_m, \xi_m \xleftarrow{\$} \mathbb{Z}_q$. It sets $K_m = g_t^{\xi_m}$ and generates a ciphertext $C_m = (\mathbf{c}_{m,0}, (\mathbf{c}_{m,t})_{t \in (\Gamma_{m,v} \cup \Gamma_{m,i})})$ where

$$\mathbf{c}_{m,0} = (\omega_m, 0, \xi_m)_{\mathbb{B}} \qquad \mathbf{c}_{m,t} = (\sigma_{m,t}(t, -1), \omega_m, 0, 0, 0, u_{m,t}, 0, 0)_{\mathbb{D}}$$

for all the attributes $t \in \Gamma_{m,v} \uplus \Gamma_{m,i}$, $\sigma_{m,t} \xleftarrow{\$} \mathbb{Z}_q$ and $u_{m,t} \xleftarrow{\$} \mathbb{Z}_q^*$ if $t \in \Gamma_{m,i}$ or $u_{m,t} \leftarrow 0$ if $t \in \Gamma_{m,v}$.

$\mathsf{RoAPKeyGen}(\tilde{\mathcal{T}}, \mathcal{L}_a, \mathcal{L}_p)$**:** On the unique query on an access-tree $\tilde{\mathcal{T}}$ of its choice, with a list $\mathcal{L} = (\mathcal{L}_a \uplus \mathcal{L}_p)$ of active and passive leaves, the simulator chooses a random scalar $a_0 \xleftarrow{\$} \mathbb{Z}_q$, and a random a_0-labeling $(a_\lambda)_\lambda$ of the access-tree. It then sets the real key dk_0 as follows, with $r_\lambda \xleftarrow{\$} \mathbb{Z}_q^*$ if $\lambda \in \mathcal{L}_a$, or $r_\lambda \leftarrow 0$ if $\lambda \in \mathcal{L}_p$:

$$\mathbf{k}^*_0 = (a_0, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_\lambda = (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, r_\lambda, 0, 0)_{\mathbb{D}^*}$$

On the other hand, it sets the all-passive key dk_1 as:

$$\mathbf{k}^*_0 = (a_0, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_\lambda = (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}$$

for all λ. According to the real or all-passive ($b \xleftarrow{\$} \{0, 1\}$), one outputs dk_b.

From the adversary's guess b' for b, one forwards it as the output β, unless for some $(\Gamma_{m,v}, \Gamma_{m,i})$ asked to the $\mathsf{OEncaps}$-oracle, some active leaf $\lambda \in \mathcal{L}_a$ from the challenge key corresponds to some invalid attribute $t \in \Gamma_{m,i}$, in which case one outputs a random $\beta \xleftarrow{\$} \{0, 1\}$. We denote $\mathsf{Adv}_0 = \Pr[\beta = 1 | b = 1] - \Pr[\beta = 1 | b = 0]$.

We stress that in this distinct key-indistinguishability security game, the active keys in the challenge key ($\lambda \in \mathcal{L}_a$ with possibly $r_\lambda \neq 0$) correspond to valid ciphertexts only ($t \in \Gamma_{m,i}$ with $u_{m,t} = 0$, for all queries). But we do not exclude accepting access-trees.

Game $\mathbf{G}_1$: In the second and final game, we set $r_\lambda \leftarrow 0$ for all the leaves in the real key dk_0:

$$\mathbf{k}^*_0 = (a_0, 0, 0)_{\mathbb{B}^*} \qquad \mathbf{k}^*_\lambda = (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}$$

It is then clear than $\mathsf{Adv}_1 = 0$, as all challenge keys are independent from b.

We detail the sub-sequence starting from $\mathbf{G}_{0.p.0}$ to prove the indistinguishability between $\mathbf{G}_0$ and $\mathbf{G}_1$. In the new hybrid sequence $\mathbf{G}_{0.p.*}$, we will modify all the keys associated to the p-th attribute, in an indistinguishable way, using the $\mathsf{Index\text{-}Ind}$ property. It is clear that $\mathbf{G}_{0.1.0} = \mathbf{G}_0$, whereas $\mathbf{G}_{0.P+1.0} = \mathbf{G}_1$, and $\mathbf{G}_{0.p.4} = \mathbf{G}_{0.p+1.0}$.

Game $\mathbf{G}_{0.p.0}$: One defines the hybrid game for p :

$$\begin{aligned} & \mathbf{c}_{m,t} = (\sigma_{m,t}(1, t), \omega_m, 0, 0, 0, u_{m,t}, 0, 0)_{\mathbb{D}} \\ t_\lambda < p \quad & \mathbf{k}^*_\lambda = (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*} \\ t_\lambda \geq p \quad & \mathbf{k}^*_\lambda = (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, r_\lambda, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

$$\mathbf{c}_0 = \quad (\omega \quad 0 \quad \xi) \qquad \mathbf{k}^*_{\ell,0} = \quad (a_{\ell,0} \quad 0 \quad 1)$$

$\mathbf{G}_{0.p.0}$ Hybrid game for $\mathbf{G}_0$, with $1 \leq p \leq P+1$, such that $u_{m,p} = 0$ for all m

$$\begin{array}{lrcccc|cccc|c} & \mathbf{c}_{m,t} = & (& \sigma_{m,t}(1,t) & \omega_m & | & 0 & 0 & u_{m,t} & | & 0\ 0\ ) \\ t_\lambda < p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & 0 & | & 0\ 0\ ) \\ t_\lambda \geq p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & r_\lambda & | & 0\ 0\ ) \end{array}$$

$\mathbf{G}_{0.p.1}$ Formal basis change, on $(\mathbb{D}, \mathbb{D}^*)_{6,7}$, to duplicate $u_{m,t}$ in the 6-th column

$$\begin{array}{lrcccc|cccc|c} & \mathbf{c}_{m,t} = & (& \sigma_{m,t}(1,t) & \omega_m & | & 0 & 0 & u_{m,t} & u_{m,t} & | & 0\ 0\ ) \\ t_\lambda < p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & 0 & 0 & | & 0\ 0\ ) \\ t_\lambda \geq p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & 0 & r_\lambda & | & 0\ 0\ ) \end{array}$$

$\mathbf{G}_{0.p.2}$ Swap-Ind Property, on $(\mathbb{D}, \mathbb{D}^*)_{1,6,7}$, to swap r_λ, for $t_\lambda = p$, in the 6-th column

$$\begin{array}{lrcccc|cccc|c} & \mathbf{c}_{m,t} = & (& \sigma_{m,t}(1,t) & \omega_m & | & 0 & 0 & u_{m,t} & u_{m,t} & | & 0\ 0\ ) \\ t_\lambda < p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & 0 & 0 & | & 0\ 0\ ) \\ t_\lambda = p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(p,-1) & a_\lambda & | & 0 & 0 & r_\lambda & 0 & | & 0\ 0\ ) \\ t_\lambda > p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & 0 & r_\lambda & | & 0\ 0\ ) \end{array}$$

$\mathbf{G}_{0.p.3}$ Index-Ind Property, on $(\mathbb{D}, \mathbb{D}^*)_{1,2,6}$, between $u_{m,t}$ and 0, for $t \neq p$

$$\begin{array}{lrcccc|cccc|c} & \mathbf{c}_{m,p} = & (& \sigma_{m,t}(1,t) & \omega_m & | & 0 & 0 & 0 & 0 & | & 0\ 0\ ) \\ t \neq p & \mathbf{c}_{m,t} = & (& \sigma_{m,t}(1,t) & \omega_m & | & 0 & 0 & 0 & u_{m,t} & | & 0\ 0\ ) \\ t_\lambda < p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & 0 & 0 & | & 0\ 0\ ) \\ t_\lambda = p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(p,-1) & a_\lambda & | & 0 & 0 & r_\lambda & 0 & | & 0\ 0\ ) \\ t_\lambda > p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & 0 & r_\lambda & | & 0\ 0\ ) \end{array}$$

$\mathbf{G}_{0.p.4}$ SubSpace-Ind Property, on $(\mathbb{D}^*, \mathbb{D})_{1,6}$, between u_p and 0

$$\begin{array}{lrcccc|cccc|c} & \mathbf{c}_{m,t} = & (& \sigma_{m,t}(1,t) & \omega_m & | & 0 & 0 & 0 & u_{m,t} & | & 0\ 0\ ) \\ t_\lambda < p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & 0 & 0 & | & 0\ 0\ ) \\ t_\lambda = p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(p,-1) & a_\lambda & | & 0 & 0 & 0 & 0 & | & 0\ 0\ ) \\ t_\lambda > p & \mathbf{k}^*_\lambda = & (& \pi_\lambda(t_\lambda,-1) & a_\lambda & | & 0 & 0 & 0 & r_\lambda & | & 0\ 0\ ) \end{array}$$

Figure 6.6: Sub-sequence of games for Distinct Key-Indistinguishability

Game $\mathbf{G}_{0.p.1}$: In this game, we duplicate every $u_{m,t}$ into the 5-th column of the ciphertext. To this aim, one defines the matrices

$$D = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}_{6,7} \qquad D' = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix}_{6,7} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^* \qquad \mathbb{D} = D \cdot \mathbb{V}$$

which only modifies $\mathbf{d}_7$, which is secret, and $\mathbf{d}_6^*$, which is hidden, so the change is indistinguishable for the adversary. One can compute the keys and ciphertexts as follows, for all leaves λ, and for each of each query m of the adversary:

$$\begin{aligned} & & \mathbf{c}_{m,t} &= (\sigma_{m,t}(1,t), \omega_m, 0, 0, 0, u_{m,t}, 0, 0)_{\mathbb{V}} \\ & & &= (\sigma_{m,t}(1,t), \omega_m, 0, 0, u_{m,t}, u_{m,t}, 0, 0)_{\mathbb{D}} \\ t_\lambda < p & & \mathbf{k}_\lambda^* &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{V}^*} \\ & & &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*} \\ t_\lambda \geq p & & \mathbf{k}_\lambda^* &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, r_\lambda, 0, 0)_{\mathbb{V}^*} \\ & & &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, r_\lambda, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

Hence, the perfect indistinguishability between the two games: $\mathsf{Adv}_{0.p.1} = \mathsf{Adv}_{0.p.0}$.

Game $\mathbf{G}_{0.p.2}$: The previous game and this game are indistinguishable under the DSDH assumption in $\mathbb{G}_2$: one essentially uses theorem 4. Given a tuple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$ in $\mathbb{G}_2$, where $c = ab + \mu \bmod q$ with either $\mu = 0$ or $\mu = 1$, the 7-th component of the leaf λ of the challenge key, with $t_\lambda = p$. When we start from random dual orthogonal bases $(\mathbb{U}, \mathbb{U}^*)$ and $(\mathbb{V}, \mathbb{V}^*)$ of size 3 and 7 respectively, one considers the matrices:

$$D = \begin{pmatrix} 1 & 0 & 0 \\ a & 1 & 0 \\ -a & 0 & 1 \end{pmatrix}_{2,6,7} \qquad D' = \begin{pmatrix} 1 & -a & a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{2,6,7} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

We can calculate all vectors but $\mathbf{d}_6$ and $\mathbf{d}_7$, which are not in the public key. Through $\mathbb{V}$, we calculate the challenge key for the attribute of the p-th ciphertext

We choose additional random scalars $\beta_\lambda \xleftarrow{\$} \mathbb{Z}_q$, to virtually set $b_\lambda = r_\lambda \cdot b + \beta_\lambda$ and $c_\lambda = r_\lambda \cdot c + \beta_\lambda \cdot a$, then $c_\lambda - ab_\lambda = \mu \cdot r_\lambda$, which is either 0 or r_λ.

$$\begin{aligned} t_\lambda = p \quad \mathbf{k}_\lambda^* &= (0, 0, a_\lambda, 0, 0, 0, r_\lambda)_{\mathbb{D}^*} + (b_\lambda(t_\lambda, -1), 0, 0, 0, c_\lambda, -c_\lambda, 0, 0)_{\mathbb{V}^*} \\ &= (0, 0, a_\lambda, 0, 0, 0, r_\lambda, 0, 0)_{\mathbb{D}^*} + (b_\lambda(t_\lambda, -1), 0, 0, 0, c_\lambda - ab_\lambda, -c_\lambda + ab_\lambda, 0, 0)_{\mathbb{D}^*} \\ &= (b_\lambda(t_\lambda, -1), a_\lambda, 0, 0, \mu \cdot r_\lambda, r_\lambda - \mu \cdot r_\lambda, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

If $\mu = 0$, we are in the previous game. If $\mu = 1$, then we are in the current game. Then, every other key is computed directly in $\mathbb{D}^*$:

$$\begin{aligned} t_\lambda < p \quad \mathbf{k}_\lambda^* &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*} \\ t_\lambda > p \quad \mathbf{k}_\lambda^* &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, r_\lambda, 0, 0)_{\mathbb{D}^*} \end{aligned}$$

as well as the answers to OKeyGen-queries.

The ciphertexts are calculated through $\mathbb{V}$ but are unchanged by the change of basis because the 6-th and 7-th components are exactly the same for every ciphertext query m, and thus cancel themselves in the 2nd component. We thus have $\mathsf{Adv}_{0.p.1} - \mathsf{Adv}_{0.p.2} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

Game $\mathbf{G}_{0.p.3}$: We keep the $u_{m,p}$ value (at the 6-th hidden position) in the ciphertexts, and replace it in all other ciphertexts by 0. To show this is possible without impacting the other

vectors, we use the Index-Ind property from Theorem 7, but in another level of sequence of hybrid games, for $\gamma \in \{1, \ldots, P\}\backslash\{p\}$. We will enumerate γ in their order of appearance in the security game (wether in key queries, or in ciphertexts), therefore we can treat an unbounded number of γ.

Game $\mathbf{G}_{0.p.2.\gamma}$: We consider the following hybrid game, where the first satisfied condition on the indices is applied:

$$\begin{aligned}
\mathbf{c}_{m,p} &= (\sigma_{m,p}(1,p), \omega_m, 0, 0, u_{m,p}, u_{m,p}, 0, 0)_{\mathbb{D}} \\
\mathbf{c}_{m,t} &= (\sigma_{m,t}(1,t), \omega_m, 0, 0, 0, u_{m,t}, 0, 0)_{\mathbb{D}} && p \neq t < \gamma \\
\mathbf{c}_{m,t} &= (\sigma_{m,t}(1,t), \omega_m, 0, 0, u_{m,t}, u_{m,t}, 0, 0)_{\mathbb{D}} && p \neq t \geq \gamma
\end{aligned}$$

Keys are unchanged throughout the hybrid game

$$\begin{aligned}
\mathbf{k}^*_\lambda &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*} && t_\lambda < p \\
\mathbf{k}^*_\lambda &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, r_\lambda, 0, 0, 0)_{\mathbb{D}^*} && t_\lambda = p \\
\mathbf{k}^*_\lambda &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, 0, r_\lambda, 0, 0)_{\mathbb{D}^*} && t_\lambda > p
\end{aligned}$$

When $\gamma = 1$, this is the previous game: $\mathbf{G}_{0.p.2.1} = \mathbf{G}_{0.p.2}$, whereas with $\gamma = P + 1$, this is the current game: $\mathbf{G}_{0.p.2.P+1} = \mathbf{G}_{0.p.3}$.
We will gradually replace the $u_{m,t}$ values, at the 6-th hidden position, by 0 (when $t \neq p$): in this game, we deal with the case $t = \gamma$, for the m-th ciphertext query.
For this, we use the Adaptive Index-Ind property on $(\mathbb{D}^*, \mathbb{D})_{1,2,6,8,9}$, with:

$$\begin{aligned}
\mathbf{k}^*_\lambda &= (\pi_\lambda(t_\lambda, -1), a_\lambda, 0, 0, r_\lambda, 0, 0, 0)_{\mathbb{D}^*} && t_\lambda = p \\
\mathbf{c}_{m,t} &= (\sigma_{m,t}(1,t), \omega_m, 0, 0, u_{m,t}, u_{m,t}, 0, 0)_{\mathbb{D}} && t = \gamma
\end{aligned}$$

We remind that $u_{m,p} = 0$ because $r_\lambda \neq 0$. If $r_\lambda = 0$, then we would have skipped directly to the hybrid $p + 1$ game.

With all this sequence, we have $\mathsf{Adv}_{0.p.2} - \mathsf{Adv}_{0.p.3} \leq 2P \cdot (8 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t))$.

Game $\mathbf{G}_{0.p.4}$: In this final game for p, we can finally cancel out r_λ in each key with $t_\lambda = p$ because it corresponds to a coordinate where all other values (in keys and ciphertexts) are 0. We consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \alpha \mod q$, with either $\alpha = 0$ or $\alpha = r_\lambda$. One defines the matrices

$$D = \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix}_{1,6} \qquad D' = \begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix}_{1,6} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

Note that we can compute all the basis vectors excepted $\mathbf{d}_6$, but all the ciphertexts have a 0 components in 6-th position. So one can set all the values honestly in $\mathbb{D}$ and $\mathbb{D}^*$, except for

$$\begin{aligned}
\mathbf{k}_\lambda &= (0, 0, a_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{D}} + (b(p, -1), 0, 0, 0, c, 0, 0, 0)_{\mathbb{V}} \\
&= (0, 0, a_\lambda, 0, 0, 0, 0, 0, 0)_{\mathbb{D}} + (b(p, -1), 0, 0, 0, c - ab, 0, 0, 0)_{\mathbb{D}} \\
&= (b(1, p), a_\lambda, 0, 0, \alpha, 0, 0, 0)_{\mathbb{D}}
\end{aligned}$$

When $\alpha = 0$, this is exactly the current game, with $\pi_\lambda = b$, whereas $\alpha = r_\lambda$, this is the previous game. Then, $\mathsf{Adv}_{0.p.3} - \mathsf{Adv}_{0.p.4} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t)$.

In total, this sequence of games, for a given p, satisfies Then,

$$\begin{aligned}
\mathsf{Adv}_{\mathbf{G}_{0.p.4}} - \mathsf{Adv}_{\mathbf{G}_{0.p.0}} &\leq 4 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 2P \cdot (8 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t) + 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)) \\
&\leq (24P + 4) \cdot \mathsf{Adv}^{\mathsf{sxdh}}(t)
\end{aligned}$$

In the last game, the adversary has zero advantage. Indeed, whether $b = 0$ or $b = 1$, the distributions of dk_0 and dk_1 are perfectly identical, with all-passive leaves.

dAtt-IND-Security – Proof of Theorem 15

PROOF We start with the distinct variant, where all the invalid attributes in the challenge ciphertext do not correspond to any active leaf in the obtained keys. Our proof will proceed by games.

Game $\mathbf{G}_0$: This is the real security game, where the simulator honestly emulates the challenger, with $\mathsf{PK} = \{(\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}_1^*), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1^*, \mathbf{d}_2^*, \mathbf{d}_3^*)\}$ and $\mathsf{MK} = \{\mathbf{b}_3^*, \mathbf{d}_7^*\}$, from random dual orthogonal bases. The public parameters PK are provided to the adversary. Since $\mathbf{d}_7$ is public (empty SK), there is no need to provide access to an encryption oracle.

$\mathsf{OKeyGen}(\tilde{\mathcal{T}}_\ell)$ (or $\mathsf{ODelegate}$-queries): The adversary is allowed to issue KeyGen-queries on an access-tree $\tilde{\mathcal{T}}_\ell = (\mathcal{T}_\ell, \mathcal{L}_{\ell,a}, \mathcal{L}_{\ell,p})$ (for the ℓ-th query), for which the simulator chooses a random scalar $a_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$ and a random $a_{\ell,0}$-labeling $(a_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_\ell$, and builds the key:

$$\mathbf{k}_{\ell,0}^* = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}_{\ell,\lambda}^* = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}$$

for all the leaves λ, where $t_{\ell,\lambda} = A(\lambda)$, $\pi_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q$ and $r_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q^*$ if λ is an active leaf, or $r_{\ell,\lambda} \leftarrow 0$ otherwise. The decryption key is $\mathsf{dk}_\ell = (\mathbf{k}_{\ell,0}^*, (\mathbf{k}_{\ell,\lambda}^*)_\lambda)$;

$\mathsf{RoAVEncaps}(\Gamma_v, \Gamma_i)$**:** The challenge ciphertext is built on a set of attributes $\Gamma_v \cup \Gamma_i$, with random scalars $\omega, \xi \xleftarrow{\$} \mathbb{Z}_q$ to set $K = g_t^\xi$. Then, the simulator generates the ciphertext $C_0 = (\mathbf{c}_0, (\mathbf{c}_t)_t)$, for all the attributes $t \in \Gamma_v \cup \Gamma_i$, with $\sigma_t \xleftarrow{\$} \mathbb{Z}_q$, and where $u_t \xleftarrow{\$} \mathbb{Z}_q^*$ if $t \in \Gamma_i$, or $u_t = 0$ if $t \in \Gamma_v$:

$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1, t), \omega, 0, 0, 0, u_t, 0, 0)_{\mathbb{D}}$$

On the other hand, it computes $C_1 = (\mathbf{c}_0, (\mathbf{c}_t)_t)$ for all $t \in \Gamma_v \cup \Gamma_i$ as:

$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1, t), \omega, 0, 0, 0, 0, 0, 0)_{\mathbb{D}}$$

According to the real or all-valid game (bit $b \xleftarrow{\$} \{0, 1\}$), one outputs (K, C_b).

From the adversary's guess b' for b, if for some $\tilde{\mathcal{T}}_\ell = (\mathcal{T}_\ell, \mathcal{L}_{\ell,a}, \mathcal{L}_{\ell,p})$, there is some active leaf $\lambda \in \mathcal{L}_{\ell,a}$ such that $t_\lambda = A(\lambda) \in \Gamma_i$, then $\beta \xleftarrow{\$} \{0, 1\}$, otherwise $\beta = b'$. We denote $\mathsf{Adv}_0 = \Pr[\beta = 1 | b = 1] - \Pr[\beta = 1 | b = 0]$.

We stress that in this distinct attribute-indistinguishability security game, the invalid attributes in the challenge ciphertext ($t \in \Gamma_i$ with possibly $u_t \neq 0$) correspond to passive leaves only ($\lambda \in \mathcal{L}_{\ell,p}$ with $r_{\ell,\lambda} = 0$, for all queries). But we do not exclude accepting access-trees.

Game $\mathbf{G}_1$: The second and final game simply corresponds to the situation where $u_t = 0$ in C_0, clearly leading to $\mathsf{Adv}_1 = 0$.

Using the indexing technique, we can show this game is indistinguishable the previous game. But we need to describe a sub-sequence of games (see Figure 6.7) for proving the gap from the above $\mathbf{G}_0$ to $\mathbf{G}_1$, with the sequence $\mathbf{G}_{0.p.*}$, that will modify the p-th ciphertext in the challenge ciphertext, for $p \in \{1, \ldots, P+1\}$, where $\mathbf{G}_0 = \mathbf{G}_{0.1.0}$, and $\mathbf{G}_1 = \mathbf{G}_{0.P+1.0}$. In these games, we describe how we generate the keys and the real encapsulation C_0. C_1 will be easily simulated in an honest way.

Game $\mathbf{G}_{0.p.0}$: One thus chooses random scalars and defines the hybrid game for some p, where the first components of the ciphertext are all-valid, and the last ones are real:

$$\begin{array}{lll}
\mathbf{k}_{\ell,0}^* = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} & \mathbf{k}_{\ell,\lambda}^* = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} & \\
\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} & \mathbf{c}_t = (\sigma_t(1, t), \omega, 0, 0, 0, 0, 0, 0)_{\mathbb{D}} & t < p \\
\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} & \mathbf{c}_t = (\sigma_t(1, t), \omega, 0, 0, 0, u_t, 0, 0)_{\mathbb{D}} & t \geq p
\end{array}$$

$\mathbf{c}_0 = (\omega \quad 0 \quad \xi) \qquad \mathbf{k}^*_{\ell,0} = (a_{\ell,0} \quad 0 \quad 1)$

$\mathbf{G}_{0.p.0}$ Hybrid game for $\mathbf{G}_0$ and $\mathbf{G}_1$, with $1 \leq p \leq P+1$

	$\mathbf{k}^*_{\ell,\lambda} =$	$\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1)$	$a_{\ell,\lambda}$	0	0	0	$r_{\ell,\lambda}$	0 0
$t < p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	0	0 0
$t \geq p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	u_t	0 0

$\mathbf{G}_{0.p.1}$ Formal basis change, on $(\mathbb{D}, \mathbb{D}^*)_{6,7}$, to duplicate $r_{\ell,\lambda}$ in the 6-th column

	$\mathbf{k}^*_{\ell,\lambda} =$	$\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1)$	$a_{\ell,\lambda}$	0	0	$r_{\ell,\lambda}$	$r_{\ell,\lambda}$	0 0
$t < p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	0	0 0
$t \geq p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	u_t	0 0

$\mathbf{G}_{0.p.2}$ Swap-Ind Property, on $(\mathbb{D}, \mathbb{D}^*)_{1,6,7}$, to swap u_p alone in the 6-th column

	$\mathbf{k}^*_{\ell,\lambda} =$	$\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1)$	$a_{\ell,\lambda}$	0	0	$r_{\ell,\lambda}$	$r_{\ell,\lambda}$	0 0
$t < p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	0	0 0
	$\mathbf{c}_p =$	$\sigma_p(1,p)$	ω	0	0	u_p	0	0 0
$t > p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	u_t	0 0

$\mathbf{G}_{0.p.3}$ Index-Ind Property, on $(\mathbb{D}, \mathbb{D}^*)_{1,2,6}$, between $r_{\ell,\lambda}$ and 0, for $t_{\ell,\lambda} \neq p$

$t_{\ell,\lambda} = p$	$\mathbf{k}^*_{\ell,\lambda} =$	$\pi_{\ell,\lambda}(p, -1)$	$a_{\ell,\lambda}$	0	0	0	0	0 0
$t_{\ell,\lambda} \neq p$	$\mathbf{k}^*_{\ell,\lambda} =$	$\pi_{\ell,\lambda}(p, -1)$	$a_{\ell,\lambda}$	0	0	0	$r_{\ell,\lambda}$	0 0
$t < p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	0	0 0
	$\mathbf{c}_p =$	$\sigma_p(1,p)$	ω	0	0	u_p	0	0 0
$t > p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	u_t	0 0

$\mathbf{G}_{0.p.4}$ SubSpace-Ind Property, on $(\mathbb{D}, \mathbb{D}^*)_{1,6}$, between u_p and 0

	$\mathbf{k}^*_{\ell,\lambda} =$	$\pi_{\ell,\lambda}(p, -1)$	$a_{\ell,\lambda}$	0	0	0	$r_{\ell,\lambda}$	0 0
$t < p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	0	0 0
	$\mathbf{c}_p =$	$\sigma_p(1,p)$	ω	0	0	0	0	0 0
$t > p$	$\mathbf{c}_t =$	$\sigma_t(1,t)$	ω	0	0	0	u_t	0 0

Figure 6.7: Sub-sequence of games for Distinct Attribute-Indistinguishability

Of course, the values $r_{\ell,\lambda}$ and u_t are random in $\mathbb{Z}_q^*$ or 0 according to $\mathcal{L}_{\ell,a}/\mathcal{L}_{\ell,p}$ and Γ_i/Γ_v. In particular, if $u_p = 0$, we can directly go to $\mathbf{G}_{0.p.4}$, as there is no change from this game. The following sequence only makes sense when $u_p \neq 0$, but then necessarily $r_{\ell,\lambda} = 0$ for all the pairs (ℓ, λ) such that $t_{\ell,\lambda} = p$. We thus assume this restriction in this sequence: $u_p \neq 0$ and $r_{\ell,\lambda} = 0$ for all (ℓ, λ) such that $t_{\ell,\lambda} = p$.

Game $\mathbf{G}_{0.p.1}$: One defines the matrices

$$D = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}_{6,7} \qquad D' = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}_{6,7} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

which modifies the hidden and secret vectors $\mathbf{d}_6$ and $\mathbf{d}_7^*$, and so are not in the view of the adversary:

$$\begin{aligned}
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0, r_{\ell,\lambda}, 0, 0)_{\mathbb{V}^*} \\
&= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, r_{\ell,\lambda}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, 0, 0)_{\mathbb{V}} = (\sigma_t(1,t), \omega, 0, 0, 0, 0, 0, 0)_{\mathbb{D}} && \text{if } t < p \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, 0, u_t)_{\mathbb{V}} = (\sigma_t(1,t), \omega, 0, 0, 0, u_t, 0, 0)_{\mathbb{D}} && \text{if } t \geq p
\end{aligned}$$

We thus have $\mathsf{Adv}_{0.p.1} = \mathsf{Adv}_{0.p.0}$.

Game $\mathbf{G}_{0.p.2}$: We use the Swap-Ind-property on $(\mathbb{D}, \mathbb{D}^*)_{1,6,7}$: Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \theta \bmod q$ with either $\theta = 0$ or $\theta = u_p$. We define the matrices

$$D = \begin{pmatrix} 1 & a & -a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{1,6,7} \qquad D' = \begin{pmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ a & 0 & 1 \end{pmatrix}_{1,6,7} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

Note that we can compute all the basis vectors excepted $\mathbf{d}_6^*, \mathbf{d}_7^*$, but we define the keys on the original basis $\mathbb{V}^*$:

$$\begin{aligned}
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, r_{\ell,\lambda}, r_{\ell,\lambda}, 0, 0)_{\mathbb{V}^*} \\
&= (\pi_{\ell,\lambda} \cdot t_{\ell,\lambda} + a r_{\ell,\lambda} - a r_{\ell,\lambda}, -\pi_{\ell,\lambda}, a_{\ell,\lambda}, 0, 0, r_{\ell,\lambda}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \\
&= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, r_{\ell,\lambda}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, 0, 0, 0, 0)_{\mathbb{D}} && \text{if } t < p \\
\mathbf{c}_p &= (\sigma(1,p), \omega, 0, 0, 0, u_p, 0, 0)_{\mathbb{D}} + (b(1,p), 0, 0, 0, c, -c, 0, 0)_{\mathbb{V}} \\
&= (\sigma(1,p), \omega, 0, 0, 0, u_p, 0, 0)_{\mathbb{D}} + (b(1,p), 0, 0, 0, c - ab, -c + ab, 0, 0)_{\mathbb{D}} \\
&= ((\sigma + b)(1,p), \omega, 0, 0, \theta, u_p - \theta, 0, 0)_{\mathbb{D}} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, 0, 0, 0, u_t, 0, 0)_{\mathbb{D}} && \text{if } t > p
\end{aligned}$$

With $\theta = 0$, this is as in the previous game, where $\sigma_p = \sigma + b$. When $\theta = u_p$, this is the current game: $\mathsf{Adv}_{0.p.1} - \mathsf{Adv}_{0.p.2} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

Game $\mathbf{G}_{0.p.3}$: We make all the $r_{\ell,\lambda}$ values (at the 6-th hidden position) in the keys to be 0, excepted for $t_{\ell,\lambda} = p$. The case $t_{\ell,\lambda} = p$ is already $r_{\ell,\lambda} = 0$, by assumption in this sequence, as $u_p \neq 0$. For that, we iteratively replace all the values by zero, using the Adaptive Index-Ind-property from theorem 7, in another level of sequence of hybrid games, for $\gamma \in \{1, \ldots, P\} \backslash \{p\}$. We will enumerate γ in their order of appearance in the security game (wether in key queries, or in ciphertexts), therefore we can treat an unbounded number of γ.

Game $\mathbf{G}_{0.p.2.\gamma}$: We consider

$$\begin{aligned}
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0,0,0,0,0,0)_{\mathbb{D}^*} && \text{if } t_{\ell,\lambda} = p\\
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0,0,0,r_{\ell,\lambda},0,0)_{\mathbb{D}^*} && \text{if } p \neq t_{\ell,\lambda} < \gamma\\
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0,0,r_{\ell,\lambda},r_{\ell,\lambda},0,0)_{\mathbb{D}^*} && \text{if } p \neq t_{\ell,\lambda} \geq \gamma\\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, 0,0,0,0,0,0)_{\mathbb{D}} && \text{if } t < p\\
\mathbf{c}_p &= (\sigma_p(1,p), \omega, 0,0,u_p,0,0,0)_{\mathbb{D}}\\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, 0,0,0,u_t,0,0)_{\mathbb{D}} && \text{if } t > p
\end{aligned}$$

When $\gamma = 1$, this is the previous game: $\mathbf{G}_{0.p.2.1} = \mathbf{G}_{0.p.2}$, whereas with $\gamma = P+1$, this is the current game: $\mathbf{G}_{0.p.2.P+1} = \mathbf{G}_{0.p.3}$.
For this, we use the Adaptive Index-Ind property on $(\mathbb{D}^*, \mathbb{D})_{1,2,6,8,9}$, with:

$$\begin{aligned}
\mathbf{c}_p &= (\sigma_p(1,p), \omega, 0,0,u_p,0,0,0)_{\mathbb{D}}\\
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0,0,r_{\ell,\lambda},r_{\ell,\lambda},0,0)_{\mathbb{D}^*} && t_{\ell,\lambda} = \gamma
\end{aligned}$$

As a consequence, $\mathsf{Adv}_{0.p.2} - \mathsf{Adv}_{0.p.3} \leq 2P \cdot (8 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t))$.

Game $\mathbf{G}_{0.p.4}$: One can easily conclude by removing u_p in the ciphertext $\mathbf{c}_p$, as it corresponds to a coordinate where all the other values (in the keys and the ciphertext) are 0. To this aim, we can consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \alpha \bmod q$ with either $\alpha = 0$ or $\alpha = u_p$. One defines the matrices

$$D = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}_{1,6} \qquad D' = \begin{pmatrix} 1 & 0 \\ -a & 1 \end{pmatrix}_{1,6} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

Note that we can compute all the basis vectors excepted $\mathbf{d}^*_6$, which has only 0 components in the keys. So one can set all the values honestly in $\mathbb{D}$ and $\mathbb{D}^*$, excepted

$$\begin{aligned}
\mathbf{c}_p &= (b(1,p), \omega, 0,0,c,0,0,0)_{\mathbb{V}} = (b(1,p), \omega, 0,0,c-ab,0,0,0)_{\mathbb{D}}\\
&= (b(1,p), \omega, 0,0,\alpha,0,0,0)_{\mathbb{D}}
\end{aligned}$$

When $\alpha = 0$, this is exactly the current game, with $\sigma_p = b$, whereas for $\alpha = u_p$, this is the previous game. Then, $\mathsf{Adv}_{0.p.3} - \mathsf{Adv}_{0.p.4} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

In total, this sequence of games, for a given p, satisfies Then,

$$\begin{aligned}
\mathsf{Adv}_{\mathbf{G}_{0.p.4}} - \mathsf{Adv}_{\mathbf{G}_{0.p.0}} &\leq 4 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 2P \cdot (8 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t) + 4 \times \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_2}(t))\\
&\leq (4 + 24P) \cdot \mathsf{Adv}^{\mathsf{sxdh}}(t)
\end{aligned}$$

Att-IND-Security – Proof of Theorem 16

Proof We now prove the attribute-indistinguishability, where there are no restrictions between active leaves in the keys and invalid attributes in the challenge ciphertext, but just that the access-trees of the obtained keys reject the attribute-set of the challenge ciphertext, even in the all-valid case. Our proof will proceed by games. Not that we also assume active keys correspond to independent leaves with respect to the set of attributes $\Gamma = \Gamma_v \cup \Gamma_i$ in the challenge ciphertext.

Game $\mathbf{G}_0$: This is the real security game, where the simulator honestly emulates the challenger, with $\mathsf{PK} = \{(\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}^*_1), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_7, \mathbf{d}^*_1, \mathbf{d}^*_2, \mathbf{d}^*_3)\}$ and $\mathsf{MK} = \{\mathbf{b}^*_3, \mathbf{d}^*_7\}$, from random dual orthogonal bases. The public parameters PK are provided to the adversary. Since $\mathbf{d}_7$ is public (empty SK), there is no need to provide access to an encryption oracle.

$\mathsf{OKeyGen}(\tilde{\mathcal{T}}_\ell)$ (or $\mathsf{ODelegate}$-queries): The adversary is allowed to issue KeyGen-queries on an access-tree $\tilde{\mathcal{T}}_\ell = (\mathcal{T}_\ell, \mathcal{L}_{\ell,a}, \mathcal{L}_{\ell,p})$ (for the ℓ-th query), for which the simulator chooses a random scalar $a_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$ and a random $a_{\ell,0}$-labeling $(a_{\ell,\lambda})_\lambda$ of the access-tree $\mathcal{T}_\ell$, and builds the key:

$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, 0, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, 0, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}$$

for all the leaves λ, where $t_{\ell,\lambda} = A(\lambda)$, $\pi_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q$ and $r_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q^*$ if λ is an active leaf, or $r_{\ell,\lambda} \leftarrow 0$ otherwise. The decryption key is $\mathsf{dk}_\ell = (\mathbf{k}^*_{\ell,0}, (\mathbf{k}^*_{\ell,\lambda})_\lambda)$;

$\mathsf{RoAVEncaps}(\Gamma_v, \Gamma_i)$**:** The challenge ciphertext is built on a set of attributes $\Gamma_v \uplus \Gamma_i$, with random scalars $\omega, \xi \xleftarrow{\$} \mathbb{Z}_q$ to set $K = g_t^\xi$. Then, the simulator generates the ciphertext $C_1 = (\mathbf{c}_0, (\mathbf{c}_t)_t)$, for all the attributes $t \in \Gamma_v \uplus \Gamma_i$, with $\sigma_t \xleftarrow{\$} \mathbb{Z}_q$:

$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1, t), \omega, 0, 0, 0, 0, 0, 0)_{\mathbb{D}}$$

On the other hand, it computes $C_0 = (\mathbf{c}_0, (\mathbf{c}_t + (0, 0, 0, 0, 0, 0, u_t, 0, 0)_{\mathbb{D}})_t)$, where $u_t \xleftarrow{\$} \mathbb{Z}_q^*$ if $t \in \Gamma_i$, or $u_t = 0$ if $t \in \Gamma_v$. According to the real or all-valid game (bit $b \xleftarrow{\$} \{0, 1\}$), one outputs (K, C_b).

From the adversary's guess b' for b, if for some $\tilde{\mathcal{T}}_\ell = (\mathcal{T}_\ell, \mathcal{L}_{\ell,a}, \mathcal{L}_{\ell,p})$, for which tree a key has been obtained, $\tilde{\mathcal{T}}_\ell(\Gamma_v \uplus \Gamma_i, \emptyset) = 1$ then $\beta \xleftarrow{\$} \{0, 1\}$, otherwise $\beta = b'$. We denote $\mathsf{Adv}_0 = \Pr[\beta = 1 | b = 1] - \Pr[\beta = 1 | b = 0]$.

We now proceed with exactly the same sequence as in the IND-security proof of the $\mathsf{KP\text{-}ABE}$ in 8, except the $\mathsf{RoREncaps}$-challenge is instead a $\mathsf{RoAVEncaps}$-challenge, where we require $\tilde{\mathcal{T}}_\ell(\Gamma_v \uplus \Gamma_i, 0) = 0$ for all the obtained keys. For the same reason, the $\mathsf{OEncaps}$-queries on pairs $(\Gamma_{m,v}, \Gamma_{m,i})$, with $\Gamma_{m,i} \neq \emptyset$ can be simulated. Indeed, as above, everything on the 7-th component can be done independently, knowing both $\mathbf{d}_7$ and $\mathbf{d}^*_7$, as these vectors will be known to the simulator, almost all the time, excepted in some specific gaps. In theses cases, we will have to make sure how to simulate the $\mathsf{OEncaps}$ ciphertexts.

As in that proof, the idea of the sequence is to introduce an additional labeling $(s_{\ell,0}, (s_{\ell,\lambda})_\lambda)$ in the hidden components of each key, with a random $s_{\ell,0}$, as the trees are rejecting. We are thus able to go as in $\mathbf{G}_3$, from Figure 5.1, where each label is masked by a random z_t for each attribute t. The following sequence is described on Figure 6.8.

Game $\mathbf{G}_1$: This is as $\mathbf{G}_1$, with a random τ in the challenge ciphertext.

Game $\mathbf{G}_2$: This is as $\mathbf{G}_2$, with random z_t in the challenge ciphertext.

Game $\mathbf{G}_3$: This is as $\mathbf{G}_3$, with an additional independent $s_{\ell,0}$-labeling $(s_{\ell,\lambda})$ for each access-tree $\mathcal{T}_\ell$ and a random $r_{\ell,0}$ to define

$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, r_{\ell,0}, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, s_{\ell,\lambda}/z_{t_{k,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}$$

We stress that all these steps are not impacted by the values u_t in the 7-th component of the challenge ciphertext:

$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1, t), \omega, \tau, 0, \tau z_t, (1 - b) \cdot u_t, 0, 0)_{\mathbb{D}}$$

where b is the random bit of the challenger: when $b = 0$, the ciphertext is in the real case, whereas for $b = 1$, one gets an all-valid ciphertext.

$\mathbf{G}_0$ Real Att-IND-Security game

$\mathbf{c}_0 = (\ \omega \quad 0 \quad \xi\ )$ $\mathbf{c}_t = (\dots \mid 0 \quad 0 \quad 0 \quad (1-b)\cdot u_t \mid 0\ 0\ )$

$\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \quad 0 \quad 1\ )$ $\mathbf{k}^*_{\ell,\lambda} = (\dots \mid 0 \quad 0 \quad 0 \quad r_{\ell,\lambda} \mid 0\ 0\ )$

$\mathbf{G}_1$ SubSpace-Ind Property, on $(\mathbb{B},\mathbb{B}^*)_{1,2}$ and $(\mathbb{D},\mathbb{D}^*)_{3,4}$, between 0 and $\tau \xleftarrow{\$} \mathbb{Z}_q$

$\mathbf{c}_0 = (\ \omega \quad \tau \quad \xi\ )$ $\mathbf{c}_t = (\dots \mid \tau \quad 0 \quad 0 \quad (1-b)\cdot u_t \mid 0\ 0\ )$

$\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \quad 0 \quad 1\ )$ $\mathbf{k}^*_{\ell,\lambda} = (\dots \mid 0 \quad 0 \quad 0 \quad r_{\ell,\lambda} \mid 0\ 0\ )$

$\mathbf{G}_2$ SubSpace-Ind Property, on $(\mathbb{D},\mathbb{D}^*)_{(1,2),6}$, between 0 and τz_t

$\mathbf{c}_0 = (\ \omega \quad \tau \quad \xi\ )$ $\mathbf{c}_t = (\dots \mid \tau \quad 0 \quad \tau z_t \quad (1-b)\cdot u_t \mid 0\ 0\ )$

$\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \quad 0 \quad 1\ )$ $\mathbf{k}^*_{\ell,\lambda} = (\dots \mid 0 \quad 0 \quad 0 \quad r_{\ell,\lambda} \mid 0\ 0\ )$

$\mathbf{G}_3$ Additional random-labeling as in the IND-security proof. See Figure 5.2

$\mathbf{c}_0 = (\ \omega \quad \tau \quad \xi\ )$ $\mathbf{c}_t = (\dots \mid \tau \quad 0 \quad \tau z_t \quad (1-b)\cdot u_t \mid 0\ 0\ )$

$\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \quad r_{\ell,0} \quad 1\ )$ $\mathbf{k}^*_{\ell,\lambda} = (\dots \mid 0 \quad 0 \quad s_{\ell,\lambda}/z_{t_{\ell,\lambda}} \quad r_{\ell,\lambda} \mid 0\ 0\ )$

$\mathbf{G}_4$ Index-Ind property to suppress u_t, when $b = 0$. See Figure 6.9

$\mathbf{c}_0 = (\ \omega \quad \tau \quad \xi\ )$ $\mathbf{c}_t = (\dots \mid \tau \quad 0 \quad \tau z_t \quad 0 \mid 0\ 0\ )$

$\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \quad r_{\ell,0} \quad 1\ )$ $\mathbf{k}^*_{\ell,\lambda} = (\dots \mid 0 \quad 0 \quad s'_{\ell,\lambda}/z_{t_{\ell,\lambda}} \quad r_{\ell,\lambda} \mid 0\ 0\ )$

$\mathbf{G}_5$ Limitation of independent active leaves

$\mathbf{c}_0 = (\ \omega \quad \tau \quad \xi\ )$ $\mathbf{c}_t = (\dots \mid \tau \quad 0 \quad \tau z_t \quad 0 \mid 0\ 0\ )$

$\mathbf{k}^*_{\ell,0} = (\ a_{\ell,0} \quad r_{\ell,0} \quad 1\ )$ $\mathbf{k}^*_{\ell,\lambda} = (\dots \mid 0 \quad 0 \quad s_{\ell,\lambda}/z_{t_{\ell,\lambda}} \quad r_{\ell,\lambda} \mid 0\ 0\ )$

Figure 6.8: Global sequence of games for the Att-IND-security proof of our SA-KP-ABE

Game $\mathbf{G}_4$: We remove all u_t from the RoAVEncaps challenge query, in the case $b = 1$:

$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, 0, 0, 0)_{\mathbb{D}}$$
$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, r_{\ell,0}, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, s'_{\ell,\lambda}/z_{t_{k,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}$$

where $s'_{\ell,\lambda}$ is either the label $s_{\ell,\lambda}$ or an independent random value when $u_{t_{k,\lambda}} \cdot r_{k,\lambda} \neq 0$, in the case $b = 0$. And nothing is changed when $b = 1$. To this aim, we use a different sequence $\mathbf{G}_{3.p.*}$ presented in the Figure 6.9, when $b = 1$ only, for $p \in \{1, \dots, P\}$, that will modify the p-th ciphertext in the challenge ciphertext, where $\mathbf{G}_3 = \mathbf{G}_{3.1.0}$, and $\mathbf{G}_4 = \mathbf{G}_{3.P+1.0}$.

Game $\mathbf{G}_{3.p.0}$: One thus chooses random scalars and defines the hybrid game for some p, where the first components of the ciphertext are all-valid, and the last ones are real:

$$\mathbf{k}^*_{\ell,0} = (a_{\ell,0}, r_{\ell,0}, 1)_{\mathbb{B}^*} \qquad \mathbf{k}^*_{\ell,\lambda} = (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, 0, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}$$
$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, 0, 0, 0)_{\mathbb{D}} \quad \text{if } t < p$$
$$\mathbf{c}_0 = (\omega, 0, \xi)_{\mathbb{B}} \qquad \mathbf{c}_t = (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} \quad \text{if } t \geq p$$

Of course, the values $r_{\ell,\lambda}$ and u_t are random in $\mathbb{Z}_q^*$ or 0 according to $\mathcal{L}_{\ell,a}/\mathcal{L}_{\ell,p}$ and Γ_i/Γ_v. In particular, if $u_p = 0$, we can directly go to $\mathbf{G}_{3.p.5}$, as there is no change from this game. But there is no need to know it in advance, and so we can follow this sequence in any case and set u_p in the ciphertext at the challenge-time.

$\mathbf{G}_{3.p.0}$	Hybrid game for $\mathbf{G}_3$ and $\mathbf{G}_4$, with $1 \leq p \leq P+1$									
$t < p$	$\mathbf{c}_t =$	(	$\sigma_t(1,t)$	ω	|	τ	0	τz_t	0	| 0 0)
$t \geq p$	$\mathbf{c}_t =$	(	$\sigma_t(1,t)$	ω	|	τ	0	τz_t	u_t	| 0 0)
	$\mathbf{k}^*_{\ell,0} =$	(	$a_{\ell,0}$	$r_{\ell,0}$	1)					
	$\mathbf{k}^*_{\ell,\lambda} =$	(	$\pi_{\ell,\lambda}(t_{\ell,\lambda},-1)$	$a_{\ell,\lambda}$	|	0	0	$s_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	| 0 0)
$\mathbf{G}_{3.p.1}$	Formal basis change, on $(\mathbb{D},\mathbb{D}^*)_{5,7}$, to duplicate $r_{\ell,\lambda}$ in the 5-th column									
	$\mathbf{c}_p =$	(	$\sigma_p(1,p)$	ω	|	τ	0	τz_p	u_p	| 0 0)
	$\mathbf{k}^*_{\ell,\lambda} =$	(	$\pi_{\ell,\lambda}(t_{\ell,\lambda},-1)$	$a_{\ell,\lambda}$	|	0	$r_{\ell,\lambda}$	$s_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	| 0 0)
$\mathbf{G}_{3.p.2}$	Swap-Ind Property, on $(\mathbb{D},\mathbb{D}^*)_{1,5,7}$, to swap u_p alone in the 5-th column									
	$\mathbf{c}_p =$	(	$\sigma_p(1,p)$	ω	|	τ	u_p	τz_p	0	| 0 0)
$t \neq p$	$\mathbf{c}_t =$	(	$\sigma_t(1,t)$	ω	|	τ	0	τz_t	u_t	| 0 0)
$\mathbf{G}_{3.p.3}$	Index-Ind Property, on $(\mathbb{D}^*,\mathbb{D})_{1,2,5}$, between $r_{\ell,\lambda}$ and 0, for $t_{\ell,\lambda} \neq p$									
	$\mathbf{c}_p =$	(	$\sigma_p(1,p)$	ω	|	τ	u_p	τz_p	0	| 0 0)
$t_{\ell,\lambda} \neq p$	$\mathbf{k}^*_{\ell,\lambda} =$	(	$\pi_{\ell,\lambda}(t_{\ell,\lambda},-1)$	$a_{\ell,\lambda}$	|	0	0	$s_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	| 0 0)
$t_{\ell,\lambda} = p$	$\mathbf{k}^*_{\ell,\lambda} =$	(	$\pi_{\ell,\lambda}(t_{\ell,\lambda},-1)$	$a_{\ell,\lambda}$	|	0	$r_{\ell,\lambda}$	$s_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	| 0 0)
$\mathbf{G}_{3.p.4}$	SubSpace-Ind Property, on $(\mathbb{D},\mathbb{D}^*)_{6,5}$, between u_p and 0									
	$\mathbf{c}_p =$	(	$\sigma_p(1,p)$	ω	|	τ	0	τz_p	0	| 0 0)
$t_{\ell,\lambda} \neq p$	$\mathbf{k}^*_{\ell,\lambda} =$	(	$\pi_{\ell,\lambda}(t_{\ell,\lambda},-1)$	$a_{\ell,\lambda}$	|	0	0	$s_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	| 0 0)
$t_{\ell,\lambda} = p$	$\mathbf{k}^*_{\ell,\lambda} =$	(	$\pi_{\ell,\lambda}(t_{\ell,\lambda},-1)$	$a_{\ell,\lambda}$	|	0	$r_{\ell,\lambda}$	$s'_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	| 0 0)
$\mathbf{G}_{3.p.5}$	SubSpace-Ind Property, on $(\mathbb{D}^*,\mathbb{D})_{6,5}$, between $r_{\ell,\lambda}$ and 0, for $t_{\ell,\lambda} = p$									
$t \leq p$	$\mathbf{c}_t =$	(	$\sigma_t(1,t)$	ω	|	τ	0	τz_t	0	| 0 0)
$t > p$	$\mathbf{c}_t =$	(	$\sigma_t(1,t)$	ω	|	τ	0	τz_t	u_t	| 0 0)
$t_{\ell,\lambda} \neq p$	$\mathbf{k}^*_{\ell,\lambda} =$	(	$\pi_{\ell,\lambda}(t_{\ell,\lambda},-1)$	$a_{\ell,\lambda}$	|	0	0	$s_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	| 0 0)
$t_{\ell,\lambda} = p$	$\mathbf{k}^*_{\ell,\lambda} =$	(	$\pi_{\ell,\lambda}(t_{\ell,\lambda},-1)$	$a_{\ell,\lambda}$	|	0	0	$s'_{\ell,\lambda}/z_{t_{\ell,\lambda}}$	$r_{\ell,\lambda}$	| 0 0)

Figure 6.9: Hybrid game on p for the Att-IND-security proof of our SA-KP-ABE, when $b = 0$

Game $\mathbf{G}_{3.p.1}$: One defines the matrices

$$D = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}_{5,7} \qquad D' = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}_{5,7} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

which modifies the hidden and secret vectors $\mathbf{d}_6$ and $\mathbf{d}_7^*$, and so are not in the view of the adversary:

$$\begin{aligned}
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0, 0, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{V}^*} \\
&= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, 0, 0, 0)_{\mathbb{V}} = (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, 0, 0, 0)_{\mathbb{D}} && \text{if } t < p \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, u_t, 0, 0)_{\mathbb{V}} = (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && \text{if } t \geq p
\end{aligned}$$

We thus have $\mathsf{Adv}_{3.p.1} = \mathsf{Adv}_{3.p.0}$.

Game $\mathbf{G}_{3.p.2}$: We use the Swap-Ind-property on $(\mathbb{D}, \mathbb{D}^*)_{1,5,7}$: Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \theta \bmod q$ with either $\theta = 0$ or $\theta = u_p$. We define the matrices

$$D = \begin{pmatrix} 1 & a & -a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{1,5,7} \qquad D' = \begin{pmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ a & 0 & 1 \end{pmatrix}_{1,5,7} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

Note that we can compute all the basis vectors excepted $\mathbf{d}_5^*, \mathbf{d}_7^*$, but we define the keys on the original basis $\mathbb{V}^*$:

$$\begin{aligned}
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{V}^*} \\
&= (\pi_{\ell,\lambda} \cdot t_{\ell,\lambda} + a r_{\ell,\lambda} - a r_{\ell,\lambda}, -\pi_{\ell,\lambda}, a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \\
&= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, 0, 0, 0)_{\mathbb{D}} && \text{if } t < p \\
\mathbf{c}_p &= (\sigma(1,p), \omega, \tau, 0, \tau z_p, u_p)_{\mathbb{D}} + (b(1,p), 0, 0, c, 0, -c, 0, 0)_{\mathbb{V}} \\
&= (\sigma(1,p), \omega, \tau, 0, \tau z_p, u_p)_{\mathbb{D}} + (b(1,p), 0, 0, c - ab, 0, -c + ab, 0, 0)_{\mathbb{D}} \\
&= ((\sigma + b)(1,p), \omega, \tau, \theta, \tau z_p, u_p - \theta, 0, 0)_{\mathbb{D}} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && \text{if } t > p
\end{aligned}$$

With $\theta = 0$, this is as in the previous game, where $\sigma_p = \sigma + b$. When $\theta = u_p$, this is the current game: $\mathsf{Adv}_{3.p.1} - \mathsf{Adv}_{3.p.2} \leq 2 \cdot \mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}_1}(t)$.

Game $\mathbf{G}_{3.p.3}$: We make all the $r_{\ell,\lambda}$ values (at the 5-th hidden position) in the keys to be 0, excepted when $t_{\ell,\lambda} = p$. For that, we iteratively replace all the values by zero, using Adaptive Index-Ind-property from theorem 7, in another level of sequence of hybrid games, for $\gamma \in \{1, \ldots, P\} \backslash \{p\}$. We will enumerate γ in their order of appearance in the security game (wether in key queries, or in ciphertexts), therefore we can treat an unbounded number of γ.

Game $\mathbf{G}_{3.p.2.\gamma}$: We consider

$$\begin{aligned}
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && \text{if } t_{\ell,\lambda} = p \\
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0, 0, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && \text{if } p \neq t_{\ell,\lambda} < \gamma \\
\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda},-1), a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} && \text{if } p \neq t_{\ell,\lambda} \geq \gamma \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, 0, 0, 0)_{\mathbb{D}} && \text{if } t < p \\
\mathbf{c}_p &= (\sigma_p(1,p), \omega, \tau, u_p, \tau z_p, 0, 0, 0)_{\mathbb{D}} \\
\mathbf{c}_t &= (\sigma_t(1,t), \omega, \tau, 0, \tau z_t, u_t, 0, 0)_{\mathbb{D}} && \text{if } t > p
\end{aligned}$$

When $\gamma = 1$, this is the previous game: $\mathbf{G}_{3.p.2.1} = \mathbf{G}_{3.p.2}$, whereas with $\gamma = P + 1$, this is the current game: $\mathbf{G}_{3.p.2.P+1} = \mathbf{G}_{3.p.3}$.
For this, we use the Adaptive Index-Ind property on $(\mathbb{D}^*, \mathbb{D})_{1,2,5,8,9}$, with:

$$\begin{aligned}\mathbf{c}_p &= (\sigma_p(1,p), \omega, \tau, u_p, \tau z_p, 0, 0, 0)_{\mathbb{D}}\\ \mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(t_{\ell,\lambda}, -1), a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s_{\ell,\lambda}/z_{t_{\ell,\lambda}}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*} \qquad t_{\ell,\lambda} = \gamma\end{aligned}$$

Game $\mathbf{G}_{3.p.4}$: We use the SubSpace-Ind-property on $(\mathbb{D}, \mathbb{D}^*)_{6,5}$: Indeed, we can consider a triple $(a \cdot G_1, b \cdot G_1, c \cdot G_1)$, where $c = ab + \theta \bmod q$ with either $\theta = 0$ or $\theta = u_p$. We define the matrices

$$D = \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix}_{5,6} \qquad D' = \begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix}_{5,6} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

Note that we can compute all the basis vectors excepted $\mathbf{d}^*_5$ that is not public, and not used excepted for the keys with $t_{\ell,\lambda} = p$, which will be defined in the original basis $\mathbb{V}^*$:

$$\begin{aligned}\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(p,-1), a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s_{\ell,\lambda}/z_p, r_{\ell,\lambda}, 0, 0)_{\mathbb{V}^*}\\ &= (\pi_{\ell,\lambda}(p,-1), a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s_{\ell,\lambda}/z_p + a r_{\ell,\lambda}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}\\ &= (\pi_{\ell,\lambda}(p,-1), a_{\ell,\lambda}, 0, r_{\ell,\lambda}, s'_{\ell,\lambda}/z_p, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}\\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, b, 0, b z_t, 0, 0, 0)_{\mathbb{D}} \qquad \text{if } t < p\\ \mathbf{c}_p &= (\sigma_p(1,p), \omega, b, c, b z_p, 0, 0, 0)_{\mathbb{V}} = (\sigma_p(1,p), \omega, b, c - ab, b z_p, 0, 0, 0)_{\mathbb{D}}\\ &= (\sigma_p(1,p), \omega, b, \theta, b z_p, 0, 0, 0)_{\mathbb{D}}\\ \mathbf{c}_t &= (\sigma_t(1,t), \omega, b, 0, b z_t, u_t, 0, 0)_{\mathbb{D}} \qquad \text{if } t > p\end{aligned}$$

When $\theta = 0$, this is this game, whereas when $\theta = u_p$, this is the previous game, with $\tau = b$ and $s'_{\ell,\lambda} = s_{\ell,\lambda} + a z_p r_{\ell,\lambda}$ a new random and independent value for each active leaf associated to the attribute p.

Game $\mathbf{G}_{3.p.5}$: We use the SubSpace-Ind-property on $(\mathbb{D}^*, \mathbb{D})_{6,5}$: Indeed, we can consider a triple $(a \cdot G_2, b \cdot G_2, c \cdot G_2)$, where $c = ab + \zeta \bmod q$ with either $\zeta = 0$ or $\zeta = 1$. We define the matrices

$$D' = \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix}_{5,6} \qquad D = \begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix}_{5,6} \qquad \mathbb{D} = D \cdot \mathbb{V} \qquad \mathbb{D}^* = D' \cdot \mathbb{V}^*$$

Note that we can compute all the basis vectors excepted $\mathbf{d}_5$ that is not public, and not used in the ciphertext. All the vectors can be computed in the new bases, excepted the keys for $t_{\ell,\lambda} = p$, for which one chooses additional random scalars $\beta_{\ell,\lambda} \xleftarrow{\$} \mathbb{Z}_q$, to virtually set $b_{\ell,\lambda} = r_{\ell,\lambda} \cdot b + \beta_{\ell,\lambda}$ and $c_{\ell,\lambda} = r_{\ell,\lambda} \cdot c + \beta_{\ell,\lambda} \cdot a$, $c_{\ell,\lambda} - a b_{\ell,\lambda} = r_{\ell,\lambda} \cdot \zeta$.

$$\begin{aligned}\mathbf{k}^*_{\ell,\lambda} &= (\pi_{\ell,\lambda}(p,-1), a_{\ell,\lambda}, 0, c_{\ell,\lambda}, b_{\ell,\lambda}, r_{\ell,\lambda}, 0, 0)_{\mathbb{V}^*}\\ &= (\pi_{\ell,\lambda}(p,-1), a_{\ell,\lambda}, 0, c_{\ell,\lambda} - a b_{\ell,\lambda}, b_{\ell,\lambda}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}\\ &= (\pi_{\ell,\lambda}(p,-1), a_{\ell,\lambda}, 0, \zeta \cdot r_{\ell,\lambda}, b_{\ell,\lambda}, r_{\ell,\lambda}, 0, 0)_{\mathbb{D}^*}\end{aligned}$$

When $\zeta = 0$, this is this game, whereas when $\zeta = 1$, this is the previous game, with $s'_{\ell,\lambda} = z_p \cdot b_{\ell,\lambda}$, a truly random and independent value for each active leaf associated to the attribute p.

Game $\mathbf{G}_5$: Under the assumption of independent active leaves with respect to the set of attributes $\Gamma = \Gamma_v \cup \Gamma_i$ in the challenge ciphertext, the random values $s'_{\ell,\lambda}$ are indistinguishable from real labels $s'_{\ell,\lambda}$. Indeed, labels that correspond to leaves that are associated to attributes not in Γ are unknown, as the masks z_t are not revealed. This shows that the advantage of the adversary in this last game is 0.

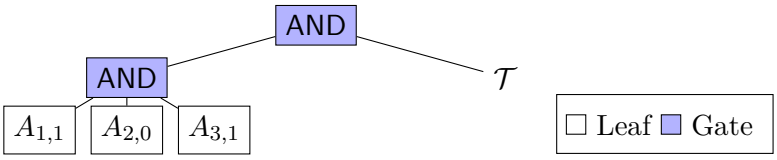

Figure 6.10: Tracing sub-tree for the codeword $w = (1, 0, 1)$

6.5 Application to Traitor-Tracing

Before going into the formalism, we give an intuition of our method. In our Traitor-Tracing approach, any user would be given a key associated to a word in a traceable code at key generation time. To embed a word inside a key, the key generation authority only needs to create a new policy for a user with policy $\mathcal{T}$: the new policy will be a root AND gate, that connects the original access-tree $\mathcal{T}$ as one child, and a word-based access-tree composed of active leaves as another child, as illustrated on Figure 6.10.

From there, the tracing authority, using the secret key SK, could trace any Pirate Decoder by invalidating attributes associated to the positions in words, one position at a time. Since an adversary cannot know whether attributes are valid or invalid, he will answer each queries of the tracer, as long as not impacted by the invalid attributes (thanks to the Distinct Attribute-Indistinguishability), effectively revealing the bits of his word on each position. When the tracer finds his complete word, he eventually traces back the traitors, from the traceable-code properties. Furthermore, thanks to the Attribute-Indistinguishability (not Distinct), a traitor that has been identified by the tracing authority can be removed from the target set at tracing time, and can thus no longer participate in the coalition, as it will be excluded from the policy, whatever the valid/invalid attributes. We stress that the secret key SK is required for invalidating some attributes, and so for the tracing. We thus have secret-key black-box traceability.

6.5.1 Delegatable and Traceable KP-ABE

Our initial motivation was to adapt KP-ABE with delegation to support tracing, which should not be detectable by the pirate decoder. We now explain how our SA-KP-ABE primitive allows that. We recall the definitions of tracing, and then we illustrate with a possible family of policies with switchable leaves and attributes. We first add a Tracing algorithm to the initial definition of delegatable KP-ABE from 5.2:

$\mathsf{Setup}(1^\kappa, n, t)$**.** From the security parameter κ, the total number n of users in the system, and the maximal size t on the collusion, the algorithm defines all the global parameters PK, the master secret key MK, and the tracing key TK;

$\mathsf{KeyGen}(\mathsf{MK}, \mathcal{T}, \mathsf{id})$**.** From a master key MK and an access tree $\mathcal{T}$, the algorithm outputs a key $\mathsf{dk}_{\mathsf{id},\mathcal{T}}$, specific to the user id;

$\mathsf{Delegate}(\mathsf{dk}_{\mathsf{id},\mathcal{T}}, \mathcal{T}', \mathsf{id}')$**.** Given a key $\mathsf{dk}_{\mathsf{id},\mathcal{T}}$ and a more restrictive access-tree $\mathcal{T}' \leq \mathcal{T}$, the algorithm outputs a decryption key $\mathsf{dk}_{\mathsf{id}',\mathcal{T}'}$;

$\mathsf{Encaps}(\mathsf{PK}, \Gamma)$**.** For a set Γ of attributes, the algorithm generates the ciphertext C and an encapsulated key K;

$\mathsf{Decaps}(\mathsf{dk}_{\mathsf{id},\mathcal{T}}, C)$**.** Given the key $\mathsf{dk}_{\mathsf{id},\mathcal{T}}$ and the ciphertext C, the algorithm outputs the encapsulated key K;

$\mathsf{Trace}^D(\mathsf{SK}, \Gamma)$**.** Given the secret key SK, and a black-box access to a Pirate Decoder D, the tracing algorithm outputs an index set I which identifies a set of malicious users, among the users id and id' compatible with Γ.

In the above definition, id' might be for a specific device of user id. Then the authority generates keys for users, and users delegate for devices, with any more restrictive policy $\mathcal{T}' \leq \mathcal{T}$: one can consider that $\mathsf{id}' = \mathsf{id}\|d$, for device d. One can then trace users and devices.

We expect two properties from the Trace algorithm on a perfect Pirate Decoder for a set Γ (that always decrypts the encapsulated key), when the number of traitors compatible with Γ is at most t: it always outputs a non-empty set of traitors, but does never wrongly accuse anybody.

Definition 13 (Traceability) Initialize: The challenger runs the Setup algorithm and gives the public parameters PK to the adversary;

$\mathsf{OKeyGen}(\mathsf{id}, \mathcal{T})$**:** The adversary is allowed to issue KeyGen-queries for any access-tree $\mathcal{T}$ of its choice, the corresponding secret key $\mathsf{dk}_{\mathsf{id},\mathcal{T}}$ is generated;

$\mathsf{ODelegate}(\mathsf{id}, \mathcal{T}, \mathsf{id}', \mathcal{T}')$**:** The adversary is allowed to issue several $\mathsf{Delegate}$-queries for any more restrictive access-tree $\mathcal{T}' \leq \mathcal{T}$ of its choice, for an already generated decryption key for $\mathcal{T}$, and the corresponding secret key $\mathsf{dk}_{\mathsf{id}',\mathcal{T}'}$ is generated;

$\mathsf{OGetKey}(\mathsf{id}, \mathcal{T})$**:** The adversary can then ask and see the secret key $\mathsf{dk}_{\mathsf{id},\mathcal{T}}$, if it has been generated, else it gets $\perp$;

Finalize: The adversary generates a set of attributes Γ and a perfect Decoder D on Γ, the challenger runs $\mathsf{Trace}^D(\mathsf{SK}, \Gamma)$ to get back I. Let us denote U_c (corrupted users) the set of id' for which $\mathcal{T}'$ has been asked such that $\mathcal{T}'(\Gamma) = 1$. If the size of U_c is at most t, but $I \not\subset U_c$ or I is empty, one outputs 1, otherwise one outputs 0.

The success $\mathsf{Adv}^{\mathsf{trace}}(\mathcal{A})$ of an adversary $\mathcal{A}$ in this game is the probability to have 1 as output.

We stress that the above definition requires a perfect Pirate Decoder. This could be relaxed, but this is enough for our illustration.

6.5.2 Fingerprinting Code

Our technique will exploit traceable codes as in [CFN94] that allow to trace back codewords from words that have been derived from legitimate codewords. It uses the definition of feasible set, the list the words that can be derived from a set of words:

Definition 14 (Feasible Set) Let $W = \{w^{(1)}, \ldots, w^{(t)}\}$ be a set of t words in $\{0,1\}^\ell$. We say a word $w \in \{0,1\}^\ell$ is **feasible** for W if for all $i = 1, \ldots, \ell$, there is a $j \in \{1, \ldots, t\}$ such that $w_i = w_i^{(j)}$. The set of words feasible for W is the **feasible set** of W, denoted $F(W) = \{w \in \{0,1\}^\ell, \forall i, \exists w' \in W, w_i = w'_i\}$.

A fingerprinting code is a particular traceable code. It defines a set of codewords that allows correct and efficient tracing to recover the traitor codewords from a word derived from them (in the feasible set). For the sake of clarity, we focus on binary codes:

Definition 15 (Fingerprinting Code) A **fingerprinting code** is a pair of algorithms (G, T) defined as follows:

Code generator G is a probabilistic algorithm that takes a tuple (n, t) as input, where n is the number of codewords to output, and t is the maximal collusion size. The algorithm outputs a **code** Π of n codewords of bit-length ℓ.

Tracing algorithm T is a deterministic algorithm that takes as input a word $w^* \in \{0,1\}^\ell$ to trace. The algorithm T outputs a subset $S \subseteq \Pi$ of possible traitors.

Such a fingerprinting code is said t-**secure** if for all $n > t$ and all subsets $C \subseteq \{1, \ldots, n\}$ of size at most t, when we set $\Pi = \{w^{(1)}, \ldots, w^{(n)}\} \leftarrow G(n, t)$ and $W_C = \{w^{(i)}\}_{i \in C}$, for any word $w^* \in F(W_C)$, then $\emptyset \neq T(w^*) \subseteq C$.

Again, we could relax the definition with error probabilities in identifying a traitor and in framing an honest user. Tardos codes [Tar03] are examples of short codes with probabilistic tracing capabilities and low error rates.

6.5.3 Delegatable and Traceable KP-ABE from SA-KP-ABE

We now explain how our SA-KP-ABE primitive is enough for tracing. For the sake of simplicity, in the following, we will keep $\mathsf{id}' = \mathsf{id}$, without specifying the device, still with any $\mathcal{T}' \leq \mathcal{T}$, but then devices of the same user cannot be traced. Only users can be traced, but various devices might have different policies:

$\mathsf{Setup}^{\mathsf{Tr}}(1^\kappa, n, t)$**.** The algorithm calls $\mathsf{Setup}(1^\kappa)$ and gets back $\mathsf{PK}, \mathsf{MK}, \mathsf{SK}$. It also calls code generator algorithm $G(n, t)$ to get the code Π. It sets the parameters as $\mathsf{PK}^{\mathsf{Tr}} = \mathsf{PK}$, $\mathsf{MK}^{\mathsf{Tr}} = (\mathsf{MK}, \Pi)$ and $\mathsf{TK}^{\mathsf{Tr}} = (\mathsf{SK}, T)$.

$\mathsf{KeyGen}^{\mathsf{Tr}}(\mathsf{MK}^{\mathsf{Tr}}, \mathsf{id}, \mathcal{T})$**.** For an access-tree $\mathcal{T}$, the algorithm defines $\mathcal{T}^{\mathsf{Tr}}$, where $\mathcal{T}^{\mathsf{Tr}} = \mathcal{T} \wedge \mathcal{T}_{\mathsf{Tr}}$ are linked by an AND-gate at their root. The access-tree $\mathcal{T}_{\mathsf{Tr}}$ is constructed as follows (see Figure 6.10) :

- Choose a word $w_{\mathsf{id}} = w_{\mathsf{id},1} \ldots w_{\mathsf{id},\ell}$ from Π, for any new id;
- Set $\mathcal{T}_{\mathsf{Tr}}$ as the AND of active leaves λ_i associated to the attributes $A_{i,w_{\mathsf{id},i}}$, for $i = 1, \ldots, \ell$.

The algorithm then calls $\mathsf{KeyGen}(\mathsf{MK}, \tilde{\mathcal{T}}^{\mathsf{Tr}})$, where all leaves are passive in $\mathcal{T}$ and all leaves are active in $\mathcal{T}_{\mathsf{Tr}}$, and gets back $\mathsf{dk}_{\tilde{\mathcal{T}}^{\mathsf{Tr}}}$, and finally sets $\mathsf{dk}^{\mathsf{Tr}}_{\mathsf{id},\mathcal{T}} \leftarrow \mathsf{dk}_{\tilde{\mathcal{T}}^{\mathsf{Tr}}}$.

$\mathsf{Delegate}^{\mathsf{Tr}}(\mathsf{dk}^{\mathsf{Tr}}_{\mathsf{id},\mathcal{T}}, \mathcal{T}')$**.** Given a private key for an access-tree $\mathcal{T}$ and a more restrictive sub-tree $\mathcal{T}' \leq \mathcal{T}$, but for the same identity (as we focus on $\mathsf{id}' = \mathsf{id}$), the algorithm calls $\mathsf{Delegate}(\mathsf{dk}_{\tilde{\mathcal{T}}}, \tilde{\mathcal{T}}')$, where $\tilde{\mathcal{T}}$ and $\tilde{\mathcal{T}}'$ are $\mathcal{T}$ and $\mathcal{T}'$ combined with $\mathcal{T}_{\mathsf{Tr}}$ as above, to get a new delegated key $\mathsf{dk}_{\tilde{\mathcal{T}}'}$, and sets $\mathsf{dk}^{\mathsf{Tr}}_{\mathsf{id},\mathcal{T}'} = \mathsf{dk}_{\tilde{\mathcal{T}}'}$.

$\mathsf{Encaps}^{\mathsf{Tr}}(\mathsf{PK}^{\mathsf{Tr}}, \Gamma)$**.** For a set Γ of attributes, the algorithm defines $\Gamma^{\mathsf{Tr}} = \{A_{1,0}, A_{1,1}, \ldots, A_{\ell,0}, A_{\ell,1}\}$. It then calls $\mathsf{Encaps}(\mathsf{PK}, \Gamma \cup \Gamma^{\mathsf{Tr}})$ and gets the output K and C. It then sets $K^{\mathsf{Tr}} = K$ and $C^{\mathsf{Tr}} = C$.

$\mathsf{Decaps}^{\mathsf{Tr}}(\mathsf{dk}^{\mathsf{Tr}}_{\mathsf{id},\mathcal{T}}, C)$**.** The algorithm calls $\mathsf{Decaps}^{\mathsf{Tr}}(\mathsf{dk}_{\tilde{\mathcal{T}}}, C)$, for $\mathsf{dk}_{\tilde{\mathcal{T}}} = \mathsf{dk}^{\mathsf{Tr}}_{\mathsf{id},\mathcal{T}}$, to get K, and outputs $K^{\mathsf{Tr}} = K$.

$\mathsf{Trace}^{\mathsf{Tr}}(\mathsf{TK}^{\mathsf{Tr}}, \Gamma)$**.** On input the tracing key $\mathsf{TK}^{\mathsf{Tr}} = (\mathsf{SK}, T)$, and access to a perfect Pirate Decoder D, the algorithm repeats the following experiment, for $j = 1, \ldots, \ell$, to build the word w^*:

1. Set $\Gamma_v^{(0)} = \Gamma \cup \{A_{k,\ell}, k \neq j, \ell \in \{0,1\}\} \cup \{A_{j,0}\}$ and $\Gamma_i^{(0)} = \{A_{j,1}\}$;
2. Set $\Gamma_v^{(1)} = \Gamma \cup \{A_{k,\ell}, k \neq j, \ell \in \{0,1\}\} \cup \{A_{j,1}\}$ and $\Gamma_i^{(1)} = \{A_{j,0}\}$;
3. Compute the two challenges $(K_0, C_0) \leftarrow \mathsf{Encaps}^*(\mathsf{SK}, (\Gamma_v^{(0)}, \Gamma_i^{(0)}))$ and $(K_1, C_1) \leftarrow \mathsf{Encaps}^*(\mathsf{SK}, (\Gamma_v^{(1)}, \Gamma_i^{(1)}))$;
4. Flip a random coin $b \xleftarrow{\$} \{0,1\}$, and ask for the decryption K' of C_b to D;
5. If $K' = K_b$ then set $w_j^* \leftarrow b$, else set $w_j^* \leftarrow 1 - b$.

Eventually, the algorithm runs the tracing algorithm $T(w^*)$ to get S, the set of traitors, that it outputs.

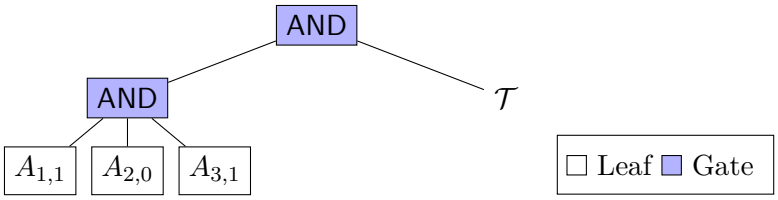

Figure 6.11: Tracing sub-tree for the codeword $w = (1, 0, 1)$

Security Analysis. Again, we stress that we assume perfect Pirate Decoder D, but relaxed version would be possible. Hence, here, we know that D will successfully decrypt any normal ciphertext when Γ is acceptable for all the traitors. Then, during the tracing procedure, for any index j, there are three possibilities:

- If $w_j = b$ for all keys in U_c, then the ciphertext C_b is indistinguishable from the one where $A_{j,1-b}$ is in Γ_v because of the Distinct Attribute-Indistinguishability (Att-IND) property of the scheme, hence D will always output K_b. We correctly set $w_j^* = b$.
- If $w_j = 1 - b$ for all keys in U_c, K_b will be unpredictable because of the Delegation-Indistinguishability for Encaps*, we correctly set $w_j^* = 1 - b$.
- If w_j has mixed values in 0 and 1 among users in U_c, D can detect that C_b involves active keys. But we could anyway set $w_j^* \leftarrow 0$ or $w_j^* \leftarrow 1$.

This way, the built word w^* satisfies that, for each position j, $w_j^* = w_j$, for some w in U_c: $w^* \in F(U_c)$. If the fingerprinting code is t-secure, since the size of U_c is at most t, $\emptyset \neq T(w^*) \subseteq U_c$. As a consequence, under the Distinct Attribute-Indistinguishability and the Delegation-Indistinguishability for Encaps*, the delegatable KP-ABE is traceable.

Discussions. Our tracing system is presented with basic fingerprinting notions, for the sake of clarity, but more advanced features are possible. In particular, our tracing algorithm works as well with non-perfect Pirate Decoder, at the cost of more calls to D to increase the quality of the estimation. It is also compatible with [BN08], to drastically reduce the ciphertext size. Eventually, one could also let the user to delegate traceable keys to each devices. However, as we do not allow public traceability, only the tracing authority can run the tracing procedure, to trace users or devices.

Chapter

7 ABS with Delegation and Tracing

Chapter content

Our last contribution is to improve the ABS scheme presented in Section 5.3.2 with two different kinds of delegation, either delegating a subset of attributes from a key, or restricting the signing rights to a specific policy, while still keeping anonymity. This goal was the motivation behind separating message and policy in two different hash values in the signature. Furthermore, we prove that our new delegateable signature scheme is compatible with the tracing, which is a common feature for signatures. We deploy this traceability through the use of One-Time Linearly Homomorphic Signature.

7.1 Attribute and Policy Delegations

7.1.1 Definition of Delegateable ABS

In addition to the initial definition of an ABS, with the algorithms Setup, KeyGen, Sig, and Verif, we also consider delegation algorithms, with an additional signing algorithm:

Delegate-Attributes$(\mathsf{SK}_{\mathsf{id},\Gamma}, \bar{\mathsf{id}}, \Gamma')$. From $\mathsf{SK}_{\mathsf{id},\Gamma}$, and for a subset $\Gamma' \subset \Gamma$, one can derive a signing key $\mathsf{SK}_{\bar{\mathsf{id}},\Gamma'}$ for a user $\bar{\mathsf{id}}$.

Delegate-Policy$(\mathsf{SK}_{\mathsf{id},\Gamma}, \bar{\mathsf{id}}, \mathcal{P})$. For a private key $\mathsf{SK}_{\mathsf{id},\Gamma}$ on a set of attributes Γ and a policy $\mathcal{P}$ that accepts Γ, the algorithm outputs a policy key $\mathsf{SK}_{\bar{\mathsf{id}},\mathcal{P}}$;

DelegateSig$(\mathsf{SK}_{\mathsf{id},\mathcal{P}}, m)$. For a delegated key $\mathsf{SK}_{\mathsf{id},\mathcal{P}}$ on a policy $\mathcal{P}$ and a message m, the algorithm outputs a signature σ;

The delegated keys from the first algorithm can be used in a similar way as a fresh key. For this reason, when we refer to a key in the following, it can be from either KeyGen or Delegate-Attributes without distinction, except if specified otherwise. It thus allows multi-hop delegation of attributes. On the other hand, policy delegation provides different keys, hence another signing algorithm, and will usually be called *policy keys*. However, for the correctness, we add that the Verif algorithm should output 1 with overwhelming probability on $(\sigma, m, \mathcal{P})$ even if σ has been generated on m with a policy key $\mathsf{SK}_{\bar{\mathsf{id}},\mathcal{P}}$. In both cases, we note $\bar{\mathsf{id}}$ the new full identity associated to the keys, which could be formed for example by concatenation: $\bar{\mathsf{id}} = \mathsf{id}||\mathsf{id}'$, for some id', and even possibly a longer chain, as only delegated attributes under the exact same chain might be combined as a new key.

7.1.2 Security Model for Delegateable ABS

Unforgeability and Privacy need some changes to be consistent with the new nature of delegated keys: One should not be able to produce an accepted signature under a policy $\mathcal{P}$ if one does not own the appropriate attributes or the delegated key for unforgeability, and signatures generated by fresh keys or delegated keys should be indistinguishable for privacy. Thus, we update the security definitions given in Section 3.4.2 to take delegation into account. In particular, for the delegation of attributes, we now consider that the adversary will have access to delegated keys via the ODelegateAttributes oracle, which he can use on keys he previously queried for via either OKeyGen or ODelegateAttributes. Regarding the policy delegation, we add two additional oracles: ODelegatePolicy to obtain a policy key from a previous OKeyGen or ODelegateAttributes query, and ODelegateSig to obtain a signature from a policy key ODelegatePolicy that has already been queried. None of the keys are actually revealed to the adversary, unless he queries specifically for them via OGet, to model the real information learnt by the adversary. Indeed, some keys can be generated, but only as source of delegations, and only delegated keys will be known to the adversary, in case the adversary is just a delegatee.

Definition 16 (Existential Unforgeability) EUF for ABS with delegation is defined by the following game between the adversary and a challenger:

Initialize: The challenger runs the Setup algorithm of ABS and gives the public parameters PK to the adversary;

Oracles: The following oracles can be called in any order and any number of times:

- $\mathsf{OKeyGen}(\mathsf{id}, \Gamma)$**:** to model KeyGen-queries for any identity id and any set of attributes Γ of its choice, and gets back the index k of the key;
- $\mathsf{ODelegateAttributes}(k, \bar{\mathsf{id}}, \Gamma')$**:** to model Delegate-Attributes-queries for identity $\bar{\mathsf{id}}$ and any subset of attributes $\Gamma' \subset \Gamma$, for the k-indexed generated key from Γ. It generates the decryption key but only outputs the index k' of the new key;
- $\mathsf{ODelegatePolicy}(k, \bar{\mathsf{id}}, \mathcal{P})$**:** to model Delegate-Policy-queries for identity $\bar{\mathsf{id}}$ and any policy $\mathcal{P}$, from the k-indexed generated key for Γ so that $\mathcal{P}(\Gamma) = 1$. It generates the new policy key but only outputs the index k' of the new policy key;
- $\mathsf{OGet}(k)$**:** the adversary gets back the k-indexed key generated by one of the above oracles;
- $\mathsf{OSig}(\mathsf{id}, m, \mathcal{P})$**:** to model Sig-queries under any policy $\mathcal{P}$ of its choice for a message m, for the identity id, and gets back the signature;
- $\mathsf{ODelegateSig}(\bar{\mathsf{id}}, m, \mathcal{P}))$**:** to model DelegateSig-queries for any message m, for identity $\bar{\mathsf{id}}$ and policy $\mathcal{P}$. It generates and outputs the signature.

Finalize(b'): The adversary outputs a forgery $(m', \mathcal{P}', \sigma')$. If for some attribute set Γ corresponding to a key asked to the OGet oracle, $\mathcal{P}'(\Gamma) = 1$, or if the adversary queried OSig or

$\mathsf{ODelegateSig}$ on $(m', \mathcal{P}')$, or if the adversary queries $\mathsf{ODelegatePolicy}$ on $\mathcal{P}'$, one outputs 0. Otherwise one outputs $\mathsf{Verif}(\mathsf{PK}, m', \mathcal{P}', \sigma')$.

The advantage $\mathsf{Adv}^{\mathsf{del\text{-}euf}}(\mathcal{A})$ of an adversary $\mathcal{A}$ in this game is defined as the probability to output 1.

As usual, the Finalize-step excludes trivial attacks, where the adversary owns a key able to generate an acceptable signature or just forwards a query asked to the signing oracle.

We also update the definition of anonymity, to take into account the possibilities of delegation: whether coming from some kind of delegation or not, every signature must depend only on the policy that is signed, and not on the specific user's keys.

Definition 17 (Anonymity) An ABS with delegation scheme is said anonymous if :

- for any $(\mathsf{PK}, \mathsf{MK}) \xleftarrow{\$} \mathsf{Setup}$, any message m, any identities $\mathsf{id}_0, \mathsf{id}_1$, any attribute sets Γ_0, Γ_1
- for any signing keys $\mathsf{SK}_0 \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}_0, \Gamma_0)$, $\mathsf{SK}_1 \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}_1, \Gamma_1)$
- for any delegated keys, with $\Gamma'_0 \subset \Gamma_0$ and $\Gamma'_1 \subset \Gamma_1$, $\mathsf{SK}'_0 \xleftarrow{\$} \mathsf{Delegate\text{-}Attributes}(\mathsf{SK}_0, \mathsf{id}'_0, \Gamma'_0)$, $\mathsf{SK}'_1 \xleftarrow{\$} \mathsf{Delegate\text{-}Attributes}(\mathsf{SK}_1, \mathsf{id}'_1, \Gamma'_1)$
- for any policy keys $\tilde{\mathsf{SK}}'_0 \xleftarrow{\$} \mathsf{Delegate\text{-}Policy}(\mathsf{SK}_0, \mathcal{P})$, $\tilde{\mathsf{SK}}'_1 \xleftarrow{\$} \mathsf{Delegate\text{-}Attributes}(\mathsf{SK}_1, \mathcal{P})$, for any policy $\mathcal{P}$ that accepts both Γ'_0 and Γ'_1

the six distributions of the signatures from $\mathsf{Sig}(\mathsf{SK}_0, m, \mathcal{P}), \mathsf{Sig}(\mathsf{SK}'_0, m, \mathcal{P}), \mathsf{DelegateSig}(\tilde{\mathsf{SK}}'_0, m)$, $\mathsf{Sig}(\mathsf{SK}_1, m, \mathcal{P}), \mathsf{Sig}(\mathsf{SK}'_1, m, \mathcal{P}), \mathsf{DelegateSig}(\tilde{\mathsf{SK}}'_1, m)$ are indistinguishable.

Indistinguishability can be perfect, statistical or computational, which leads to perfect, statistical or computational anonymity.

7.2 Description of our Delegateable ABS

$\mathsf{Setup}(1^\kappa)$**.** The algorithm chooses three random dual orthogonal bases, in a pairing-friendly setting $\mathcal{PG} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$:

$$\mathbb{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_4) \qquad \mathbb{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_{10}) \qquad \mathbb{H} = (\mathbf{h}_1, \ldots, \mathbf{h}_8)$$
$$\mathbb{B}^* = (\mathbf{b}^*_1, \ldots, \mathbf{b}^*_4) \qquad \mathbb{D}^* = (\mathbf{d}^*_1, \ldots, \mathbf{d}^*_{10}) \qquad \mathbb{H}^* = (\mathbf{h}^*_1, \ldots, \mathbf{h}^*_8).$$

It picks two hash functions $\mathcal{H}$ and $\mathcal{H}'$ onto $\mathbb{Z}_q$. It sets the public parameters $\mathsf{PK} = \{\mathcal{PG}, \mathcal{H}, \mathcal{H}', (\mathbf{b}_1, \mathbf{b}_3), (\mathbf{b}^*_2), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_5), (\mathbf{d}^*_1, \mathbf{d}^*_2, \mathbf{d}^*_3, \mathbf{d}^*_4), (\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3, \mathbf{h}_5), (\mathbf{h}^*_4)\}$, and master secret key $\mathsf{MK} = \{(\mathbf{b}^*_1), (\mathbf{h}^*_1, \mathbf{h}^*_2, \mathbf{h}^*_3)\} \cup \mathsf{PK}$.

$\mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}, \Gamma)$**.** A random scalar $\delta \xleftarrow{\$} \mathbb{Z}^*_q$ is associated to id, to define

$$\mathbf{k}^*_0 = \delta \cdot \mathbf{b}^*_1 + \phi_0 \cdot \mathbf{b}^*_2 \qquad \mathbf{k}^*_t = \delta \cdot \mathbf{d}^*_1 + \pi_t \cdot \mathbf{d}^*_2 + t\pi_t \cdot \mathbf{d}^*_3 \qquad + \phi_t \cdot \mathbf{d}^*_4$$
$$\mathbf{r}^*_1 = \delta \cdot \mathbf{h}^*_1 + \psi_1 \cdot \mathbf{h}^*_4 \qquad \mathbf{r}^*_2 = \delta \cdot \mathbf{h}^*_2 + \psi_2 \cdot \mathbf{h}^*_4 \qquad \mathbf{r}^*_3 = \delta \cdot \mathbf{h}^*_3 + \psi_3 \cdot \mathbf{h}^*_4$$

for all attributes $t \in \Gamma$, with $\phi_0, \psi_1, \psi_2, \psi_3, (\phi_t)_t, (\pi_t)_t \xleftarrow{\$} \mathbb{Z}^*_q$ for each t. The signing key $\mathsf{SK}_{\mathsf{id},\Gamma}$ is then $(\mathbf{k}^*_0, (\mathbf{k}^*_t)_{t\in\Gamma}, \mathbf{r}^*_1, \mathbf{r}^*_2, \mathbf{r}^*_3)$. It can be completed later for new attributes for id, with extra $\mathbf{k}^*_t$, using the same δ.

$\mathsf{Delegate\text{-}Attributes}(\mathsf{SK}_{\mathsf{id},\Gamma}, \bar{\mathsf{id}}, \Gamma')$**.** Pick random scalars $\alpha, \phi'_0, (\phi'_t)_t, \psi'_1, \psi'_2, \psi'_3 \xleftarrow{\$} \mathbb{Z}_q$ for $t \in \Gamma' \subset \Gamma$, and compute

$$\bar{\mathbf{k}}^*_0 = \alpha \cdot \mathbf{k}^*_0 + \phi'_0 \cdot \mathbf{b}^*_2 \qquad \bar{\mathbf{k}}^*_t = \alpha \cdot \mathbf{k}^*_t + \phi'_t \cdot \mathbf{d}^*_4$$
$$\bar{\mathbf{r}}^*_1 = \alpha \cdot \mathbf{r}^*_1 + \psi'_1 \cdot \mathbf{h}^*_4 \qquad \bar{\mathbf{r}}^*_2 = \alpha \cdot \mathbf{r}^*_2 + \psi'_2 \cdot \mathbf{h}^*_4 \qquad \bar{\mathbf{r}}^*_3 = \alpha \cdot \mathbf{r}^*_3 + \psi'_3 \cdot \mathbf{h}^*_4$$

The delegated signing key $\mathsf{SK}_{\bar{\mathsf{id}},\Gamma'}$ is then set as $(\bar{\mathbf{k}}_0^*, (\bar{\mathbf{k}}_t^*)_{t\in\Gamma'}, \bar{\mathbf{r}}_1^*, \bar{\mathbf{r}}_2^*, \bar{\mathbf{r}}_3^*)$. More delegations can be provided with additional $\bar{\mathbf{k}}_t^*$ for $\bar{\mathsf{id}}$ from id using the same α, specific to the attribute-delegation from id to $\bar{\mathsf{id}}$.

$\mathsf{Delegate\text{-}Policy}(\mathsf{SK}_{\mathsf{id},\Gamma}, \bar{\mathsf{id}}, \mathcal{T})$**.** Let $\mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma)$ be an Evaluation Pruned Tree, $\nu, \xi, \zeta, \psi, (\omega_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q^*$, and $(\alpha_\lambda)_\lambda$ the 1-labeling of $\mathcal{T}^*$ associated to $\mathcal{T}'$ (see Proposition 2), where $\alpha_\lambda = 1$ if $\lambda \in \mathcal{L}_{\mathcal{T}'}$, else $\alpha_\lambda = 0$. This is possible as $\mathcal{T}(\Gamma) = 1$. Then, compute $(\beta_\lambda)_\lambda$ to be a random 0-labeling of $\mathcal{T}^*$, and $(q_\lambda)_\lambda$ random scalars. Eventually, set, for $H = \mathcal{H}(\mathcal{T})$:

$$U^* = \xi\mathbf{k}_0^* + \zeta\mathbf{b}_2^* \qquad S_\lambda^* = \alpha_\lambda\xi \cdot \mathbf{k}_{t_\lambda}^* + \beta_\lambda\mathbf{d}_1^* + \omega_\lambda(\mathbf{d}_2^* + t_\lambda \cdot \mathbf{d}_3^*) + q_\lambda \cdot \mathbf{d}_4^*$$
$$\mathbf{r}_3'^* = \xi\mathbf{r}_3^* + \psi\mathbf{h}_4^* \qquad V^* = \xi(\mathbf{r}_1^* + H \cdot \mathbf{r}_2^*) + \nu \cdot \mathbf{h}_4^*$$

for all the leaves λ, where t_λ is the associated attribute of λ. The delegated key is thus $\mathsf{SK}_{\bar{\mathsf{id}},\mathcal{T}} = (U^*, V^*, \mathbf{r}_3'^*, (S_\lambda^*)_\lambda)$.

$\mathsf{Sig}(\mathsf{SK}_{\mathsf{id},\Gamma}, \mathcal{T}, m)$**.** Let $\mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma)$ be an Evaluation Pruned Tree, and pick $\nu, \xi, \zeta, (\omega_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q^*$, and (α_λ) the 1-labeling of $\mathcal{T}^*$ associated to $\mathcal{T}'$ (see Proposition 2), where $\alpha_\lambda = 1$ if $\lambda \in \mathcal{L}_{\mathcal{T}'}$, else $\alpha_\lambda = 0$. This is possible as $\mathcal{T}(\Gamma) = 1$. Then, compute (β_λ) to be a random 0-labeling of $\mathcal{T}^*$, and $(q_\lambda)_\lambda$ random scalars. Set, for $H = \mathcal{H}(\mathcal{T})$ and $H' = \mathcal{H}'(m)$:

$$U'^* = \xi\mathbf{k}_0^* + \zeta\mathbf{b}_2^* \qquad S_\lambda'^* = \alpha_\lambda\xi \cdot \mathbf{k}_{t_\lambda}^* + \beta_\lambda\mathbf{d}_1^* + \omega_\lambda(\mathbf{d}_2^* + t_\lambda \cdot \mathbf{d}_3^*) + q_\lambda \cdot \mathbf{d}_4^*$$
$$V'^* = \xi(\mathbf{r}_1^* + H \cdot \mathbf{r}_2^* + H' \cdot \mathbf{r}_3^*) + \nu \cdot \mathbf{h}_4^*$$

for all the leaves λ, where t_λ is the associated attribute of λ.

The signature is thus $\sigma = (U'^*, V'^*, (S_\lambda'^*)_\lambda)$.

$\mathsf{DelegateSig}(\mathsf{SK}_{\mathsf{id},\mathcal{T}}, m)$**.** Let $\mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma)$ be an Evaluation Pruned Tree, $\nu, \xi, \zeta, (\omega_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q^*$, $(\beta_\lambda)_\lambda$ a random 0-labeling of $\mathcal{T}^*$, and $(q_\lambda)_\lambda$ random scalars. Set, for $H' = \mathcal{H}'(m)$:

$$U'^* = \xi U^* + \zeta\mathbf{b}_2^* \qquad S_\lambda'^* = \xi \cdot S_\lambda^* + \beta_\lambda\mathbf{d}_1^* + \omega_\lambda(\mathbf{d}_2^* + t_\lambda \cdot \mathbf{d}_3^*) + q_\lambda \cdot \mathbf{d}_4^*$$
$$V'^* = \xi(V^* + H' \cdot \mathbf{r}_3'^*) + \nu \cdot \mathbf{h}_4^*$$

for all the leaves λ, where t_λ is the associated attribute of λ.

The signature is thus $\sigma = (U'^*, V'^*, (S_\lambda'^*)_\lambda)$.

$\mathsf{Verif}(\mathsf{PK}, m, \mathcal{T}, \sigma)$**.** Let $\kappa, \kappa_0, (\kappa_\lambda)_\lambda, s, s_0, \theta, \theta', (\theta_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q$. Let $(s_\lambda)_\lambda$ be a random s_0-labeling of $\mathcal{T}$, then set, for $\bar{H} = \mathcal{H}(\mathcal{T}), \bar{H}' = \mathcal{H}'(m)$:

$$u = -(s_0 + s) \cdot \mathbf{b}_1 + \kappa_0 \cdot \mathbf{b}_3 \qquad c_\lambda = s_\lambda \cdot \mathbf{d}_1 + \theta_\lambda t_\lambda \cdot \mathbf{d}_2 - \theta_\lambda \cdot \mathbf{d}_3 + \kappa_\lambda \cdot \mathbf{d}_5$$
$$v = (s + \theta\bar{H} + \theta'\bar{H}') \cdot \mathbf{h}_1 - \theta \cdot \mathbf{h}_2 - \theta' \cdot \mathbf{h}_3 + \kappa \cdot \mathbf{h}_5$$

If $e(\mathbf{b}_1, U^*) \neq 1_{\mathbb{G}_t} \wedge e(u, U^*) \cdot e(v, V^*) \cdot \prod e(c_\lambda, S_\lambda^*) = 1_{\mathbb{G}_t}$, accept, else reject.

Correctness for the algorithm without delegation is the exact same as for the construction from Section 5.3.2. We detail the distribution of the new delegation algorithms in the first game of the unforgeability proof in Section 7.2.2, just after stating the security theorem for our construction.

7.2.1 Security Results

About the above ABS with delegation, one can still claim the unforgeability and the perfect anonymity.

Theorem 17 (Existential Unforgeability) *The ABS scheme with delegation described in Section 7.2 is existentially unforgeable under the collision-resistance of the hash functions $\mathcal{H}, \mathcal{H}'$ and the SXDH assumption.*

Theorem 18 (Perfect Anonymity) *The ABS scheme with delegation described in Section 7.2 is perfectly anonymous.*

7.2.2 Security Proofs

We start by the anonymity result, as this will allow to perfectly simulate signing queries in the proof of unforgeability.

Perfect Anonymity. We prove anonymity with the same alternative signing algorithm AltSig as in the anonymity proof of Theorem 11, that uses the master secret key instead of an individual signing key. As an additional important note for the incoming EUF proof, we note that, should it be necessary, AltSig would also be able to return the value $\mathbf{r}_3^* = \delta' \cdot \mathbf{h}_3^*$, which will be useful to simulate answer to ODelegatePolicy queries, and output $\mathbf{r}_3'^*$.

Existential unforgeability We can reduce the EUF proof for our ABS with delegation to our proof for simple ABS. To do this, we need to prove that each new oracle can be simulated with oracles from the former proof: keys from ODelegateAttributes can be simulated with OKeyGen,signatures from ODelegateSig can be simulated with AltSig, and policy keys from ODelegatePolicy can be simulated with AltSig. For this reason, we will use the same notation id to count ODelegateAttributes and OKeyGen queries, and the notation i to count OSig, ODelegatePolicy and ODelegateSig.

Game $\mathbf{G}_0$: Setup, KeyGen, Sig and Verif work exactly as in game $\mathbf{G}_0$.

We detail here the distribution of the new algorithms output. Keys generated by the algorithm Delegate-Attributes, for user id and subset Γ', follow the distribution

$$\begin{aligned}
&&\mathbf{r}^*_{\mathsf{id},1} &= (\alpha\delta_{\mathsf{id}}, 0, 0, \psi_{\mathsf{id},1}, 0^4)_{\mathbb{H}^*} \\
\mathbf{r}^*_{\mathsf{id},2} &= (0, \alpha\delta_{\mathsf{id}}, 0, \psi_{\mathsf{id},2}, 0^4)_{\mathbb{H}^*} &\mathbf{r}^*_{\mathsf{id},3} &= (0, 0, \alpha\delta_{\mathsf{id}}, \psi_{\mathsf{id},3}, 0^4)_{\mathbb{H}^*} \\
\mathbf{k}^*_{\mathsf{id},0} &= (\alpha\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0^2)_{\mathbb{B}^*} &\mathbf{k}^*_{\mathsf{id},t} &= (\alpha\delta_{\mathsf{id}}, \alpha\pi_{\mathsf{id},t}(1,t), \phi_{\mathsf{id},t}, 0^6)_{\mathbb{D}^*} \qquad \forall t \in \Gamma'
\end{aligned}$$

for random $\alpha, \delta_{\mathsf{id}}, \psi_{\mathsf{id},1}, \psi_{\mathsf{id},2}, \psi_{\mathsf{id},3}, \phi_{\mathsf{id},0}, (\phi_{\mathsf{id},t})_t, (\pi_{\mathsf{id},t})_t \xleftarrow{\$} \mathbb{Z}_q$ for $t \in \Gamma'$.

The i-th policy keys generated by the Delegate-Policy algorithm, for user id and access-tree $\mathcal{T}$, follow the distribution, for $H = \mathcal{H}(\mathcal{T})$:

$$\begin{aligned}
U_i^* &= (\delta_i, \zeta_i, 0^2)_{\mathbb{B}^*} &S_\lambda^* &= ((\alpha_{\lambda,i}\delta_i + \beta_{\lambda,i}), \omega_{i,\lambda}(1, t_\lambda), q_{i,\lambda}, 0^6)_{\mathbb{D}^*} \\
V^* &= (\delta_i \cdot (1, H, 0), \nu_i, 0^4)_{\mathbb{H}^*} &\mathbf{r}_3^* &= (\delta_i \cdot (0, 0, 1), \psi'_{i,3}, 0^4)_{\mathbb{H}^*}
\end{aligned}$$

for random scalars $\delta_i, \zeta_i, (q_{i,\lambda})_\lambda, (\omega_{i,\lambda})_\lambda, \nu_i, \psi_{i,3} \xleftarrow{\$} \mathbb{Z}_q$ for $\lambda \in \mathcal{L}_\mathcal{T}$.

Signatures generated via policy keys with the DelegateSig algorithm for the i-th signature are generated as, for $H_i = \mathcal{H}(\mathcal{T}_i)$, $H'_i = \mathcal{H}'(m_i)$:

$$\begin{aligned}
U^* &= (\delta_i, \zeta_i, 0^2)_{\mathbb{B}^*} &S_\lambda^* &= ((\alpha_{\lambda,i}\delta_i + \beta_{\lambda,i}), \omega_{\lambda,i}(1, t_\lambda), q_{i,\lambda}, 0^6)_{\mathbb{D}^*} \\
V^* &= (\delta_i \cdot (1, H_i, H'_i), \nu_i, 0^4)_{\mathbb{H}^*}
\end{aligned}$$

for random scalars $\delta_i, \zeta_i, \nu_i, (q_{i,\lambda})_\lambda, (\omega_{i,\lambda})_\lambda \xleftarrow{\$} \mathbb{Z}_q^*$, and still $(\alpha_{\lambda,i})$ a 1-labeling of $\mathcal{T}^*$, and $(\beta_{\lambda,i})$ a random 0-labeling of $\mathcal{T}^*$.

Game $\mathbf{G}_1$: We simulate all Delegate-Attributes queries using KeyGen exclusively. When the adversary ask for Delegate-Attributes on the set Γ' from another key $\mathsf{SK}_{\mathsf{id},\Gamma}$, where $\Gamma' \subset \Gamma$, we simulate the answer as $\mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}', \Gamma')$ for a new random id'. As the correctness analysis has shown, the distribution between original keys and delegated keys is exactly the same, hence: $\mathsf{Adv}_0 = \mathsf{Adv}_1$

Game $\mathbf{G}_2$: We simulate all Sig and DelegateSig queries with the AltSig algorithm. Queries for the i-th Sig, or the i-th signature with DelegateSig, on access-tree $\mathcal{T}_i$ and message m_i, is simulated as, for $H_i = \mathcal{H}(\mathcal{T}_i), H'_i = \mathcal{H}'_i(m_i)$:

$$U_i^* = (\delta_i, \zeta_i, 0^2)_{\mathbb{B}^*} \qquad S_{i,\lambda}^* = (\delta_i \beta'_{i,\lambda}, \gamma_{\lambda,i}(1, t_\lambda), q_{i,\lambda}, 0, ^6)_{\mathbb{D}^*}$$
$$V_i^* = (\delta_i \cdot (1, H_i, H'_i), \nu_i, 0^4)_{\mathbb{H}^*}$$

where $(\beta'_\lambda)_\lambda$ is a random 1-labeling of $\mathcal{T}^*$ and $(\gamma_{\lambda,i}) \xleftarrow{\$} \mathbb{Z}_q^*$. As shown in the above perfect anonymity proof, the distribution is exactly the same, hence the simulation is perfect: $\mathsf{Adv}_1 = \mathsf{Adv}_2$.

Game $\mathbf{G}_3$: We simulate the Delegate-Policy queries with the AltSig algorithm, with only a simple tweak on the element V^*. Queries for the i-th Delegate-Policy, on identity $\bar{\mathsf{id}}$, access-tree $\mathcal{T}_i$ and message m_i, is simulated by calling $\mathsf{AltSig}(\mathsf{MK}, m, \mathcal{T}_i)$ for a random m to get back $(U_i^*, V_i^*, (S_{i,\lambda}^*)_{\lambda \in \mathcal{L}_{\mathcal{T}_i}})$. We then set $\mathbf{r}_{i,3}^* = \delta_i \cdot \mathbf{h}_3^* + \psi_{i,3} \mathbf{h}_4^*$ and $V_i'^* = V_i^* - H'_i \cdot \mathbf{r}_{i,3}^*$, for $\psi_{i,3} \xleftarrow{\$} \mathbb{Z}_q^*$: $V_i'^* = V_i^* - H'_i \cdot \mathbf{r}_{i,3}^* = (\delta_i \cdot (1, H_i, H'_i), \nu_i, 0^4)_{\mathbb{H}^*} - (\delta_i \cdot (0, 0, H'_i), \psi_{i,3} H'_i, 0^4)_{\mathbb{H}^*}$, which is thus as $(\delta_i \cdot (1, H_i, 0), \nu'_i, 0^4)_{\mathbb{H}^*}$, for $\nu'_i \xleftarrow{\$} \mathbb{Z}_q^*$. To properly simulate these queries, we only need to ensure that we can simulate $\mathbf{r}_{i,3}^*$ in all games where we modify the answers of the signature oracle. We finally output the policy key from the query as: $(U_i^*, V_i'^*, \mathbf{r}_{i,3}^*, (S_{i,\lambda}^*)_{\lambda \in \mathcal{L}_{\mathcal{T}_i}})$. Once again the simulation is perfect: $\mathsf{Adv}_2 = \mathsf{Adv}_3$.

Then, we are in a similar game as $\mathbf{G}_0$ for the Existential Unforgeability proof of Theorem 10. The sequence of games can continue the same way, with thus the same security bounds. The only difference is we need to ensure we can properly simulate $\mathbf{r}_{i,3}^*$ in all games.

7.3 Traceable ABS

7.3.1 Definition of Traceable ABS and Security Model

A common property of signature scheme is traceability of the signer, as in group signatures [Cv91, BMW03], where an opener is able to trace back the signer of a message and prove it. For the sake of simplicity, our tracing scheme doesn't include the delegation features we introduced above, but we claim that both this scheme and the delegation one are fully compatible, to make a scheme with delegation and tracing.

We extend the initial definition of an ABS, with the algorithms Setup (with additional tracing key TK and verification key VK), KeyGen, Sig, and Verif, to also consider the Trace and Judge algorithms:

$\mathsf{Trace}(\mathsf{TK}, m, \mathcal{P}, \sigma)$. Given the tracing key TK and a valid signature σ on $(m, \mathcal{P})$, the algorithm outputs the identity id of the signer together with a proof π, both set to $\perp$ in case of failure.

$\mathsf{Judge}(\mathsf{VK}, m, \sigma, \mathsf{id}, \pi)$. Given the verification key VK, a signature σ for a message m, and a proof π that user id generated $(m, \mathcal{P}, \sigma)$, the algorithm outputs 1 if π is valid or 0 else.

Correctness, unforgeability and anonymity are the same as for a regular ABS (see Definitions 6 and 7), except that anonymity cannot be perfect, but computational. We also ask for any valid

signature to be traced back to its signer, with a convincing proof π, either for a judge or anybody when VK is public.

Definition 18 (Traceability) Traceability for ABS is defined by the following game between the adversary and a challenger:

Initialize: The challenger runs the Setup algorithm of ABS and gives the public parameters PK to the adversary;

Oracles: The following oracles can be called in any order and any number of times.

$\mathsf{OKeyGen}(\mathsf{id}, \Gamma)$**:** to model KeyGen-queries for any identity id and any set of attributes Γ of its choice, and the adversary gets back the key $\mathsf{SK}_{\mathsf{id},\Gamma}$;

$\mathsf{OSig}(\mathsf{id}, m, \mathcal{P})$**:** to model Sig-queries for any identity id and under any policy $\mathcal{P}$ of its choice for a message m, and the adversary gets the signature σ;

Finalize(b')**:** The adversary outputs a signature $(m', \mathcal{P}', \sigma')$. One asks $(\mathsf{id}, \pi) = \mathsf{Trace}(\mathsf{TK}, \sigma')$. If one of the following is true (non-legitimate attack)

- $(m', \mathcal{P}')$ has been queried to the OSig-oracle;
- id has been queried to the $\mathsf{OKeyGen}$-oracle with Γ such that $\mathsf{Judge}(\mathsf{VK}, m', \sigma', \mathsf{id}, \pi) = 1$ and $\mathcal{P}'(\Gamma) = 1$;

then output 0, otherwise output $\mathsf{Verif}(\mathsf{PK}, m', \mathcal{P}', \sigma')$.

The success $\mathsf{Adv}^{\mathsf{trace}}(\mathcal{A})$ of an adversary $\mathcal{A}$ against traceability is the probability to have 1 as output in this game.

More precisely, we consider the adversary wins the traceability game if it manages to mislead the tracing procedure: by making it either fail or output an honest user (not under the control of the adversary, with a key asked to the key-oracle), or by making the result of the tracing impossible to prove. Of course, we will ignore the output if it exactly corresponds to a signing-query.

We stress that key-queries model corruptions of some signers. In the literature, corruptions can be *static* (all the corrupted users are known before generating the global parameters) or *adaptive* (the corrupted users are chosen adaptively by the adversary during the security game). Even in the adaptive case, we can add some restriction of disjoint sets of identities in key-queries and signing-queries. We will call it the *distinct-user setting*.

7.3.2 One-Time Linearly-Homomorphic Signature

For our traceable ABS scheme derived from the scheme detailed in Section 5.3.2, we will make use of a (One-Time) Linearly-Homomorphic Signature ($\mathsf{OT\text{-}LH}$). Let us first recall the definition.

$\mathsf{KeyGen}(1^\kappa, n)$**.** From the security parameter κ, and a dimension n, the algorithm outputs a signing key sk and a verification key vk;

$\mathsf{Sig}(\mathsf{sk}, m)$**.** For a signing key sk and a message m of dimension n, the algorithm outputs a signature σ;

$\mathsf{DerivSign}(\mathsf{vk}, (\alpha_i, m_i, \sigma_i))$**.** For a verification key vk, several messages m_i together with their signatures σ_i, and some coefficients α_i, the algorithm outputs a signature σ of the linear combination $\sum_i \alpha_i m_i$;

$\mathsf{Verif}(\mathsf{vk}, m, \sigma)$**.** Given a verification key vk, a message m, and a signature σ, the algorithm outputs 1 for accept or 0 for reject;

Correctness and unforgeability are similar as for usual signature schemes, except that an output (m', σ') will be considered a forgery if m' is not in the span of the messages m_i asked to the signing oracle. Indeed, linear combination of signatures on the m_i's is accessible to the adversary. We will then denote by $\mathsf{Adv}^{\mathsf{euf}}_{\mathsf{OT\text{-}LH}}$ the best advantage an adversary can have in generating a valid signature for a message out of the span of the initially signed messages. Furthermore, any signature generated by linear combinations using $\mathsf{DerivSign}$ should be perfectly indistinguishable from a fresh signature generated by Sig.

Such linearly-homomorphic signatures have been proposed in the literature, as in [LPJY13, FHS19]. But we can also use the simplified version from [HPP20] which has been proven in the generic group model, even together with an extractor that provides the coefficients in the linear combination of the initial messages for the new signed message. From this paper, we will also use the following theorem, that states the intractability of the Linear-Square problem:

Theorem 19 (Linear-Square Problem) *Given n Square Diffie-Hellman tuples $(g_i, a_i = g_i^{w_i}, b_i = a_i^{w_i})$, together with w_i, for random $g_i \xleftarrow{\$} \mathbb{G}^*$ and $w_i \xleftarrow{\$} \mathbb{Z}_q^*$, outputting $(\alpha_i)_{i=1,\ldots,n}$ such that $(G = \prod g_i^{\alpha_i}, A = \prod a_i^{\alpha_i}, B = \prod b_i^{\alpha_i})$ is a valid Square Diffie-Hellman, with at least two non-zero coefficients α_i, is computationally hard under the Discrete Logarithm assumption.*

7.4 Description of our Traceable ABS

We consider any $\mathsf{OT\text{-}LH}$ scheme $(\mathsf{KeyGen}', \mathsf{Sig}', \mathsf{DerivSign}', \mathsf{Verif}')$ in $\mathbb{G}_2^n$. We will also use a non-interactive zero-knowledge proof of knowledge $(\mathsf{NIZKPoK\text{-}SqDH}, \mathsf{VERIF\text{-}SqDH})$ of the witness w for a Square Diffie-Hellman tuple $(h_t, h_t^w, h_t^{w^2})$ in $\mathbb{G}_t$ and a non-interactive zero-knowledge proof $(\mathsf{NIZKPoK\text{-}DH}, \mathsf{VERIF\text{-}DH})$ of Diffie-Hellman tuple $(g_t, g_t^w, g_t^\delta, g_t^{\delta w})$ in $\mathbb{G}_t$. For both proofs, one can simply use Schnorr-like proofs with the Fiat-Shamir paradigm [Sch91, FS87]. They are known to be sound, with simulation-extractability in the Random Oracle Model, as well as perfectly zero-knowledge. We refer to [FKMV12] for more details.

We now detail our construction, with access-trees for policies, where we just complete the signing key $\mathbf{k}_0^*$ with a square Diffie-Hellman tuple where one can identify the signer, if and only if the scalar w_{id} is known. The public value $g_t^{w_{\mathsf{id}}}$ associated to user id will then be enough to verify the tracing, without revealing w_{id}:

$\mathsf{Setup}(1^\kappa)$**.** The algorithm chooses three random dual orthogonal bases, in a pairing-friendly setting $\mathcal{PG} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, G_1, G_2, q)$:

$$\mathbb{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_6) \qquad \mathbb{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_{10}) \qquad \mathbb{H} = (\mathbf{h}_1, \ldots, \mathbf{h}_8)$$
$$\mathbb{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_6^*) \qquad \mathbb{D}^* = (\mathbf{d}_1^*, \ldots, \mathbf{d}_{10}^*) \qquad \mathbb{H}^* = (\mathbf{h}_1^*, \ldots, \mathbf{h}_8^*).$$

It also chooses two hash function $\mathcal{H}$ and $\mathcal{H}'$ onto $\mathbb{Z}_q$. The algorithm calls the $\mathsf{OT\text{-}LH}$ signature algorithm $\mathsf{KeyGen}'(1^\kappa, 6)$, for vectors in $\mathbb{G}_2^6$, and gets back the keys sk and vk. It also gets $\Sigma_2 = \mathsf{Sig}'(\mathsf{sk}, \mathbf{b}_2^*)$, and sets the public parameters as $\mathsf{PK} = \{\mathcal{PG}, \mathcal{H}, (\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}_5, \mathbf{b}_6), (\mathbf{b}_2^*, \Sigma_2), (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_5), (\mathbf{d}_1^*, \mathbf{d}_2^*, \mathbf{d}_3^*, \mathbf{d}_4^*), (\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3, \mathbf{h}_5), (\mathbf{h}_4^*), \mathsf{vk}\}$, and the master secret key is set as $\mathsf{MK} = \{(\mathbf{b}_1^*, \mathbf{b}_5^*, \mathbf{b}_6^*), (\mathbf{h}_1^*, \mathbf{h}_2^*, \mathbf{h}_3^*), \mathsf{sk}\}$. Finally, the tracing key TK and the verification key VK are initialized as empty sets.

$\mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}, \Gamma)$**.** Random scalars $\delta_{\mathsf{id}}, w_{\mathsf{id}} \xleftarrow{\$} \mathbb{Z}_q^*$ are associated to id, with

$$\mathbf{k}_0^* = \delta_{\mathsf{id}} \cdot \mathbf{b}_1^* + \phi_0 \cdot \mathbf{b}_2^* + \delta_{\mathsf{id}} \cdot w_{\mathsf{id}} \cdot \mathbf{b}_5^* + \delta_{\mathsf{id}} \cdot w_{\mathsf{id}}^2 \cdot \mathbf{b}_6^*$$
$$\mathbf{k}_t^* = \delta_{\mathsf{id}} \cdot \mathbf{d}_1^* + \pi_t \cdot (\mathbf{d}_2^* + t \cdot \mathbf{d}_3^*) + \phi_t \cdot \mathbf{d}_4^*$$
$$\mathbf{r}_1^* = \delta_{\mathsf{id}} \cdot \mathbf{h}_1^* + \psi_1 \cdot \mathbf{h}_4^* \qquad \mathbf{r}_2^* = \delta_{\mathsf{id}} \cdot \mathbf{h}_2^* + \psi_2 \cdot \mathbf{h}_4^* \qquad \mathbf{r}_3^* = \delta_{\mathsf{id}} \cdot \mathbf{h}_3^* + \psi_3 \cdot \mathbf{h}_4^*$$

for all attributes $t \in \Gamma$, with $\phi_0, (\phi_t)_t, (\pi_t)_t \xleftarrow{\$} \mathbb{Z}_q^*$ for each t. The algorithm calls for $\Sigma_{\mathsf{id}} = \mathsf{Sig}'(\mathsf{sk}, \mathbf{k}_0^*)$. The signing key $\mathsf{SK}_{\mathsf{id},\Gamma}$ is set as $(w_{\mathsf{id}}, \mathbf{k}_0^*, \Sigma_{\mathsf{id}}, (\mathbf{k}_t^*)_{t\in\Gamma}, \mathbf{r}_1^*, \mathbf{r}_2^*, \mathbf{r}_3^*)$, for id. It can be completed later for new attributes, but only by using the same δ_{id}. The pair $(\mathsf{id}, w_{\mathsf{id}})$ is appended to TK, and $(\mathsf{id}, g_t^{w_{\mathsf{id}}})$ is appended to VK.

$\mathsf{Sig}(\mathsf{SK}_{\mathsf{id},\Gamma}, m, \mathcal{T})$. Let $\mathcal{T}' \in \mathsf{EPT}(\mathcal{T}, \Gamma)$ be an Evaluation Pruned Tree, $\nu, \xi, \zeta \xleftarrow{\$} \mathbb{Z}_q^*$. Compute the following 1-labeling of the dual tree $\mathcal{T}^*$: for each leaf λ, choose $\alpha_\lambda = 1$ if $\lambda \in \mathcal{L}_{\mathcal{T}'}$, else $\alpha_\lambda = 0$. Then, choose a random 0-labeling (β_λ) of $\mathcal{T}^*$, and $(q_\lambda)_\lambda, (\omega_\lambda)_\lambda$ random scalars, and set, for $H = \mathcal{H}(\mathcal{T}), H' = \mathcal{H}'(m)$:

$$U^* = \xi \mathbf{k}_0^* + \zeta \mathbf{b}_2^* \qquad S_\lambda^* = \alpha_\lambda \xi \cdot \mathbf{k}_{t_\lambda}^* + \beta_\lambda \mathbf{d}_1^* + \omega_\lambda(\mathbf{d}_2^* + t_\lambda \cdot \mathbf{d}_3^*) + q_\lambda \cdot \mathbf{d}_4^*$$
$$V^* = \xi(\mathbf{r}_1^* + H \cdot \mathbf{r}_2^* + H' \cdot \mathbf{r}_3^*) + \nu \cdot \mathbf{h}_4^*$$

for all the leaves λ, where t_λ is the associated attribute of λ. From the linearly-homomorphic property, one can compute a signature on $U^* \in \mathbb{G}_2^6$:

$$\Sigma = \mathsf{DerivSign}'(\mathsf{vk}, ((\xi, \mathbf{k}_0^*, \Sigma_{\mathsf{id}}), (\zeta, \mathbf{b}_2^*, \Sigma_2))$$

Eventually, using w_{id}, one can generate the proof of Square Diffie-Hellman tuple :

$$\Pi = \mathsf{NIZKPoK\text{-}SqDH}(w_{\mathsf{id}}, (e(\mathbf{b}_1, U^*), e(\mathbf{b}_5, U^*), e(\mathbf{b}_6, U^*)))$$

as this tuple is equal to $(h_t, h_t^{w_{\mathsf{id}}}, h_t^{w_{\mathsf{id}}^2})$, for some $h_t \in \mathbb{G}_t$. The final signature consists of the tuple $\sigma = (U^*, V^*, (S_\lambda^*)_\lambda), \Sigma, \Pi)$.

$\mathsf{Verif}(\mathsf{PK}, m, \mathcal{T}, \sigma)$. Let $\kappa, \kappa_0, (\kappa_\lambda)_\lambda, s, s_0, \theta, \theta' (\theta_\lambda)_\lambda \xleftarrow{\$} \mathbb{Z}_q$. Let $(s_\lambda)_\lambda$ be a random s_0-labeling of $\mathcal{T}$, then set, for $\bar{H} = \mathcal{H}(\mathcal{T}), \bar{H}' = \mathcal{H}'(\mathcal{T})$:

$$u = -(s_0 + s) \cdot \mathbf{b}_1 + \kappa_0 \cdot \mathbf{b}_3 \qquad c_\lambda = s_\lambda \cdot \mathbf{d}_1 + \theta_\lambda t_\lambda \cdot \mathbf{d}_2 - \theta_\lambda \cdot \mathbf{d}_3 + \kappa_\lambda \cdot \mathbf{d}_5$$
$$v = (s + \theta \bar{H} + \theta' \bar{H}') \cdot \mathbf{h}_1 - \theta \cdot \mathbf{h}_2 - \theta' \cdot \mathbf{h}_3 + \kappa \cdot \mathbf{h}_5$$

Accept if $e(\mathbf{b}_1, U^*) \neq 1_{\mathbb{G}_t}$ and $e(u, U^*) \cdot e(v, V^*) \cdot \prod e(c_\lambda, S_\lambda^*) = 1_{\mathbb{G}_t}$, but also if $\mathsf{Verif}'(\mathsf{vk}, U^*, \Sigma) = 1$ and $\mathsf{VERIF\text{-}SqDH}((e(\mathbf{b}_1, U^*), e(\mathbf{b}_5, U^*), e(\mathbf{b}_6, U^*)), \Pi) = 1$, otherwise reject.

$\mathsf{Trace}(\mathsf{TK}, \sigma')$. Compute $B_1 = e(\mathbf{b}_1, U^*)$ and $B_2 = e(\mathbf{b}_5, U^*)$. Then, for $(\mathsf{id}, w_{\mathsf{id}}) \in \mathsf{TK}$, check until $B_1^{w_{\mathsf{id}}} = B_2$. When the equality holds, generate the proof

$$\pi = \mathsf{NIZKPoK\text{-}DH}(g_t, g_t^{w_{\mathsf{id}}}, (e(\mathbf{b}_1, U^*), e(\mathbf{b}_5, U^*)))$$

and output (id, π). Otherwise output $\perp$.

$\mathsf{Judge}(\mathsf{VK}, m, \sigma', \mathsf{id}, \pi)$. Extract $g_t^{w_{\mathsf{id}}}$ corresponding to id from VK and output :

$$\mathsf{VERIF\text{-}DH}(g_t, g_t^{w_{\mathsf{id}}}, (e(\mathbf{b}_1, U^*), e(\mathbf{b}_5, U^*)), \pi)$$

.

As VK can be a public list, anybody can run the Judge algorithm.

Correctness This construction is a slight variation of the previous ABS scheme:

$\mathsf{KeyGen}(\mathsf{MK}, \mathsf{id}, \Gamma)$. The difference is only in the 5-th and 6-th positions of $\mathbf{k}_0^*$, with $\delta_{\mathsf{id}} \cdot w_{\mathsf{id}}$ and $\delta_{\mathsf{id}} \cdot w_{\mathsf{id}}^2$, which will help for the tracing procedure of the generated signature: $\mathbf{k}_0^* = (\delta_{\mathsf{id}}, \phi_0, 0, 0, \delta_{\mathsf{id}} \cdot w_{\mathsf{id}}, \delta_{\mathsf{id}} \cdot w_{\mathsf{id}}^2)_{\mathbb{B}^*}$.

$\mathsf{Sig}(\mathsf{SK}_{\mathsf{id},\Gamma}, m, \mathcal{T})$. The first difference is again in the 5-th and 6-th positions of U^*, with $\xi\delta_{\mathsf{id}} \cdot w_{\mathsf{id}}$ and $\xi\delta_{\mathsf{id}} \cdot w_{\mathsf{id}}^2$, which will be used in the tracing procedure: $U^* = (\xi\delta_{\mathsf{id}}, \xi\phi_0 + \zeta, 0, 0, \xi\delta_{\mathsf{id}} \cdot w_{\mathsf{id}}, \xi\delta \cdot w_{\mathsf{id}}^2)_{\mathbb{B}^*}$. There are also the two additional elements to provide the collusion-resistance, with the signature Σ on U^* that can be built thanks to the linear property of the signature, and the proof Π that ensure the presence of a square Diffie-Hellman tuple.

$\mathsf{Verif}(\mathsf{PK}, m, \mathcal{T}, \sigma)$. The verification is the same as before, with vectors u, v, and c_λ, to check the previous relations. One additionally checks the signature Σ and the square Diffie-Hellman tuple $(e(\mathbf{b}_1, U^*), e(\mathbf{b}_5, U^*), e(\mathbf{b}_6, U^*))$ in $\mathbb{G}_t$.

7.4.1 Security Results

Since the verification process is even more restrictive than in the previous scheme, one can claim the same unforgeability result:

Theorem 20 (Existential Unforgeability) *The ABS scheme described in Section 7.4 is existentially unforgeable under the collision-resistance of the hash functions $\mathcal{H}, \mathcal{H}'$ and the SXDH assumption, according to the Definition 6.*

Because of the additional elements in the signature (which are useful for tracing), the signature is no longer perfectly anonymous, but still computationally anonymous:

Theorem 21 (Computational Anonymity) *The ABS scheme described in Section 7.4 is computationally anonymous, according to the Definition 7, when w_0 and w_1, in SK_0 and SK_1, are unknown, under the Decisional Square Diffie-Hellman assumption in $\mathbb{G}_2$, and the perfect zero-knowledge of the* NIZKPoK-SqDH *in the ROM.*

Proof The additional elements are the Square Diffie-Hellman tuple in the 1-st, 5-th, and 6-th components of $U^* = (\delta', \phi'_0, 0, 0, \delta' \cdot w_{\mathsf{id}}, \delta' \cdot w_{\mathsf{id}}^2)_{\mathbb{B}^*}$, the signature Σ, and the proof Π.

The Square Diffie-Hellman tuple in U^* can be generated from a Square Diffie-Hellman tuple $(\delta' G_2, w \cdot \delta' G_2, w^2 \cdot \delta' G_2) \in \mathbb{G}_2^3$. Under the Decisional Square Diffie-Hellman assumption in $\mathbb{G}_2$, such a tuple is indistinguishable from a random tuple in $\mathbb{G}_2^3$. This makes U^* generated from w_0 or w_1 indistinguishable when those scalars are unknown. Since Σ is a signature of U^* that is itself indistinguishable for w_0 and w_1, Σ is also indistinguishable for w_0 and w_1. Eventually, Π being a zero-knowledge proof on the above tuple, it can be simulated without knowing the witness. It thus does not leak any additional information. Hence the anonymity under the Decisional Square Diffie-Hellman assumption in $\mathbb{G}_2$.

We can state the traceability result.

Theorem 22 (Traceability) *The ABS scheme described in Section 7.4 is traceable in the ROM, in the distinct-user setting, under the security of the OT-LH signature scheme, the intractability of the Linear-Square problem, the simulation-extractability of the* NIZKPoK-SqDH*, and the soundness of the* NIZKPoK-DH*, according to the Definition 18.*

7.4.2 Proof of the Traceability

In this proof, we will first recall the way we can simulate the keys and the signatures. Then, we will show how the linearly-homomorphic signature and the Linear-Square problem will prevent attacks:

Game $\mathbf{G}_0$: As shown in the previous section, keys generated by the KeyGen algorithm, for user id, follow the distribution:

$$\mathbf{k}^*_{\mathsf{id},0} = (\delta_{\mathsf{id}}, \phi_{\mathsf{id},0}, 0, 0, \delta_{\mathsf{id}} \cdot w_{\mathsf{id}}, \delta_{\mathsf{id}} \cdot w_{\mathsf{id}}^2)_{\mathbb{B}^*} \qquad \mathbf{r}^*_{\mathsf{id},1} = (\delta_{\mathsf{id}}, 0, 0, \psi_{\mathsf{id},1}, 0, 0, 0, 0)_{\mathbb{H}^*}$$
$$\mathbf{k}^*_{\mathsf{id},t} = (\delta_{\mathsf{id}}, \pi_{\mathsf{id},t}(1,t), \phi_{\mathsf{id},t}, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*} \qquad \mathbf{r}^*_{\mathsf{id},2} = (0, \delta_{\mathsf{id}}, 0, \psi_{\mathsf{id},2}, 0, 0, 0, 0)_{\mathbb{H}^*}$$
$$\Sigma_{\mathsf{id}} = \mathsf{Sig}'(\mathsf{sk}, \mathbf{k}^*_{\mathsf{id},0}) \qquad \mathbf{r}^*_{\mathsf{id},3} = (0, 0, \delta_{\mathsf{id}}, \psi_{\mathsf{id},3}, 0, 0, 0, 0)_{\mathbb{H}^*}$$

The i-th signature generated by the Sig algorithm follows the distribution

$$U_i^* = (\xi_i \delta_i, \zeta_i, 0, 0, \xi_i \delta_i \cdot w_i, \xi_i \delta_i \cdot w_i^2)_{\mathbb{B}^*} \quad V_i^* = (\xi_i \delta_i (1, H_i, H'_i), \nu_i, 0, 0, 0, 0)_{\mathbb{H}^*}$$
$$S^*_{i,\lambda} = (\beta'_{i,\lambda}, \gamma_{i,\lambda}(1, t_\lambda), q_{i,\lambda}, 0, 0, 0, 0, 0, 0)_{\mathbb{D}^*}$$

where $\delta_i, w_i, \mathbf{k}^*_{i,0}, \Sigma_i$ correspond to new and fresh $\delta_{\mathsf{id}}, w_{\mathsf{id}}, \mathbf{k}^*_{\mathsf{id},0}, \Sigma_{\mathsf{id}}$ of the signer id (for an implicitly freshly generated key as signing queries and key queries should not correspond to

the same identities, as we are in the distinct-user setting), and $H_i = \mathcal{H}(\mathcal{T}_i), H'_i = \mathcal{H}'(m_i)$, together with

$$\Sigma = \mathsf{DerivSign}'(\mathsf{vk}, ((\xi_i, \mathbf{k}^*_{i,0}, \Sigma_i), (\zeta_i, \mathbf{b}^*_2, \Sigma_2))$$
$$\Pi = \mathsf{NIZKPoK\text{-}SqDH}(w_i, (e(\mathbf{b}_1, U^*), e(\mathbf{b}_5, U^*), e(\mathbf{b}_6, U^*)))$$

For the decision of the challenge signature $\sigma = (U^*, V^*, (S^*_\lambda)_\lambda, \Sigma, \Pi)$ on message m and policy $\mathcal{T}$, different from any query-answer to the signing oracle, one uses

$$u = (-s_0 - s, 0, \kappa_0, 0, 0, 0)_{\mathbb{B}} \quad v = (s + \theta \cdot \bar{H} + \theta' \cdot \bar{H}', -\theta, -\theta', 0, \kappa, 0, 0, 0)_{\mathbb{H}}$$
$$c_\lambda = (s_\lambda, \theta_\lambda(t_\lambda, -1), 0, \kappa_\lambda, 0, 0, 0, 0, 0)_{\mathbb{D}}$$

where $(\bar{H}, \bar{H}') \neq (H_i, H'_i)$ for all i. Instead of outputting just the decision, one can consider the challenger outputs $(u, v, (c_\lambda)_\lambda)$, and everybody can make the final verification, with $B_1 = e(\mathbf{b}_1, U^*)$, $B_2 = e(\mathbf{b}_5, U^*)$, and $B_3 = e(\mathbf{b}_6, U^*)$:

$$e(\mathbf{b}_1, U^*) \neq 1_{\mathbb{G}_t} \qquad e(u, U^*) \cdot e(v, V^*) \cdot \prod e(c_\lambda, S^*_\lambda) = 1_{\mathbb{G}_t}$$
$$\mathsf{Verif}'(\mathsf{vk}, U^*, \Sigma) = 1 \qquad \mathsf{VERIF\text{-}SqDH}((B_1, B_2, B_3), \Pi) = 1$$

For the tracing procedure, one checks $B_1^{w_{\mathsf{id}}} = B_2$ for all the id's asked to the key generation oracle. If no w_{id} matches, we output 1. Otherwise we output 0. We denote by Adv_0 the probability to output 1 in this game.

Game $\mathbf{G}_1$: We replace Σ by a fresh signature (during the signing queries), $\Sigma = \mathsf{Sig}'(\mathsf{sk}, U^*)$, as signatures from $\mathsf{DerivSign}'$ and Sig' follows perfectly indistinguishable distributions in an $\mathsf{OT\text{-}LH}$ signature scheme: $\mathsf{Adv}_0 = \mathsf{Adv}_1$.

Game $\mathbf{G}_2$: The simulator generates the proofs Π, during the signing queries only, with the zero-knowledge simulator. Thanks to the (perfect) zero-knowledge property, this game is indistinguishable from the previous one: $\mathsf{Adv}_1 = \mathsf{Adv}_2$. Now, the simulator does not need to know the scalars w_{id} for signing queries, but only for key queries.

Game $\mathbf{G}_3$: The simulator generates the signatures using Square Diffie-Hellman tuples (h_i, h'_i, h''_i), with unknown scalars w_i: $\mathsf{Adv}_2 = \mathsf{Adv}_3$.

Game $\mathbf{G}_4$: In this game, we always output 0, meaning the tracing procedure always succeeds, and so $\mathsf{Adv}_4 = 0$.

Let us study the gap:

- If we consider $\mathsf{OSig}'(m)$ the signature oracle in the EUF security game of the $\mathsf{OT\text{-}LH}$ scheme, under the unforgeability result on Σ, the U^* of the output signature $(m', \mathcal{T}', \sigma)$ of our adversary necessarily involves a linear combination of the U^*_i, which implies a linear combination of the $\mathbf{k}^*_{\mathsf{id},0}$ and $\mathbf{b}^*_2$. Using the signature from [HPP20], we even get the coefficients of this linear combination;
- If we consider the 1-st, 5-th and 6-th components, which constitute a Square Diffie-Hellman tuple with an exponent that is a (known) linear combination of the scalars involved in the keys or signatures, the Theorem 19 implies that there is necessarily either a w_i involved in a signing-query or a w_{id} involved in a key-query in this U^*;
- With the additional proof of knowledge Π, from the simulation-extractability, one can extract this scalar: which is as hard as breaking the Decisional Square Diffie-Hellman if this is a w_i from a signing-query, because of the challenge (h_i, h'_i, h''_i), which could either be a random tuple or a Square Diffie-Hellman tuple.

As a consequence, except with probability bounded by $\mathsf{Adv}^{\mathsf{euf}}_{\mathsf{OT\text{-}LH}}(t) + \mathsf{Adv}^{\mathsf{lsqp}}(t) + \mathsf{Adv}^{\mathsf{dsqdh}}(t)$, this is necessarily a w_{id} from a key-query, which can thus be extracted by the tracing algorithm: $\mathsf{Adv}_3 - \mathsf{Adv}_4 \leq \mathsf{Adv}^{\mathsf{euf}}_{\mathsf{OT\text{-}LH}}(t) + \mathsf{Adv}^{\mathsf{lsqp}}(t) + \mathsf{Adv}^{\mathsf{dsqdh}}(t)$.

Chapter

8 Conclusion

In this thesis, we studied attribute-based constructions in cryptography with the DPVS framework, and in particular how to make these constructions delegatable and traceable.

We first introduced Attribute-Based Encryption and Attribute-Based Signature, in particular we showed how these primitives can guarantee cryptographic access-control based on policies expressed as logical access-trees.

Then, we presented our new approach to the DPVS framework, originally designed by Okamoto and Takashima. In particular, we provided a simple generic toolbox that can be used to prove constructions in the DPVS. This work is proved under the SXDH assumption, which is the DDH assumption on the two source groups of a pairing (typically of type 3). We also proved a new theorem for unbounded universe of attributes, that we use in our later constructions.

Finally, we detailed our concrete ABE and ABS constructions, with the DPVS framework, in two steps. The first step was to prove the original constructions of Okamoto and Takashima with our new setting and a few adjustments. Despite their similarities, a few tweaks had to be considered to achieve adaptive security on both schemes, in particular the new theorem for unbounded universe of attributes that we mentioned above. The second step was to augment these base constructions with our new features, which we separate in two paragraphs in the following.

Concerning encryption, we augmented the original construction into a new primitive called SA-KP-ABE, for Key-Policy ABE with Switchable Attributes. This original approach allows the central authority to deactivate the attributes of the messages and keys of the users in an indistinguishable manner. As a direct application, we showed how to construct a traceable ABE from our primitive by using the traditional fingerprinting approach, where each user is associated to a unique fingerprint that is embedded inside his keys. Our new method incurs better performances than the standard ABE to traceable ABE conversion regarding the size of the keys, which is generally inefficient.

Concerning signature, we designed two modular approaches, one for delegation, one for signature, which are suitable to be used together or separately, depending on the need. In particular, our delegation approach allows to delegate, in a totally independent manner, pre-signed policies that can then be used to sign messages exclusively, or delegated keys that are subset of the original keys for the attributes. Our tracing approach is also different from the usual tracing for signatures, as we use a One-Time Linearly Homomorphic Signature scheme to make the user prove he didn't tamper with his keys in an unintended manner to make a valid signature.

Bibliography

[AKPS12] Murat Ak, Aggelos Kiayias, Serdar Pehlivanoglu, and Ali Aydin Selcuk. Generic construction of trace and revoke schemes. Cryptology ePrint Archive, Report 2012/531, 2012. https://eprint.iacr.org/2012/531. 14

[AT20] Nuttapong Attrapadung and Junichi Tomida. Unbounded dynamic predicate compositions in ABE from standard assumptions. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 405–436. Springer, Heidelberg, December 2020. 14

[Att16] Nuttapong Attrapadung. Dual system encryption framework in prime-order groups via computational pair encodings. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 591–623. Springer, Heidelberg, December 2016. 13

[BCC+15] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, Jens Groth, and Christophe Petit. Short accountable ring signatures based on DDH. In Günther Pernul, Peter Y. A. Ryan, and Edgar R. Weippl, editors, *ESORICS 2015, Part I*, volume 9326 of *LNCS*, pages 243–265. Springer, Heidelberg, September 2015. 15

[BDJR97] Mihir Bellare, Anand Desai, Eric Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *38th FOCS*, pages 394–403. IEEE Computer Society Press, October 1997. 73

[BF03] Dan Boneh and Matthew K. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003. 2, 10

[BMW03] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 614–629. Springer, Heidelberg, May 2003. 15, 116

[BN08] Dan Boneh and Moni Naor. Traitor tracing with constant size ciphertext. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM CCS 2008*, pages 501–510. ACM Press, October 2008. 13, 110

[BS95] Dan Boneh and James Shaw. Collusion-secure fingerprinting for digital data (extended abstract). In Don Coppersmith, editor, *CRYPTO'95*, volume 963 of *LNCS*, pages 452–465. Springer, Heidelberg, August 1995. 13, 15

[BSW06] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 573–592. Springer, Heidelberg, May / June 2006. 13

[BSW07] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *2007 IEEE Symposium on Security and Privacy*, pages 321–334. IEEE Computer Society Press, May 2007. 3, 11, 13

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, March 2011. 2, 10

[BW06] Dan Boneh and Brent Waters. A fully collusion resistant broadcast, trace, and revoke system. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 211–220. ACM Press, October / November 2006. 13, 15

[CFN94] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 257–270. Springer, Heidelberg, August 1994. 108

[CGW15] Jie Chen, Romain Gay, and Hoeteck Wee. Improved dual system ABE in prime-order groups via predicate encodings. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 595–624. Springer, Heidelberg, April 2015. 13, 14

[CGW18] Jie Chen, Junqing Gong, and Hoeteck Wee. Improved inner-product encryption with adaptive security and full attribute-hiding. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 673–702. Springer, Heidelberg, December 2018. 6, 14, 17

[CLL+13] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. Shorter IBE and signatures via asymmetric pairings. In Michel Abdalla and Tanja Lange, editors, *PAIRING 2012*, volume 7708 of *LNCS*, pages 122–140. Springer, Heidelberg, May 2013. 27

[CPP17] Jérémy Chotard, Duong Hieu Phan, and David Pointcheval. Homomorphic-policy attribute-based key encapsulation mechanisms. In Phong Q. Nguyen and Jianying Zhou, editors, *ISC 2017*, volume 10599 of *LNCS*, pages 155–172. Springer, Heidelberg, November 2017. 3, 11

[Cv91] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer, Heidelberg, April 1991. 2, 10, 15, 116

[DGM18] Constantin Catalin Dragan, Daniel Gardham, and Mark Manulis. Hierarchical attribute-based signatures. In Jan Camenisch and Panos Papadimitratos, editors, *CANS 18*, volume 11124 of *LNCS*, pages 213–234. Springer, Heidelberg, September / October 2018. 15, 16

[DGP22] Cécile Delerablée, Lénaïck Gouriou, and David Pointcheval. Key-policy ABE with switchable attributes. In Clemente Galdi and Stanislaw Jarecki, editors, *The 13th Conference on Security in Communication Networks (SCN '22)*, volume 13409 of *LNCS*, pages 147–171, Amalfi, Italy, 2022. Springer, Heidelberg. https://eprint.iacr.org/2021/867. 51

[DOT18] Pratish Datta, Tatsuaki Okamoto, and Katsuyuki Takashima. Adaptively simulation-secure attribute-hiding predicate encryption. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 640–672. Springer, Heidelberg, December 2018. 13

[DOT19] Pratish Datta, Tatsuaki Okamoto, and Katsuyuki Takashima. Efficient attribute-based signatures for unbounded arithmetic branching programs. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part I*, volume 11442 of *LNCS*, pages 127–158. Springer, Heidelberg, April 2019. 15

[EGK14] Ali El Kaafarani, Essam Ghadafi, and Dalia Khader. Decentralized traceable attribute-based signatures. In Josh Benaloh, editor, *CT-RSA 2014*, volume 8366 of *LNCS*, pages 327–348. Springer, Heidelberg, February 2014. 15, 70

[FHS19] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. *Journal of Cryptology*, 32(2):498–546, April 2019. 118

[FKMV12] Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. On the non-malleability of the Fiat-Shamir transform. In Steven D. Galbraith and Mridul Nandi, editors, *INDOCRYPT 2012*, volume 7668 of *LNCS*, pages 60–79. Springer, Heidelberg, December 2012. 118

[Fre10] David Mandell Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 44–61. Springer, Heidelberg, May / June 2010. 13

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987. 118

[GKW18] Rishab Goyal, Venkata Koppula, and Brent Waters. Collusion resistant traitor tracing from learning with errors. Cryptology ePrint Archive, Report 2018/346, 2018. https://eprint.iacr.org/2018/346. 13

[GM19] Daniel Gardham and Mark Manulis. Hierarchical attribute-based signatures: Short keys and optimal signature length. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19*, volume 11464 of *LNCS*, pages 89–109. Springer, Heidelberg, June 2019. 7, 8, 16, 18

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309. 1, 2, 3, 5, 6, 10, 11, 13, 14, 15, 16, 17, 20, 23, 24, 38

[Gui13] Aurore Guillevic. Comparing the pairing efficiency over composite-order and prime-order elliptic curves. In Michael J. Jacobson Jr., Michael E. Locasto, Payman Mohassel, and Reihaneh Safavi-Naini, editors, *ACNS 13*, volume 7954 of *LNCS*, pages 357–372. Springer, Heidelberg, June 2013. 13

[HL02] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 466–481. Springer, Heidelberg, April / May 2002. 14

[HPP20] Chloé Hébant, Duong Hieu Phan, and David Pointcheval. Linearly-homomorphic signatures and scalable mix-nets. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 597–627. Springer, Heidelberg, May 2020. 7, 16, 18, 70, 118, 121

[Kha07] Dalia Khader. Attribute based group signatures. Cryptology ePrint Archive, Report 2007/159, 2007. https://eprint.iacr.org/2007/159. 15

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 146–162. Springer, Heidelberg, April 2008. 14

[KY05] Aggelos Kiayias and Moti Yung. Group signatures with efficient concurrent join. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 198–214. Springer, Heidelberg, May 2005. 15

[Lew12] Allison B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 318–335. Springer, Heidelberg, April 2012. 13

[LLLW17] Xiaoyi Li, Kaitai Liang, Zhen Liu, and Duncan S. Wong. Attribute based encryption: Traitor tracing, revocation and fully security on prime order groups. In Donald Ferguson, Víctor Méndez Muñoz, Jorge S. Cardoso, Markus Helfert, and Claus Pahl, editors, *CLOSER 2017*, pages 281–292. SciTePress, 2017. 15

[LOS+10] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 62–91. Springer, Heidelberg, May / June 2010. 13, 27, 38

[LPJY13] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Linearly homomorphic structure-preserving signatures and their applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 289–307. Springer, Heidelberg, August 2013. 118

[LPQ12] Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 206–224. Springer, Heidelberg, May 2012. 14

[LT18] Junzuo Lai and Qiang Tang. Making any attribute-based encryption accountable, efficiently. In Javier López, Jianying Zhou, and Miguel Soriano, editors, *ESORICS 2018, Part II*, volume 11099 of *LNCS*, pages 527–547. Springer, Heidelberg, September 2018. 5, 15, 16

[LW10] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 455–479. Springer, Heidelberg, February 2010. 27

[LW11] Allison B. Lewko and Brent Waters. Decentralizing attribute-based encryption. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 568–588. Springer, Heidelberg, May 2011. 3, 11

[LW15] Zhen Liu and Duncan S. Wong. Practical ciphertext-policy attribute-based encryption: Traitor tracing, revocation, and large universe. In Tal Malkin, Vladimir Kolesnikov, Allison Bishop Lewko, and Michalis Polychronakis, editors, *ACNS 15*, volume 9092 of *LNCS*, pages 127–146. Springer, Heidelberg, June 2015. 6, 15, 17

[LWW04] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups (extended abstract). In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *ACISP 04*, volume 3108 of *LNCS*, pages 325–335. Springer, Heidelberg, July 2004. 15

[LYZ19] Jiguo Li, Qihong Yu, and Yichen Zhang. Hierarchical attribute based encryption with continuous leakage-resilience. *Inf. Sci.*, 484(C):113–134, may 2019. 14

[MPR11] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 376–392. Springer, Heidelberg, February 2011. 2, 10, 15, 25

[NNL01] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 41–62. Springer, Heidelberg, August 2001. 14

[OT08] Tatsuaki Okamoto and Katsuyuki Takashima. Homomorphic encryption and signatures from vector decomposition. In Steven D. Galbraith and Kenneth G. Paterson, editors, *PAIRING 2008*, volume 5209 of *LNCS*, pages 57–74. Springer, Heidelberg, September 2008. 27

[OT09] Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 214–231. Springer, Heidelberg, December 2009. 2, 10, 13

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 191–208. Springer, Heidelberg, August 2010. 14, 27, 33

[OT11] Tatsuaki Okamoto and Katsuyuki Takashima. Efficient attribute-based signatures for non-monotone predicates in the standard model. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 35–52. Springer, Heidelberg, March 2011. 5, 7, 8, 13, 15, 16, 18, 37, 70

[OT12a] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 591–608. Springer, Heidelberg, April 2012. 5, 6, 13, 14, 16, 17

[OT12b] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure unbounded inner-product and attribute-based encryption. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 349–366. Springer, Heidelberg, December 2012. 14, 27, 32, 33, 37, 39, 41, 69, 77

[OT13] Tatsuaki Okamoto and Katsuyuki Takashima. Decentralized attribute-based signatures. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 125–142. Springer, Heidelberg, February / March 2013. 51, 69

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 552–565. Springer, Heidelberg, December 2001. 2, 10, 15

[SAH16] Yusuke Sakai, Nuttapong Attrapadung, and Goichiro Hanaoka. Attribute-based signatures for circuits from bilinear map. In Chen-Mou Cheng, Kai-Min Chung,

Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 283–300. Springer, Heidelberg, March 2016. 15

[Sch91] Claus-Peter Schnorr. Factoring integers and computing discrete logarithms via Diophantine approximations. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 281–293. Springer, Heidelberg, April 1991. 118

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 47–53. Springer, Heidelberg, August 1984. 2, 10

[SKAH18] Yusuke Sakai, Shuichi Katsumata, Nuttapong Attrapadung, and Goichiro Hanaoka. Attribute-based signatures for unbounded languages from standard assumptions. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 493–522. Springer, Heidelberg, December 2018. 15

[SW08] Elaine Shi and Brent Waters. Delegating capabilities in predicate encryption systems. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP 2008, Part II*, volume 5126 of *LNCS*, pages 560–578. Springer, Heidelberg, July 2008. 14

[Tar03] Gábor Tardos. Optimal probabilistic fingerprint codes. In *35th ACM STOC*, pages 116–125. ACM Press, June 2003. 109

[Wat05] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, Heidelberg, May 2005. 37

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, Heidelberg, August 2009. 2, 6, 10, 13, 17, 24, 27, 37, 71

[WLW10] Guojun Wang, Qin Liu, and Jie Wu. Hierarchical attribute-based encryption for fine-grained access control in cloud storage services (poster presentation). In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 2010*, pages 735–737. ACM Press, October 2010. 14

[Zha21] Mark Zhandry. White box traitor tracing. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 303–333, Virtual Event, August 2021. Springer, Heidelberg. 13

RÉSUMÉ

Le nombre d'appareils connectés par personne a massivement augmenté lors de la dernière décennie, en particulier dans les pays occidentaux. Étant donné que la sécurité des structures connectées au réseau peut être compromise à partir de n'importe quel point d'entrée par un attaquant malveillant, la sécurisation des trousseaux de clefs cryptographiques des utilisateurs devient centrale pour construire une infrastructure sécurisée. Afin d'explorer les réponses à cette problématique, nous proposons des solutions pour que les utilisateurs puissent gérer des appareils multiples. En particulier, nous nous penchons sur les pratiques cryptographiques qui sont compatibles avec les méthodologies modernes de contrôle d'accès basées sur des attributs. Ces pratiques doivent aussi intégrer les outils qui sont au cœur de la gestion d'appareils multiples par des utilisateurs. Le premier de ces outils est la délégation, qui permet aux utilisateurs de délimiter les capacités de déchiffrement de chacun de leurs appareils en fonction de leurs besoins. La délégation doit être possible sans nécessiter d'interaction avec une quelconque autorité, afin de préserver la vie privée et l'autonomie de l'utilisateur. Le second outil est le traçage, où nous exigeons que les appareils des utilisateurs puissent être identifiés en cas d'abus ou d'utilisation illégitime, afin que tout utilisateur puisse être tenu responsable de la gestion de ses appareils.

Nous commençons par présenter une approche pratique des Dual Pairing Vector Spaces (DPVS), qui est un système de construction et de preuves qui permet d'assurer le meilleur niveau de sécurité pour la cryptographie basée sur les attributs. Le DPVS est compatible avec des caractéristiques importantes de cette cryptographie telles que la richesse de l'expression des politiques de contrôle d'accès. Ensuite, nous présentons une nouvelle contribution pour le chiffrement basé sur les attributs sous la forme d'une nouvelle primitive : le Switchable-Attribute Key-Policy Attribute-Based Encryption (SA-KP-ABE). Dans un SA-KP-ABE, les attributs des utilisateurs et des chiffrés peuvent être "activés/désactivés" d'une manière indistinguable pour les utilisateurs. Nous prouvons que cette approche permet le traçage et qu'elle est compatible avec la délégation. Nous fournissons également une construction de SA-KP-ABE avec le DPVS. Notre dernière contribution est un schéma de signature basée sur des attributs qui permet deux méthodes de délégation. La première est la délégation habituelle de clefs, et la seconde permet de déléguer des politiques d'accès pré-approuvées qui peuvent être réutilisées pour signer différents messages. L'une ou l'autre de ces deux méthodes peut être utilisée en fonction du risque liée à une mauvaise gestion ou de la compromission de l'appareil recevant les clefs déléguées, car la seconde méthode implique moins de dommages potentiels que la première. De plus, nous prouvons également que notre schéma est compatible avec le traçage de signatures, où une autorité désignée peut lever l'anonymat des signatures suspectes.

MOTS CLÉS

Chiffrement basé sur les attributs ⋆ Signature basée sur les attributs ⋆ Délégation ⋆ Traçage

ABSTRACT

The last decade has seen a massive increase in the number of connected devices per person, especially in western countries. As the security of connected structures can be compromised from any entry point by a malicious attacker, securing sets of cryptographic keys of users becomes an important keystone to build secure infrastructure. To explore answers to this problem, we propose solutions oriented towards the management of multiple devices for each user. In particular, we consider cryptographic practices that are compatible with modern access-control methodologies based on attributes. These practices should be compatible with features that are central to management of multiple devices. The first is delegation, as a means for users to delimit the decryption capabilities of each of their devices depending on their needs. Delegation should be doable without requiring interaction with any sort of authority, in order to preserve the user's intimacy and autonomy. The second is tracing, in the sense that we require that devices can be tracked in the case of abuse or illegitimate use so that any user can be held accountable.

We begin by presenting a practical approach to the Dual Pairing Vector Spaces (DPVS), a framework that allows for full security of attribute-based schemes, while being compatible with important features like expressive policies. Then, we present a new contribution for attribute-based encryption schemes in the form of a new primitive: Switchable-Attribute Key-Policy Attribute-Based Encryption (SA-KP-ABE). In a SA-KP-ABE, the attributes of users and ciphertexts can be "turned on/off" in a way that is indistinguishable for the users. We prove that this approach allows for tracing, and is fully compatible with delegation. We also provide a construction of SA-KP-ABE in the DPVS framework. Our last contribution is an attribute-based signature scheme that allows for two types of delegation. The first one is the usual delegation of keys, and the second one allows to delegate pre-approved policies that can be re-used to sign different messages. Either of these two approaches can be used depending on the risk of mismanagement or corruption of the device receiving the delegated keys, as the latter incurs less possible damage than the first one. Furthermore, we also prove our scheme to be compatible with tracing techniques, where a designated authority can lift the anonymity of suspicious signatures.

KEYWORDS

Attribute-Based Encryption ⋆ Attribute-Based Signature ⋆ Delegation ⋆ Traitor-Tracing