



HAL
open science

Sécurité de la base de données cadastrales

Firas El Khalil

► **To cite this version:**

Firas El Khalil. Sécurité de la base de données cadastrales. Informatique. Université de la Polynésie Française, 2015. Français. NNT : 2015POLF0001 . tel-01482905

HAL Id: tel-01482905

<https://theses.hal.science/tel-01482905>

Submitted on 3 Mar 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ DE LA POLYNÉSIE FRANÇAISE

ÉCOLE DOCTORALE DU PACIFIQUE ED 469

Laboratoire GePaSud

THÈSE

présentée et soutenue publiquement par

FIRAS EL KHALIL

le 28 Janvier 2015

en vue de l'obtention du titre de

Docteur de l'Université de la Polynésie française

Département : Sciences

Discipline : Informatique

Spécialité : Sécurité

Sécurité de la Base de Données Cadastreale

Sous la direction du Professeur Alban Gabillon

et du Docteur Patrick Capolsini

Jury

Vijay Varadharajan	Professeur à Macquarie University, Australie	Rapporteur
Frédéric Cuppens	Professeur à Télécom Bretagne, France	Rapporteur
Alban Gabillon	Professeur à l'Université de la Polynésie française	Directeur de Thèse
Patrick Capolsini	Maître de Conférences à l'Université de la Polynésie française	Co-directeur de Thèse
Hakima Chaouchi	Professeur à Télécom SudParis, France	Examinatrice



Ce travail est mis à disposition selon les termes de la licence *Creative Commons Attribution – Pas d'Utilisation Commerciale – Partage dans les Mêmes Conditions 4.0 Internationale* :
<http://creativecommons.org/licenses/by-nc-sa/4.0/>.

This work is licensed under a *Creative Commons Attribution – NonCommercial – ShareAlike 4.0 International License*: <http://creativecommons.org/licenses/by-nc-sa/4.0/>.

«*Science is the belief in the ignorance of experts*»

—Richard Feynman, *What is Science*, 1966.

to my parents

Acknowledgments

First of all I would like to express my gratitude to Prof. Frédéric Cuppens and Prof. Vijay Varadharajan for sparing some of their precious time for the task of reviewing and commenting on this manuscript.

I would like to thank Prof. Hakima Chaouchi for accepting to be part of the jury and examining this thesis.

This work would have never been accomplished without the careful guidance of my advisor, Prof. Alban Gabillon. I am grateful for his insightful discussions, patience, permanent attention to details, and his scientific spirit that has deeply affected me personally and professionally. His unadulterated sense of humor made working with him a real joy. Thank you for supporting me and believing in my work especially when I was seriously—and unadmittedly—doubting it.

I am also grateful to my co-advisor Dr. Patrick Capolsini. He always managed to find time in his notoriously busy schedule to review and discuss my work, making sure it is consistent and coherent. Thank you for all the academic, logistic, and administrative help that you provided during this journey.

I am thankful to all my colleagues at GePaSud for welcoming me at the laboratory. I am especially thankful to Prof. Jean-Pierre Barriot, the director of the laboratory, for all the help he provided and for all the charming discussions we had around numerous cups of coffee on science, physics, and space.

I am also thankful for the tremendous help of Emmanuel Bouniot, the head of the GIS cell at the computer science service of French Polynesia, and Bertrand Malet, the director of the cadastral department of the real-estate service of French Polynesia, who did their best to make sure that I really understand their problematics. Thank you for providing me with all necessary resources that had deepened my understanding of the subject. I would also like to thank Philippe Siquin and Tehei Taiore from the GIS cell for the technical aid they provided.

I want to thank the members of GAATI, the mathematics laboratory of the University of French Polynesia, for their help especially Dr. Gaëtan Bisson for all the fruitful discussions.

I am forever grateful to Dr. Bechara Al Bouna and Prof. Richard Chbeir whose passion for research made me rethink my whole career. I would have never gotten into scientific research without their support and encouragement.

I am also grateful to all my friends that I have met in Tahiti who made my journey a real delight, particularly Aymeric Hermann, Jonathan Serafini, and Martin Loesdau with whom I shared a lot of great moments and long ethanol-induced “*philosophical debates*”—if one may call them so. Of course I will never forget my wonderful self-appointed parents, Nicolas Loiseau and Jennifer Gless, whose presence alone makes me a happy camper.

Clara de Gaillande, Gaëlle Legras, Mathilde Menoud, and Simon Van Wynsberge, thank you for tolerating my not-so-occasional OCD-ish walks in every single room of the house during the writing of this manuscript. I cannot describe how happy I am that I left the horrible grotto I used to live in. You are family to me.

My dear friends in Lebanon who supported me since day one and helped, entertained, and cheered me during all the phases of development of this thesis from the opposite side of the planet: François Zard, Rami Arnaout, and Salem Fayad, thank you.

Elissa and Tarek, you are more than friends or a sister and a brother. Thank you for everything.

Last but not least, I am eternally in debt to my parents who sacrificed so much in their lives and tolerated the difficulties of living in troublesome environments and rigid conditions just to ensure that we—Elissa, Tarek, and I—get the best education they could possibly provide. Thank you for supporting me emotionally and financially to finish this thesis.

Résumé

Le contrôle d'agrégation dit *Quantity Based Aggregation* (QBA) est lié au contrôle de l'inférence dans les bases de données et a rarement été traité par la communauté scientifique.

Considérons un ensemble S formé de N éléments. L'agrégation d'éléments de l'ensemble S n'est considérée comme étant confidentielle qu'à partir d'un certain seuil k , avec $k < N$. Le but du contrôle QBA est donc de garantir que le nombre d'éléments de S délivrés à un utilisateur reste toujours inférieur à k .

Dans cette thèse, nous traitons du problème du contrôle QBA dans les bases de données cadastrales. Ce travail répond à un besoin du service des affaires foncières de la Polynésie française. La politique de sécurité qu'il nous a été demandé d'implanter donne le droit à chaque utilisateur de connaître le propriétaire de n'importe quelle parcelle. Cette permission est toutefois limitée par les interdictions suivantes: l'utilisateur ne doit jamais arriver à connaître

1. tous les propriétaires dans une région donnée, et
2. toutes les parcelles appartenant au même propriétaire.

Chacune de ces interdictions correspond, de manière évidente, à un problème de type QBA. Dans ce manuscrit, nous développons d'abord un modèle pour assurer la première interdiction, ensuite nous montrons comment adapter notre modèle à la seconde interdiction. Nous présentons une implémentation de notre modèle pour les bases de données relationnelles.

Notre modèle traite de plusieurs points particuliers:

- Nous tenons compte de la possibilité de collusion (lorsque plusieurs utilisateurs coopèrent afin de contourner la politique de sécurité).
- Nous discutons de la notion de région et proposons deux types de régions élémentaires: la "zone" et la "zone dominante."

- Nous discutons du traitement des mises-à-jour spécifiques à l'application cadastrale et de leurs implications sur le contrôle QBA: opérations d'achat, de vente, de fusion et de division.
- Nous discutons de la meilleure stratégie à adopter pour le traitement de l'historique d'accès afin de garantir que les utilisateurs honnêtes ne restent pas bloqués sur un ensemble de parcelles indéfiniment, augmentant ainsi la disponibilité et l'utilité de la base de données.

Dans la dernière partie de ce manuscrit, nous présentons un prototype d'application de base de données cadastrale assurant le contrôle QBA, développé sur la base de notre modèle. Nous détaillons les résultats des tests montrant les avantages de la "*zone dominante*" sur la "*zone*." Nous montrons en particulier que l'adoption de la "*zone dominante*" en tant que région élémentaire améliore la disponibilité des données et les performances du contrôle QBA.

Abstract

Quantity Based Aggregation (QBA) control is closely related to inference control in databases and has been rarely addressed by the scientific community.

Let us consider a set S of N elements. The aggregation of k elements, at most, out of N is not considered sensitive, while the aggregation of more than k out of N elements is considered sensitive and should be prevented. The role of QBA control is to make sure that the number of disclosed elements of S is less than or equal to k , where $k < N$.

In this thesis we work on QBA problems in the context of cadastral databases. This work addresses an actual need of the real-estate service of French Polynesia. The security policy, that we were asked to implement, gives every user the right to know the owner of any parcel in the database. This permission is, however, constrained with the following prohibitions: the user cannot acquire the knowledge of

1. the owners of all parcels in a given region, and
2. all parcels belonging to the same owner.

Each prohibition represents, obviously, a distinct QBA problem. In this manuscript, we develop a model to enforce the first prohibition, then we show how this work can be adapted to the enforcement of the second prohibition. Afterwards, we present an implementation for relational databases.

Our model addresses several aspects:

- We take collusions into account (when multiple users collaborate to circumvent the security policy).
- We discuss the notion of a region and we propose two basic definitions: the “zone” and the “dominant zone.”

- We discuss database updates specific to the cadastral application, and its implications on QBA control. We propose a scheme to handle buy, sell, merge and split operations.
- We propose a strategy to handle access history in order to guarantee that honest users do not get blocked indefinitely on a set of accessible parcels, thus increasing the availability and the utility of the database.

At the end of this manuscript we present the prototype we developed to showcase QBA control, in addition to benchmarks showing the advantage of one definition of a region (the “*dominant zone*”) over another (the “*zone*”). We show that the adoption of the “*dominant zone*” increases data availability and improves the performance of the QBA control enforcement algorithm.

Contents

Acknowledgments	vii
Résumé	ix
Abstract	xi
List of Algorithms	xv
List of Figures	xvii
List of Listings	xx
List of Tables	xxi
1 Introduction	1
1.1 Contributions	10
1.2 Organization of the Manuscript	11
2 State of the Art	13
2.1 Online Cadasters	14
2.2 Inference Control	17
2.3 Aggregation Control	22
2.3.1 Literature Review	22
2.3.2 The Work of Motro, Marks and Jajodia	26
2.3.2.1 The Model	27
2.3.2.2 Discussion	30

3	The Model	31
3.1	Definitions	33
3.2	Security Policy	35
3.3	Enforcing Pr_1	37
3.4	Enforcing Pr_2	43
4	Implementing the Security Policy	45
4.1	Implementing Pr_1	47
4.2	Implementing Pr_2	52
5	Additional Aspects	57
5.1	Handling Cadastral Updates	59
5.1.1	Mutations in Pr_1	59
5.1.2	Mutations in Pr_2	62
5.2	Resetting Access	62
5.3	Inference Channels	65
5.3.1	Potential Inference Channels from external knowledge	65
5.3.2	Potential Inference Channels from Denial of Access in Pr_1	66
5.3.3	Potential Inference Channels from Denial of Access in Pr_2	66
5.4	Authentication	68
5.5	Choosing the Model's Parameters	70
6	Application to the Cadaster of French Polynesia	75
6.1	Desired Workflow	76
6.2	Authentication	78
6.3	Pr_1 , Pr_2 or Both?	79
6.4	Server-side Security	80
7	The Prototype	81
7.1	User Interface	83
7.2	Graph Generation	86

Contents	xv
7.2.1 A Primer on R-tree	86
7.2.2 Generation Method	89
7.3 The Database	91
7.4 The Server	92
7.5 The Client	94
8 Benchmarks	97
8.1 Availability Benchmarks	99
8.1.1 Availability without Dominant Zones	99
8.1.2 Availability with Dominant Zones	102
8.2 Performance Benchmarks	106
8.2.1 Performance of QBA Control with Zones	106
8.2.2 Comparing Zones and Dominant Zones	110
9 Conclusion	115
9.1 Future Work	118
Appendix A Example of a Cadastral Excerpt	119
Appendix B The Model of MMJ applied to the Cadastre	121
Appendix C Enforcement Scripts in PL/SQL	125
Bibliography	131

List of Algorithms

1	QBA Control Enforcement Algorithm in the Model of Motro, Marks and Jajodia	29
2	QBA Control Enforcement Algorithm for Pr_1	51
3	QBA Control Enforcement Algorithm for Pr_2	56
4	Update Algorithm for Merge/Split (Pr_1)	61
5	Modifications for Algorithm 2 to Compute Dominant Zones on-the-fly	110
6	Modified QBA Control Enforcement Algorithm in the Model of Motro, Marks and Jajodia for the Cadastral Application	123

List of Figures

3.1 A graph for Pr_1	38
3.2 A graph for Pr_2	43
4.1 Entity-Relationship Diagram of the Database	46
5.1 Cumulative distribution of parcels attached to zones of different sizes	72
7.1 Prototype Overview	82
7.2 Initial Screen	83
7.3 Simulation of 2 Users	84
7.4 Part of the Cadastral Database of Maupiti	84
7.5 Part of the Cadastral Database of Maupiti with the GeoJSON Graph Plotted on Top	85
7.6 Part of the Cadastral Database of Maupiti Showing Information When Hovering over a Parcel	85
7.7 Example of R-tree for 2D Rectangles: Rectangle Visualization [Com10]	87
7.8 Example of R-tree for 2D Rectangles: Data Structure Visualization [Com10]	87
7.9 index.html	94
7.10 Access Granted	95
7.11 Access Denied: the Limit KH is Reached	95
7.12 Access Denied: the Limit y is Reached and Borders of Dominant Zones that Risk Surpassing y are Highlighted	96

8.1	Availability for Different Values of y and z for Zones	99
8.2	Availability for Different Values of y and z for 2-zones	100
8.3	Availability for Different Values of y and z for Dominant Zones . .	100
8.4	Comparison of zones, 2-zones and dominant zones	103
8.5	Example Graph Representing Parcels	104
8.6	Performance Figures for x -collusion resistance	107
8.7	Performance Figures for (x, y, z) -collusion resistance	109
8.8	Execution Time of (x, y, z) -collusion resistance for 2 Values of y (3 and 4 for Figures 8.8a and 8.8b respectively). OTF and DB Designate Algorithms 5 and 2 Respectively, where “Dominant Zones” are Computed <i>On-The-Fly</i> and Retrieved from the <i>Data Base</i> , Respectively	112
8.9	Comparison of the Number of Parcels, Resulting from Breadth-First Traversal, as a Function of z	113

List of Listings

4.1	Database Tables Corresponding to the ERD of Figure 4.1	46
4.2	Views for Pr_1	48
4.3	Actual Database Relations for Pr_1	49
4.4	Views for Pr_2	53
4.5	Actual Database Relations for Pr_2	54
7.1	BFS with SQL CTE (Common Table Expressions)	91
C.1	Enforcement Algorithm in PL/SQL	125
C.2	Building Breadth-First Traversal SQL Request	129
C.3	Breadth-First Traversal Implementation in SQL (1)	130
C.4	Breadth-First Traversal Implementation in SQL (2)	130

List of Tables

2.1 The Phonebook Example	27
B.1 Tuples of the CADASTER Relation	122

1

Introduction

This thesis addresses a specific problem known in the literature as Quantity Based Aggregation (QBA) control, with a particular application to cadastral databases. In fact, in the context of collaboration with the computer science department of French Polynesia, we managed to identify their need to develop QBA control for one of the databases they host and manage: the cadastral database of French Polynesia, which is operated by its real-estate service.

A cadastral database is used to manage parcels of a country. A parcel is a piece of land with established boundaries and owned by a legal entity. A legal entity is a legal construct designating a single natural person or a group of natural persons (e.g. a married couple, a company, a political party, the state, etc.) for legal purposes like lawsuits but especially for property ownership. Parcels are properties that can be owned by one or multiple legal entities. The main role of a cadastral database is to hold the current state—and transactional history—of the real-estate of the country: creation of new parcels in unsurveyed lands, merging multiple neighboring parcels to form a new one, splitting an existing parcel into new ones, and changing ownership after a buy/sell operation. All cadastral databases, regardless of the country that operates them, share these common characteristics.

Technically speaking, cadastral databases are stored in spatial databases which are used for a multitude of spatial operations: they can be used to create, query, and modify geo-referenced objects (e.g. points, lines, polygons, etc.) and perform spatial operations (for instance, finding the intersection of two lines or computing the list of neighboring objects of a polygon). They are used to provide different types of services, such as location based services (e.g. Foursquare ¹), maps (e.g. Google Maps ²), games (e.g. Ingress ³), etc. A Geographic Information System (GIS), broadly speaking, is a collection of tools and technologies used to deliver a service that manipulates geographic data. For our purposes, we will use the term GIS to designate the complete toolchain that makes a geographical application: from spatial databases, to the client application, and any actively involved intermediate servers.

The cadastral database of French Polynesia is accessible—using a custom application—by a handful of people: employees of the real-estate service, civil-law notaries, and cadastral surveyors ⁴. These users can query the database from the internal network of the real-estate service, or through the Internet via VPN (Virtual Private Network). That was the case at the early stages of development of this thesis. Currently, there is a GIS application connected to the database, and exposed to the Internet, granting access to the aforementioned users using basic authentication ⁵.

Access to the geometry of parcels is granted to the general public, but ownership information is strictly prohibited. The real-estate service wishes to make it accessible, but access to this information is limited by legal texts for which they found a legal interpretation but not the technological capacity to implement it. Indeed, French Polynesia is a French overseas territory, which means that it has a certain autonomy when it comes to some administrative and financial aspects of

¹<https://foursquare.com/>

²<https://www.google.com/maps>

³<https://www.ingress.com/>

⁴Géomètre-expert in French.

⁵User name and password over HTTPS.

governance. French law, regarding online publishing of cadastral data, applies to French Polynesia, but the choices made at the level of the territory are independent of the choices of the French government.

The CNIL ⁶ prohibits any party (a French public organism, private company, etc.) from publishing personally identifying information (e.g. names, social-security numbers, etc.) on the Internet. However, the Cada ⁷ says that access to administrative documents, especially excerpts of parcels ⁸, should be public. Any person has the right to present himself to the designated authority and ask for one or multiple excerpts.

In accordance with French law, especially the recommendations of the Cada, the real-estate service of French Polynesia opens its doors to citizens. They can present themselves physically, wait in a queue, and contact an employee of the service. The citizen is only required to give the identifier of the requested parcel, or its address, and usually the employee would use the cadastral application to print necessary excerpts. S/he is not required to present any ID (e.g. drivers license, credit card, etc.), however some fees may apply.

The role of the employee is paramount. S/he is asked to perform a check on submitted demands making sure that a given person is not asking for too much excerpts, and denying access to certain parcels that are evaluated as sensitive (e.g. one or more parcels owned by the president). Basically, one of the pivotal roles attributed to the employee is to perform access control.

The real-estate service wishes to capture the actual process, and reproduce it (as much as possible) in an interactive application through the Internet. In fact, they want the users to access a “*point-and-click*” mapping interface, and be able

⁶*Commission Nationale de l'Informatique et des Libertés*. An independent administrative authority whose mission is to ensure that information technology is at the service of citizens and does not undermine human identity, rights, private life, or individual and public liberties.

⁷*Commission d'accès aux documents administratifs*. An independent administrative authority responsible for ensuring freedom of access to administrative documents.

⁸Official documents mainly stating the ownership of a parcel, among other geographic and non-geographic information. See Appendix A.

to buy cadastral excerpts. Once selected, and before the actual payment happens, the user is presented with the ability to “*preview*” ownership information of the parcel. This can be handy for situations where online data are not synchronized to the latest available information. Therefore, this “*preview*” acts as a safeguard for users, and allows them to pay for correct and complete excerpts. Obviously, this workflow is susceptible to abuse by dishonest users, which will exploit this feature to reconstruct the complete database.

A proper access control mechanism ensuring that the act of publishing the cadastral database online would not clash with the recommendations of the CNIL is needed. This led the real-estate service to the specification of a security policy that captures the spirit of the access control done by the employee. Any user has the right to know the owner’s name of any given parcel, however this permission is constrained with the following prohibitions: the user is forbidden to know

Pr₁: The list of owners of all parcels in a region,

Pr₂: The list of all parcels belonging to the same legal entity (person, family, etc.).

These prohibitions are defined to limit the risk of violating people’s privacy. Indeed the violation of any of these prohibitions entails a violation of the privacy of people assisting in the database. The violation of the first prohibition may expose people to unwanted targeted commercial advertising or illegal contact by big agencies who seek to buy big regions for their own purposes (e.g. construction of a resort, a mall, etc.) Agencies who are in the business of real-estate have an advantage over people in terms of market prediction and price changes, and they may use their power to take advantage of uneducated or uninformed land owners. On the other hand, The list of goods belonging to any person, like the list of accounts a person has or the list of cars s/he owns, should not be publicly accessible to protect people’s privacy as per the recommendations of the CNIL. Parcels are goods that should be protected the same way, and this is why the list of parcels belonging to the same owner should not be published in their entirety.

Our research showed that these prohibitions intend to enforce a type of aggregation control, namely QBA. The aggregation problem is close to the inference problem and usually discussed with it. QBA problems were distinguished from inference and other aggregation problems for the first time in the work of Hinke [Hin88], under the name “*cardinality aggregation*.” Lunt [Lun89] analyzed inference and aggregation problems and showed the difference between them⁹. She coined the term quantity-based aggregation, and she gave the following example to illustrate QBA: suppose that there is a phonebook of N phone entries; a user has the right to know k entries, at most, out of N . The goal of QBA control is to enforce this “*k out of N*” disclosure control. One can clearly see the similarity between both Pr_1 and Pr_2 on one hand, and the phonebook example on the other: the list of parcels in a region and the list of parcels belonging to the same owner are analogous to a phonebook, where the association between a parcel and an owner is analogous to an entry; a user has the right to access any proper subset¹⁰ of these lists, but not the entire list.

Interestingly enough, Hinke [Hin88] noted that «*[the inference problem] appear to be more tractable. With cardinality aggregation, it is not always clear why “N” elements of a set, such as a phonebook, are classified at one level, while “M” elements are less classified, where cardinality of $N \geq$ cardinality of M .*» Indeed, the phonebook example does not induce any special interest, which was apparent while surveying the literature. As a matter of fact, the only work that addresses QBA seriously is that of Motro, Marks and Jajodia [MMJ94; MMJ96], around twenty years ago. However, the aforementioned work starts with a different hypothesis: they treated QBA in situations where users can issue “*arbitrary queries*”, e.g. when a user can query multiple entries of the same phonebook at the same time. The cadastral application is a “*point-and-click*” style application, i.e. a user can query one

⁹Section 3.1 presents definitions and examples of inference, aggregation, and QBA, highlighting the difference between them.

¹⁰A proper subset S' of a set S , denoted $S' \subset S$, is a subset that is strictly contained in S and so necessarily excludes at least one member of S . The empty set is therefore a proper subset of any nonempty set [Kam50].

entry of the phonebook at a time. This discrepancy is one of multiple reasons that renders the adaptation of the work of Motro, Marks and Jajodia to the cadastral requirements tedious and impractical.

Nevertheless, we tackled both prohibitions, and presented our model and implementation in [AGC13]. The main challenge was in Pr_1 where we needed to properly define a “*region*.” We needed a definition that captures people’s perception of a region; one that could even adapt to the differences in the perception of a region from one person to another. Indeed, a region could be interesting for one person because of its economical value (e.g. a beach strip or an industrial sector), or because of its emotional value (e.g. a nice hill close to the properties of a family), etc. Therefore, any static definition of a region (e.g. considering that a municipality is equivalent to a region) was eliminated.

Our model (for both Pr_1 and Pr_2) is based on graphs, where a parcel is a node, and two neighboring parcels are connected with an edge. For Pr_1 , two parcels are considered as neighbors in the graph if they touch or if they are separated by a road, river, etc. For Pr_2 , two parcels are considered as neighbors in the graph if they belong to the same legal entity. Then we define a zone, the most basic region that could be modeled: a zone of a parcel is made by itself and all its 1st degree neighbors in the graph. Therefore, the zone of a parcel in Pr_1 is formed by the parcel itself and all its direct geographical neighbors (touching or separated by a given distance), while a zone in Pr_2 is formed by the parcel itself and all parcels that are owned by the same legal entity. A user has the right to access ownership information of any parcel in a zone, but s/he is not allowed to aggregate the knowledge of owners of all parcels in that zone. Satisfying this condition satisfies the security policy.

Another topic we address is collusion: when multiple dishonest users collaborate to bypass QBA control. We introduce x -collusion resistance as means to prevent x users from colluding on a single zone. However, x -collusion resistance considers that all users are potential colluders. In order to minimize the probability

of detecting false colluders, we introduce (x, y, z) -collusion resistance that uses the notion of z -region: the generalization of a zone. A z -region of a parcel is formed by the zones of its neighboring parcels of degree less than or equal to z . Therefore, a z -region of a parcel in Pr_1 is formed by the parcel itself and all its geographical neighbors of degree less than or equal to z . A z -region of a parcel in Pr_2 is formed by the parcel itself, parcels belonging to the same legal entity (its zone), and all parcels belonging to owners who are at a social distance less than or equal to z from the legal entity (which requires a social graph of participating owners). In (x, y, z) -collusion resistance x users are not considered as colluders until they hit the threshold of y collusions. Once hit, the QBA control mechanism should enforce x -collusion resistance.

There is also an important aspect that should be considered while choosing parameters for QBA control, most importantly the values of x , y and z for (x, y, z) -collusion resistance. In fact, these three parameters are assigned by the security administrator, but they should not be selected arbitrarily. As a general rule of thumb, x and z contribute to the restrictiveness of QBA control, while y contributes to its permissiveness. In other words, increasing x or z would make data less available to end-users, while increasing y makes it more available. Analysis of the distribution of zones is an imperative step that should be done before assigning values to these parameters, because every cadastral database is different; the topology of a modern city is different than the topology of an ancient one. This difference in topologies is reflected in the graph we use, thus in the model, and consequently in overall data availability.

We also found out that QBA control enforcement itself can create inference channels caused by a denial of access and background knowledge. For Pr_1 , if a user is denied access, s/he does not learn much; s/he can simply verify that s/he has queried too many parcels in a region. For Pr_2 , a user can learn from a denial of access that s/he has queried the last parcel, or one of the remaining parcels

belonging to the target owner. To address this issue, we propose to deny access to the last parcel and all of its geographical neighbors, to decrease the attacker's confidence.

It is obvious by now that, after a given period of time, a user who has maximally queried a zone will be blocked. But the question is: *should he be blocked indefinitely?* The answer from the real-estate service came clear: *no, access should be renewed regularly.* We then developed a gradual resetting scheme to address this specific issue.

We also addressed another dynamic aspect of the cadastral database: mutations, i.e. buy/sell and merge/split operations. *What should happen to user access history when mutation occurs? Should new parcels inherit access history from old ones? How can we give equal rights of access to all users and avoid, as much as possible, security issues that may arise?* We found out that for Pr_1 , access history for a buy/sell operation should be kept as is (which is counter-intuitive). However, merge/split for Pr_1 , and all mutations for Pr_2 should be accompanied with a complete erasure of user access history, for security and performance reasons.

After further investigation, we were not satisfied with the initial implementation, that used graph databases, in terms of storage and processing time. That led to the development of an alternative implementation in the relational model, which not only improved performance, but also allowed us to provide a solution that is the most compatible with the computer science service's infrastructure. We presented the details of the implementation and the prototype in [AGC14a]. We also benchmarked our QBA control algorithm and showed, empirically, that its performance is linear with respect to the number of users of the database.

One problem that kept persisting was the amount of cadastral data that was available, once the parameters x , y , and z of our model were set. *Can we keep the same level of confidentiality while increasing data availability?* We needed a solution that addresses the model itself, regardless of chosen parameters or the implemen-

tation. In [AGC14b] we introduced dominant zones: a dominant zone of a parcel is the zone containing it that has the biggest cardinality. QBA control now considers dominant zones only. So instead of giving importance to all zones containing a parcel, we only give importance to a subset of zones—those with the highest cardinality. This lead to a decrease in the number of “*active zones*” we consider during QBA enforcement, thus less “*active zones*” contribute to the disclosure decision, which was reflected in more availability. This minimal change in the model proved to be not only beneficial in terms of the increase of data availability, but also in performance when compared to zones.

Currently, at the time of writing of this manuscript, we are negotiating with a third-party designated by the computer-science service—and world-renowned for its GIS solutions—the details of the implementation of our model and algorithms. The goal is to produce the desired application for the real-estate service with their desired functional requirements and workflows.

1.1 Contributions

This manuscript presents our latest results and synthesizes the work done to provide a complete overview of QBA control in cadastral databases. The work in its entirety is considered as an original contribution. To the best of our knowledge, we are the only ones who have worked on QBA problems in cadastral databases. The core of this thesis has been peer-reviewed and published in three international conferences.

[AGC13] Firas Al Khalil, Alban Gabillon, and Patrick Capolsini. “Collusion Resistant Inference Control for Cadastral Databases”. In: *Foundations and Practice of Security - 6th International Symposium, FPS 2013, La Rochelle, France, October 21-22, 2013, Revised Selected Papers*. 2013, pp. 189–208. doi: [10.1007/978-3-319-05302-8_12](https://doi.org/10.1007/978-3-319-05302-8_12) (see pp. 6, 97).

[AGC14a] Firas Al Khalil, Alban Gabillon, and Patrick Capolsini. “Implementing Quantity Based Aggregation Control for Cadastral Databases”. In: *2014 IEEE World Congress on Services, Anchorage, AK, USA, June 27 - July 2, 2014*. 2014, pp. 137–144. doi: [10.1109/SERVICES.2014.33](https://doi.org/10.1109/SERVICES.2014.33) (see pp. 8, 97).

[AGC14b] Firas Al Khalil, Alban Gabillon, and Patrick Capolsini. “Quantity Based Aggregation Control for Cadastral Databases”. In: *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Privacy in Geographic Information Collection and Analysis, GeoPrivacy '14, Dallas/Fort Worth, Texas, USA, November 4-7, 2014*. 2014, 7:1–7:8. doi: [10.1145/2675682.2676394](https://doi.org/10.1145/2675682.2676394) (see pp. 9, 98).

We have also completed an article synthesizing our work, based on this thesis, to be submitted to a peer-reviewed scientific journal.

1.2 Organization of the Manuscript

In the first chapter we have introduced the motivation behind this work, which addresses a need of the real-estate service of French Polynesia. We also presented rapidly the highlights of our research, in addition to contributions. The remainder of this manuscript is organized as follows:

Chapter 2 “*State of the Art*” – first surveys the current status of online cadastral databases in different countries. Afterwards, a review of the state of the art on inference and aggregation is presented. And finally the most relevant work done on QBA problems is presented in detail and discussed.

Chapter 3 “*The Model*” – starts with definitions that help discriminating between inference, aggregation and QBA problems. Afterwards, the security policy is defined, and details of the enforcement of Pr_1 and Pr_2 are presented.

Chapter 4 “*Implementing the Security Policy*” – shows how to implement the model in a relational database. All necessary database schemas and algorithms are listed.

Chapter 5 “*Additional Aspects*” – treats subjects that should be considered when implementing QBA control: cadastral updates, resetting access and additional inference channels that may arise from QBA control itself, and authentication. We then discuss the relationship between different parameters of the model and how to chose them properly.

Chapter 6 “*Application to the Cadaster of French Polynesia*” – addresses specificities of the French Polynesia Cadaster: from the desired workflow and its impact on authentication, to prohibitions themselves, to the choice of different parameters of the model, and server-side security.

Chapter 7 “*The Prototype*” – presents essential parts of the developed prototype. A global overview of the architecture and usage is given. Afterwards, individual components are detailed: graph generation methodology, the client, the database and the server.

Chapter 8 “*Benchmarks*” – shows how dominant zones are superior to zones in terms of both performance and at providing more data availability. It also shows that the performance of our QBA control algorithms is linear with respect to the number of users in the database.

Chapter 9 “*Conclusion*” – concludes the thesis and discusses future work.

2

State of the Art

In this chapter we will review the state of some cadastral databases, in Section 2.1, providing examples from different countries and asking the following questions: *are cadastral data accessible online? If so, what kind of information is available? To whom? And how?* As far as we know, the security policy addressed in this thesis was never encountered before in a cadastral database.

Afterwards, we will review the state of the art on inference and aggregation in Sections 2.2 and 2.3 respectively. In fact QBA problems are closely related to inferences and they are usually discussed together. Moreover, our work is close to the work of Staddon [Sta03] who treated inferences. We also review aggregation problems because of their similarity with QBA problems, especially the Chinese-Wall security policy [BN89].

Finally we will review in detail what seems to be the only work that addresses QBA directly [MMJ94] in Section 2.3.2, which also inspired our work.

In the following chapter, we will rely on literature review of inference, aggregation, and QBA to provide a set of definitions highlighting the differences between them, which is crucial for any work that targets this class of problems. The nuances are subtle, and one problem can be easily mistaken for another.

2.1 Online Cadasters

After investigating the state of online cadastral applications, we can give a couple of examples from different countries reflecting the legal point of view on the publication of parcel ownership information. We also explain the French point of view on the subject and the case of French Polynesia motivating this work.

Access to the Spanish [CV13] cadaster is provided through a mapping interface built with Google Maps. Parcel ownership information is considered sensitive and it is not available to the public¹. Land owners form a different level of users (more privileged than the public) and they are granted access to all information related to their own properties if they provide a valid X509 certificate associated with their national electronic ID.

Similarly, the Belgian cadaster is available online for the public² where ownership information is considered sensitive, thus prohibited. Using their national electronic ID, authenticated users can access through another website³ to information related to their own parcels only. We were not able to see how exactly (*a mapping interface? A simple list?*) this information is provided due to authentication requirements.

In Australia, access to the cadastral database is available⁴ through a Java applet over HTTPS. Land boundary information is considered public, but ownership information is confidential. In fact, a user can buy a “*Property Interest Report*” that gives sufficient information to help a potential buyer «*make an informed decision before [he buys].*» This includes a confirmation that a property is not affected by an interest, where to get further information, whether it is a contaminated site or heritage listed, among many others.

¹<http://www.maps.data-spain.com/cadastral/>

²<http://ccff02.minfin.fgov.be/cadgisweb/>

³<https://eservices.minfin.fgov.be/portal/fr/public/citizen/welcome>

⁴<https://www.landgate.wa.gov.au/bmvf/app/mapviewer/>

New Zealand provides its cadastral database as basic datasets ⁵ that can be bought. The database contains detailed information about parcels, including ownership information. Some owners are not present in the database because they are «*protected by a non-disclosure direction.*» No mapping application is provided.

In Croatia, parcel ownership information is public. Users can access the online website⁶ where they can submit a query on any parcel and get a list of information related to the parcel, including land ownership. Queries are submitted by selecting the desired department, office and parcel ID or deed ID (using simple rudimentary lists). Users are required to solve a CAPTCHA before query submission.

Similarly, the state of Montana, US, considers land ownership as public information and they provide the cadaster for online browsing through a mapping interface⁷. Access to cadastral data in the US depends on state-level legislation.

Canada publishes its cadaster freely ⁸. No ownership information is present, but all parcels can be downloaded as vector data (shapefiles) from their FTP site, after agreeing on a user-license agreement.

In France, the cadaster is available through a mapping interface⁹, however, only land boundaries are available to the public. This is due to the CNIL recommendation [09] where it is stated that «*the diffusion of any identifying information (directly or indirectly) on interactive terminals or public websites entails the risk of using this information for other purposes, including commercial, without the concerned people's consent.*»

The Cada [13] indicates that “*punctual demands*” of cadastral excerpts are allowed. Furthermore, cadastral excerpts may contain the name of land owners, but no other identifying information such as their national ID or their address. The frequency of demands and the number of parcels requested should be analyzed to

⁵<http://www.linz.govt.nz/survey-titles/landonline/landonline-data-services>

⁶<http://www.katastar.hr/>

⁷<http://svc.mt.gov/msl/mtcadastral/>

⁸<http://clss.nrcan.gc.ca/cadastraldata-donneescadastrales-eng.php>

⁹<http://www.geoportail.gouv.fr/>

ensure that these demands do not infringe the principle of free communication of cadastral documents. There is no clear definition of “*punctual demands*” and it is subject to various interpretations, therefore the Cada recommends a restrictive interpretation of the term.

French Polynesia is an overseas territory of France, where the recommendations of the CNIL and Cada are applicable. Currently, the punctuality of demands issued by citizens is ensured by employees of the real-estate service of French Polynesia when they are physically present at their desks. The work presented here is a requirement of the computer science service of French Polynesia expressing their interpretation of the recommendations of both CNIL and Cada in order to provide the same facilities offered by the real estate service through the internet: a user should have access to the ownership information of any parcel, at random, but s/he is not allowed to exploit the service for commercial ends (or social, ...). This interpretation is the foundation of prohibitions Pr_1 and Pr_2 presented in detail in Section [3.2](#).

2.2 Inference Control

The problem of inference and inference control has been heavily studied in the literature. Farkas and Jajodia [FJ02] present a review on inference in multiple domains: statistical databases, general and multi-level secure databases, data mining and web-based inferences. In a more recent survey, Woodall and Brereton [WB10] did a systematic literature review on inferences, categorizing different inference strategies. They identified 11 strategies from which we mention “*split query*”, “*distributed strategy*”, “*colluding users*”, “*external information*” and others. The reader is invited to read these papers for a detailed analysis on inference control. Nevertheless, we will highlight some of the notable work on the subject.

In the domain of relational databases [JM95; YL98], Delugach and Hinke [DH96] developed a system that takes the database schema and a knowledge source as input, then informs database administrators about potential inference channels. Their approach is based on conceptual graphs for knowledge representation. Cuppens and Gabillon [CG99; CG01] proposed a method based on coverstories (lies) for closing the inference channels caused by the integrity constraints of a multilevel database. In another work, Cuppens and Gabillon [CG98] came up with a set of rules that must be applied when designing object-oriented databases to ensure multi-level security [Bou+94] and prevent inferences. Chen and Chu [CC08] created a semantic inference model based on data, schema and semantic information which initiated a semantic inference graph to detect inferences while executing queries.

Toland, Farkas, and Eastman [TFE10] extended their previous work [FTE01] on inference control in the presence of database updates, to guarantee confidentiality and maximize availability; a problem that we tackle in our work. Toland, Farkas, and Eastman [TFE05] also presented D²Mon as an extension to an earlier work (DiMon; [BFJ00]) to support database updates. In fact, DiMon itself is an extension of MAC (Mandatory Access Control) where it checks if a query should

be aborted according to some MAC policy, at first. If no MAC policy prohibits the query, then the disclosure engine computes previous queries, the current one and database constraints, and finally the submitted query is re-evaluated by the MAC mechanism.

Katos, Vrakas, and Katsaros [KVK11] proposed an approach to reduce inference control to access control, where they consider the probabilistic correlation between attributes in the inference channel. Another interesting work is that of Miklau and Suci [MS04], who presented an information-theoretic approach to inference control. Indeed, they present the “*query-view security problem*” as follows: *given a set of published views, do they logically disclose information about a given confidential query?* Their work is inspired by Shannon’s work [Sha49] on perfect secrecy. They provide a theoretical foundation for any work that needs to handle information leakage, covering collusions, *a priori* knowledge, and incremental publishing of views.

Staddon [Sta03] presented in her paper a dynamic inference control scheme that does not depend (directly) on user query history, which implies fast processing time, and ensures a crowd-control property: a strong collusion resistance property that not only prevents c collaborating users (where c is the degree of collusion-resistance) from issuing complementary queries to complete an inference channel, but also guarantees that *«if a large number of users have queried all but one of the objects in an inference channel, then no one will be able to query the remaining object regardless of the level of collusion resistance provided by the scheme.»* c -collusion resistance is not desirable in QBA control because it implies that at least one object out of N can never be read by any user.

Chen and Wei [CW05] extended the work of Staddon on dynamic inference control. They have described 2 schemes that prove to be more efficient than Staddon’s which is due to their key allocation scheme. Then they present a third scheme that is resilient to what they call a “*block-an-object*” attack where a malicious user

can exhaust a channel therefore blocking access to the last object for all other database users. Their first 2 schemes can prevent an arbitrary number of collusion, unlike Staddon's, which is c -collusion resistant. The third one guarantees a minimum collusion resistance against c users. The important thing to take from this paper is what they noticed about blocking users and how effectively a time-based key-refreshing scheme should be enforced to prevent not only "block-an-object" attacks, but also blocking users on a set of accessible objects, which might render the application useless after a given period of time.

The problem with such schemes (Staddon and Chen-Wei), other than objects shared among multiple channels, is channel's length itself. It is never clear how channels with varying lengths would be treated, which is very important in a real-life application such as the cadastral database that is subject to daily updates. Not to mention that the method may suffer potential inferences by denial of access. There is no clear solution for such cases. Furthermore, they do not mention external knowledge and how would a security administrator limit inferences by external knowledge; maybe the parameter t they describe in the third scheme can work as a parameter controlling additional inferences from external knowledge.

Another close area of research is controlled query evaluation: CQE [BB04; BT11] which is a form of inference control for logic-based databases. In CQE, user's *a priori* knowledge is taken into account with the history of submitted queries in order to perform inference control. Refusal and lying are employed as means of restriction and perturbation respectively to protect the confidentiality of classified information. CQE cannot identify colluding users. The reader may refer to the PhD thesis of Lochner [Loc11] for further details on inference control (in general) and especially CQE.

Salama, Varadharajan, and Hitchens [SVH12] investigated metadata associated with user published content on the web, more specifically metadata attached to photos (e.g. timestamps, GPS coordinates, etc.) generated by users' cameras and

smartphones and uploaded to social networking sites. They managed to create a set of heuristic rules to improve decision making in forensic investigation. They note, however, that such information can be exploited by malicious attackers to further their aim. Inference in multimedia objects was also treated by Al Bouna and Chbeir [AC09], who propose an approach to detect possible inference channels in multimedia objects by combining their content (unmasked, but potentially masked; e.g. faces) with information available from social networks.

Varadharajan [Var90] tackled inference problems when he presented a model based on Petri nets [DJ01] for information flow security policies, too. In a more recent work, Wietrzyk, Takizawa, and Varadharajan [WTV01] also addressed the issue of inference for multi-level secure distributed workflow systems.

Concerning data publishing, Yang and Li [YL04] and Yixiang, Tao, and Minghua [YTM07a] worked on the inference problem in XML documents, showing how users can use common knowledge in conjunction with partially published documents to infer sensitive data. Staddon, Golle, and Zimny [SGZ07] showed how data from partial documents, when used with a source of knowledge like the web, can be used to infer hidden information.

Inference is also an issue in micro-data *publishing* (privacy preserving data publishing, or PPDP), where Sweeney [Swe00] shows that 87% of the population in the U.S. had reported characteristics that likely made them unique based only on 3 quasi-identifiers {5-digit ZIP, gender, date of birth}. Therefore removing directly identifying attributes (e.g. name or SSN) from the micro-data before publishing is not enough. Techniques such as *k* – *anonymity* [Swe02], *l* – *diversity* [Mac+07], *t*-closeness [LLV07] and *anatomy* [XT06] were developed to prevent these types of inferences, but they target a problem different from ours: these techniques look for the disassociation of data owners and their data, while we want to publish this association as long as it does not violate the given constraints (Pr_1 or Pr_2).

While PPDP focuses on anonymizing datasets before publishing them for later statistical use (by means of generalization, suppression, etc.), privacy preserving data *mining* (or PPDM) does not transform original data before publishing. In fact, in PPDM, data holders provide a querying interface for interested parties so that they can run mining queries on the original data. Data holders must ensure that the result of such queries do not violate the privacy of data subjects. The main technique used is ϵ -differential privacy [Dwo06; Dwo08; Dwo11], that shares a lot of similarities with our approach: limiting (and knowing beforehand) the types of queries permitted to be run on the original data and ensuring collusion resistance [MT07]. However, the problem that ϵ -differential privacy addresses is different from ours: the goal is to use data for statistical purposes, where personal identifying information is not accessible (like PPDP). In addition, ϵ -differential privacy is usually achieved by adding noise to the resulting queries which is unacceptable for our problem.

Clifton and Tassa [CT13] provide an interesting review on what they call “*Syntactic Anonymity*” models (which are models used for PPDP) and differential privacy, their challenges and respective critiques, and shows how these two approaches do not compete (one does not replace the other). Liu, Xiong, and Luo [LXL13] provide a unifying privacy framework for three different privacy definitions found in the literature: Bayes-optimal privacy for privacy preserving data publishing, differential privacy for statistical data release, and privacy with respect to semi-honest behavior in the secure multi-party computation setting [Fri10]. Using this framework, they were able to show that all of these definitions were equivalent.

For a comprehensive overview of inference control in statistical databases, one can refer to the seminal work of Robling Denning [Rob82, Chapter 6]. A more recent work done by Aggarwal and Yu [AY08] covers the advances in PPDP and PPDM. Fung et al. [Fun+10] presents an excellent survey on PPDP, too.

2.3 Aggregation Control

2.3.1 Literature Review

The Encyclopedia of Cryptography and Security mentions the term “Aggregation” twice [van05, pp. 4, 5] in the context of access control, when discussing *Access Control Lists* [SS94] and *Role Based Access Control* [San98]. It was used to denote aggregations of users in groups and roles. The term “Aggregator” was used three times (and “Aggregate” once) to denote information aggregators for financial services [van05, p. 284]. The term “Aggregated” was used to designate «flows [that] are virtual or real network connections that represent aggregated related and concurrent communication» [van05, p. 300]. It is interesting to see that the aggregation problem is not mentioned, even as a subproblem of inference problems, in such a reference work on security.

According to Hinke [Hin88], «the aggregation problem [arises when] aggregates [...] are more sensitive than their constituent parts.» He identifies two types of aggregation problems:

1. Cardinality aggregation, for which he used the classical phonebook problem to explain it. It corresponds to QBA.
2. Inference aggregation, which corresponds to what is currently known plainly as inference.

He argues that both cardinality aggregation and inference aggregation are subclasses of the aggregation problem. He did not work on cardinality aggregation problems because he noted that «[inference aggregation problem] appear to be more tractable. With cardinality aggregation, it is not always clear why “N” elements of a set, such as a phonebook, are classified at one level, while “M” elements are less classified, where cardinality of $N \geq$ cardinality of M .» Indeed, the phonebook example does not induce any special interest, which was apparent while surveying the literature.

The work of Lunt [Lun89] analyses inference and aggregation problems found in multilevel relational databases. She classifies some problems as inference problems and not true aggregation ones, and shows how inference problems can be remedied using proper database design. According to Lunt, the inference problem arises whenever some data x can be used to derive partial or complete information about some other data y , where y is classified higher than x . The aggregation problem arises whenever some collection of facts has a classification strictly greater than that of the individual facts forming the aggregate. To qualify as an aggregation problem, the aggregate class must strictly dominate the class of every subset of the aggregate. Under aggregation problems, she identified quantity-based aggregations (known earlier as cardinality aggregations). A QBA problem occurs whenever a collection of up to k items of a given type is not sensitive, but a collection of greater than k items is sensitive (in the original work she used N).

Jajodia and Meadows [JM95] give another definition of inference problems while surveying the literature on inference control problems in multilevel secure databases. They first introduce the notion of an inference channel, which is a mean by which one can infer data classified at a high level from data classified at a low level. The inference problem is the problem of detecting and removing inference channels. At the end of their paper, they briefly talk about aggregation problems and mention that they are similar to inference problems but not identical. They also show how different strategies could be adopted to control different aggregation problems. They give the following definition of aggregation problems: «*The aggregation problem exists when the aggregate of two or more data items is classified at a level higher than the least upper bound of the classification of the individual items*».

Brewer and Nash [BN89] presented the Chinese-Wall security policy and presented a mathematical theory to implement such a policy. They might be the first to identify a real-world aggregation problem. In fact, the main motivation for the work was to prevent a user from aggregating knowledge in a domain that would

help him learn sensitive information and conduct malicious behavior. However, this approach is very basic in terms of aggregation control. The policy doesn't allow controlling the limit on the number of requested datasets in a single conflict of interest class. The limit is always one dataset per class. Moreover, it doesn't say anything about a single dataset falling in several conflict of interest classes. Collusion is not treated at all, but the main ideas that could be taken from the paper are the following:

1. Their policy provide mandatory access control while always preserving free choice for the user:
 - (a) Who has the right to access any dataset in the same conflict of interest class
 - (b) Whose query behavior decides the set of available datasets and the set of prohibited ones
2. Any system implementing such policies should track user's history

Another notable work on the Chinese-Wall security policy include that of Lin [Lin03] who proposed an aggressive model to overcome one particular issue in Bower and Nash's theory when conflict of interest classes are not mutually exclusive.

In a different work, Meadows [Mea90] gives another definition of the aggregation problem. She says that aggregation issues arise in database security when two or more data items are considered more sensitive together than they are separately. She extends the Bower-Nash model in order to generalize it to multilevel databases. She presents a formal model that is able to handle the Chinese-Wall security policy and other types of aggregation problems. In her model, every object is assigned a security level. Aggregates are assigned a security level too. A security lattice is created from security level labels on objects. Then she defines rules of information flow: a user with a given clearance level can only have access to aggregates of the

same or lower level. Her work requires storing the complete access history of every user. It is best suited for environment where multi level security is required, i.e. where different objects of different security levels form an aggregate with an even higher security level. Collusion is not treated at all.

Cuppens [Cup91] studied the aggregation problem in multilevel databases and proposed a model based on modal logic. In fact, the author starts by proposing his model then shows how it could be instantiated to traditional multilevel security without aggregation. Then he shows how to express the aggregation problem, as presented by Meadows [Mea90], using this modal logic. He notes that «*[in order] to control the aggregation problem, the system must also keep track of the aggregate of all datasets that have previously been accessed by a subject.*»

Bezzi et al. [Bez+10; Bez+12] also treated aggregation problems. Their goal was to prevent statistical inferences. As a matter of fact, they consider that the distribution of soldier's age in a military location can allow inferring the nature of the location itself, whether it is a headquarter or a training campus. Therefore their goal is to perform a k out of N disclosure control such that the distribution of these k records does not resemble the distribution of the sensitive information.

Last but no least, we would like to mention the work of Foley [Fol91; Fol92] that addresses the aggregation problem with information flow policies.

2.3.2 The Work of Motro, Marks and Jajodia

The most relevant work is that of Motro, Marks, and Jajodia [MMJ94]. To the best of our knowledge, this is the only work that tackled QBA directly where they developed a model to handle QBA in relational databases. They start with the following hypotheses:

1. Database:
 - (a) A phonebook is represented by a single relation.
 - (b) Static: instances of the relation are immutable, i.e. they do not consider updates, insertions and deletions.
2. Sensitive aggregates: (“sensitive concepts” in the original work)
 - (a) The complete phonebook; e.g. a user is prohibited from knowing more than k out of N entries in the whole database, or
 - (b) Subsets of the phonebook; e.g. for every subset S_i of the database a user is prohibited from knowing more than k_i out of N_i entries. An example of such a subset can be the set of entries with Postal Code = 1234.
3. A user can execute an “*arbitrary*” query. More precisely, a user can request more than a single entry in the database.
 - (a) Projections and selections are only considered
 - (b) All selections are conjunctions of simple clauses `attribute = value`.

To illustrate their method, they presented the phonebook relation as follows: PHONEBOOK (NAME, TEL, DIV, MAIL, BLDG, ROOM). Table 2.1 shows the PHONEBOOK relation with some example tuples taken from the original work.

Table 2.1: The Phonebook Example

NAME	TEL	DIV	MAIL	BLDG	ROOM
A. Long	x1234	A	m404	1	307
P. Smith	x1111	B	m303	2	610
E. Brown	x2345	B	m101	3	455
C. Jones	x1234	A	m202	1	307
M. Johnson	x1234	B	m101	3	103
B. Stevenson	x2222	A	m202	1	305
S. Quinn	x2222	C	m606	3	101
R. Helmick	x1234	A	m404	1	307
A. Facey	x1122	C	m505	2	400
S. Sheets	x2345	B	m101	2	103

2.3.2.1 The Model

User submitted queries and sensitive aggregates are all considered as views of the database. Views are always represented by their expanded form; e.g. both $\pi_{NAME,ROOM}\sigma_{(ROOM=103)\wedge(DIV=B)}$ and $\pi_{NAME}\sigma_{(ROOM=103)\wedge(DIV=B)}$ describe the same information and they are replaced by their expanded form

$$\pi_{NAME,ROOM,DIV}\sigma_{(ROOM=103)\wedge(DIV=B)}.$$

Every sensitive aggregate is associated with 3 integer values: N , k and D ¹⁰ denoting the number of tuples in that aggregate, the threshold of disclosure and the actual number of disclosed tuples, respectively. For a given sensitive aggregate S : $\|S\| = N$, $k < N$, and the database should always ensure that $D \leq k$.

Subsequently, patterns are introduced as a formal notation of views. Every view is represented by a n -tuple $(p_1, p_2 \dots p_n)$ where n is the number of attributes in the relation and:

$$p_i = \begin{cases} a & \text{if the selection formula}^{11} \text{ includes } A_i = a \\ * & \text{if } A_i \text{ is a projection but not a selection attribute, } 1 \leq i \leq n \\ - & \text{otherwise} \end{cases} \quad (2.1)$$

¹⁰In the original work, k is in fact T . We opted to change it in this explanation for consistency since we use the notion of k out of N disclosure control.

¹¹ A_i is an attribute and a is a constant. Thus, $A_i = a$ is a selection condition.

For instance $\pi_{NAME,ROOM,DIV}\sigma_{(ROOM=103)\wedge(DIV=B)}$ is represented by the following pattern: $(*, -, B, -, -, 103)$. Given a sensitive aggregate S with a pattern $(s_1, s_2 \dots s_n)$ and a user submitted query Q with a pattern $(q_1, q_2 \dots q_n)$, the goal is to determine if the request should be satisfied or not and update the counter of S appropriately. Query Q possibly discloses tuples of S if q_i overlaps s_i for all $1 \leq i \leq n$, that is:

1. q_i is a constant and s_i is either the same constant, * or –
2. q_i is *
3. q_i and s_i are both –

The set of S tuples disclosed by query Q is given by the pattern $(r_1, r_2 \dots r_n)$ where r_i is the restriction of s_i to q_i . This restriction is denoted by $S|Q$ and described as follows:

$$r_i = \begin{cases} q_i & \text{if } q_i \text{ is a constant} \\ s_i & \text{otherwise} \end{cases}, 1 \leq i \leq n \quad (2.2)$$

Finally, consider a sensitive aggregate S and 2 queries Q_1 and Q_2 , where Q_1 overlaps S by m_1 tuples and Q_2 overlaps S by m_2 tuples. It is possible that some of (or all) the tuples disclosed by Q_2 have already been disclosed by Q_1 ; the user is then “charged” twice for the same tuple. To overcome this issue, the authors introduce a predicate P associated with S , initialized to *true*, describing sensitive tuples that have been disclosed. P is the disjunction of all selection attributes that have been previously requested $a_1 \vee a_2 \vee \dots \vee a_p$. When a user submits a query Q_{p+1} , the restriction of S to Q_{p+1} is computed, then the view σ_P is excluded from the restriction. The resulting tuples are those that have not been delivered already.

Furthermore, the authors note that in the presence of multiple sensitive aggregates, some hierarchical relationships between those aggregates might occur. However, this aspect is irrelevant to our discussion. Algorithm 1 shows how QBA is enforced.

Algorithm 1: QBA Control Enforcement Algorithm in the Model of Motro, Marks and Jajodia

```

Input: A user query  $Q$  and a set of sensitive aggregates  $S$ 
Output: Tuples from the database or the empty set  $\emptyset$ 
1 res = Materialize  $Q$ 
2 for all  $S_i \in S$  do
3    $M_i = 0$ 
4   if  $Q$  overlaps  $S_i$  then
5      $M_i = \|(S_i|Q)|\neg\sigma_{p_i}\|$ 
6     if  $D_i + M_i > k_i$  then
7       return  $\emptyset$ 
8 end
9 for  $i = 1, 2 \dots N$ 
10    $D_i = D_i + M_i$ 
11    $P_i = P_i \vee \alpha$ 
12 end
13 return res

```

In their subsequent work [MMJ96], they extend their model to support multi-query attacks. They consider two types of multi-query attacks: *Join* queries and *Complementary* queries. Let us suppose a sensitive aggregate S of the form $\pi_\alpha\sigma_\phi$ (e.g. $\pi_{NAME}\sigma_{(ROOM=307)\wedge(BLDG=1)}$):

- In a *Join* attack, a user submits two queries $Q_1 = \pi_{\alpha_1}\sigma_{\phi_1}$ and $Q_2 = \pi_{\alpha_2}\sigma_{\phi_2}$. $\alpha_1 \cup \alpha_2 = \alpha$ and $\alpha_1 \cap \alpha_2$ contains a key to S . $Q_1 \bowtie Q_2$ yields tuples in S (e.g. $Q_1 = \pi_{NAME,TEL}\sigma_{BLDG=1}$ and $Q_2 = \pi_{NAME,TEL}\sigma_{ROOM=307}$).
- In a *Complementary* attack, a user submits two (or more) queries $Q_1 = \pi_{\alpha'}\sigma_\theta$ and $Q_2 = \pi_{\alpha'}\sigma_{\theta\wedge\neg\phi}$ where θ is less restrictive than ϕ and α' is obtained from α by removing unnecessary selection attributes. $Q_1 - Q_2$ yields tuples in S (e.g. $Q_1 = \pi_{NAME}\sigma_{BLDG=1}$, $Q_2 = \pi_{NAME}\sigma_{(BLDG=1)\wedge(RROOM=305)}$ and $Q_3 = \pi_{NAME}\sigma_{(BLDG=1)\wedge(RROOM=455)}$. $Q_1 - Q_2 - Q_3$ yields tuples in S).

2.3.2.2 Discussion

The work of Motro, Marks and Jajodia starts with a specific hypothesis: users can submit “*arbitrary*” queries. They can select and project over relations (almost) freely. The main difference with our work is that we consider “*point-and-click*” style applications, which means that users cannot select and project as they wish. This difference renders their work unsuitable to our needs. In Appendix B we show how to apply their model to the cadastral application and why it is an unnecessary overhead. The reader is advised to go through Chapter 3 before consulting Appendix B.

Moreover, the proposed model does not deal with the following topics:

1. Collusion resistance,
2. Dynamic resetting: where the complete database is subject to continuous updates, and
3. Time based reset of access. In fact, access to individual entries is not recorded which turns the issue of resetting access problematic:
 - (a) If we want to “*release access*” to entries from oldest to newest, tracking access to individual records is imperative,
 - (b) *Shall we track the newest access to the phonebook only?* If so, we can associate a timestamp to the phonebook instead of each entry in the phonebook and update it to the most recent date it was accessed. This way, the resetting strategy changes altogether, but the question now is: *is such a resetting strategy desirable?*

It was not clear for us how these features would be incorporated in their model.

3

The Model

Before we present our model we will present the definitions of inference, aggregation and quantity based aggregation in Section 3.1. These definitions we derived from earlier work presented in Chapter 2 and from our own understanding of the subject. Our goal is to compare these problems and clarify the differences between them, which was not always clear in the literature. Indeed, confusion between these classes of problems can be easy. To further explain the subtleties of each class of problems we will use illustrating examples based on the classical phonebook example. We will also show how different works described in the literature fall into different definitions.

Afterwards we will present the security policy in Section 3.2, highlighting the fact that both prohibitions form two distinct QBA problems.

Finally, we will show how to enforce Pr_1 and Pr_2 in Sections 3.3 and 3.4, respectively. First of all we will tackle Pr_1 where we will give the basic definition of a region, namely “*zone*” and “*dominant zone*.” Then we will show how to handle collusions using x -collusion resistance. However, x -collusion resistance assumes that all users are potential colluders, therefore we will introduce (x, y, z) -collusion resistance to address this issue which uses the generalization of a zone, namely the

“*z-region*.” Finally, we will show how the model developed for Pr_1 can be applied to Pr_2 and what are the necessary modifications that should be applied.

The model we present here will be the basis of the implementation described in the next chapter which is described in terms of the relational model.

3.1 Definitions

Definition 1 [*Inference problem*] *The inference problem arises whenever a collection of information can be used to derive (infer, deduce) partial or complete knowledge about information stored in the database and classified higher than the classification of each subset of the collection. This collection forms an inference channel. Inference control is a mechanism used to eliminate inference channels and prevent users from performing inferences.*

To illustrate an inference problem, let us consider the phonebook example. A phonebook is represented by the relation PHONEBOOK (NAME, TEL, DEPT) where the classifications of NAME and TEL (say, UNCLASSIFIED) are lower than that of DEPT (say, CONFIDENTIAL). A user with an UNCLASSIFIED clearance can access both NAME and TEL, and naturally DEPT is prohibited. However, if we consider that TEL depends on DEPT (e.g. one telephone per department, or numbers of the same department have the same suffix, etc.), then a user can infer, using NAME + TEL, to which department a given employee is affiliated, or even the list of employees who work in the same department.

Notice that the definition of the inference problem does not specify the source(s) of information in the collection. They could be partially derived from the database as in the inference from external knowledge, where a user combines his *a priori* knowledge with partial knowledge acquired from objects (that s/he has the appropriate clearance to read) of the database to conduct an inference, hence deduce sensitive information.

Definition 2 [*General Aggregation problem*] *The general aggregation problem arises whenever the classification of a collection of k pieces of information, stored in the database, is higher than the classification of each subset. Aggregation control is a mechanism used to prevent users from performing aggregations.*

To illustrate general aggregation problems, let us consider 3 phone entries (tuples): A , B and C labeled SECRET. The aggregation of A and B is labeled TOP SECRET while the aggregation of A and C is labeled SECRET. A user with a SECRET clearance level should not access the aggregate $A + B$ but s/he can access $A + C$.

Definition 3 [QBA problem] *The QBA problem arises whenever the classification of more than k out of N items in a database is higher than the classification of that of k or less items. QBA control is a mechanism used to prevent users from aggregating more than k out of N items.*

Indeed, a QBA problem arises when a user has the right to query any subset of the phonebook relation (of size N), under the condition that the size of the queried subset does not exceed k (where $k < N$). The key difference between Definitions 2 and 3 is that the former does not take into account the quantity of aggregated entries. If we apply these definitions to works found in the literature, we find that inferences from dependencies on schema and data [e.g. YL98; YTM07b; CC08], or inferences from external knowledge [e.g. SGZ07] or denial of access [e.g. SJ92] fall under Definition 1. The Chinese-Wall policy [e.g. BN89; Mea90] falls under Definition 2 where the role of the policy is preventing a user from aggregating data from the same conflict of interest class, while the phonebook problem as presented by Hinke [Hin88] and Lunt [Lun89], and both aggregation problems of the cadastral database—that we will define in the next section—fall under Definition 3.

3.2 Security Policy

A cadastral database is a geographical database used to manage parcels of a country, state, municipality, etc. Parcels are pieces of land represented in the database by geo-referenced polygons. In addition to their geometric representation, parcels are associated with information like mutation history, taxation, and most importantly ownership information. Currently, access to the cadastral database in French Polynesia is limited to employees of the real-estate service, notaries, and surveyors. The computer science department of French Polynesia wishes to make this database available online and apply the following **Security Policy**: citizens (parcel owners or not) can access ownership information of any parcel through a “*point-and-click*” mapping interface (similar to Google Maps or Bing Maps). However, this access is limited by the following prohibitions:

Pr₁: A user cannot get the list of all owners in a geographical region.

Pr₂: A user cannot get the list of all parcels belonging to the same legal entity (e.g. family).

It should be obvious by now that both Pr₁ and Pr₂ are two separate QBA problems. Indeed, we have the following analogies:

- The list of owners of a given region is analogous to a phonebook (Pr₁).
- The list of parcels of a given family is analogous to a phonebook (Pr₂).
- The association between a parcel and an owner is analogous to a phonebook entry.

The violation of these prohibitions entails the risk of violation of the privacy of owners. Indeed, some commercial agencies can take advantage of ownership information in a region (Pr_1) to send targeted advertising. Moreover, real-estate agencies can exploit the list of owners in a region to contact these owners and try to strike deals with them. The aforementioned case should be prevented because real-estate agencies have an advantage over owners who might not be well-informed on the real-estate market and price changes. For example, a real-estate agency can know, from experience and specialized knowledge, that a certain political or social event (e.g. construction of a highway or a resort nearby) might cause the prices of parcels in some regions to increase in the near future. If Pr_1 is not implemented, then the agency can contact all owners in the designated region to buy their parcels at low prices.

Additionally, the publication of the list of goods belonging to a person is considered—especially in France and in French Polynesia—a violation of the privacy of that person. Organizations are prohibited from publishing the list of bank accounts belonging to a person, or the list of apartments he owns, etc. Parcels are goods that should be protected in the same manner. The complete list of parcels belonging to the same owner (Pr_2) should not be accessible otherwise we risk violating the privacy of the owner.

3.3 Enforcing Pr_1

The first challenge is to properly interpret the term “*region*.” The obvious (and naive) solution is to consider that one administrative region is equivalent to a region. This is the static definition of a region, but it is inaccurate: *Which resolution is considered optimal? Is a municipality too big? Is a neighborhood too small? Should we mix resolutions?* If we choose, for the sake of argument, a neighborhood as our definition for a region, and we gave all users the right to access k out of N parcels for every neighborhood, then a malicious user can exploit the fact that regions are static, and attack a “*geographical space*” falling on the shared borders of two neighboring regions, thus knowing the owners of that “*geographical space*” that s/he considers a region.

One should understand that people’s perception of a region is dynamic itself. Moreover, every person has multiple definitions of regions, and they are all based on personal interest; it could be economical, social, or even contextual (e.g. a region that has been featured in the news). For example, Alice finds that the beach strip is interesting because she works at a construction agency seeking a spot for its new hotel; Bob finds a remote house on the hill interesting because he wants to buy it with the part of the hill facing the sea; Charlie is interested by the economical section of the city because he wants to invest in real-estate while on a tight budget; etc. In all of those cases, there is a non-negligible chance that the user’s region of interest is distributed between multiple connected static regions.

Therefore we need a definition of a “*region*” that overcomes both problems: it needs to be resolution independent and dynamic, elastic, so that it can adapt to the human’s perception of a region, regardless of the previously mentioned subjective interest (economic, social, etc.). To that end, we use the following definition:

Definition 4 [*Zone*] A zone of a parcel p is formed by p itself and all of its neighbors.

A zone is the smallest region that could be modeled; every parcel belongs to its proper zone and the zone of every direct neighbor. Let P be the set of parcels in the cadastral database and O the set of owners; o_p denotes the set of owners of a parcel p ; $\delta(., .)$ is a function that returns the minimal euclidean distance between two parcels. We create a graph $G(V, E)$ where $G(V) = P$, while $G(E)$ is defined as follows: two parcels are neighbors in the graph if they touch each other, or if they are separated by a maximal distance τ . Formally $G(E) = \{(p, q) : p, q \in G(V), p \neq q, 0 \leq \delta(p, q) \leq \tau\}$ for a given $\tau \in \mathbb{R}_{\geq 0}$. We could select $\tau = 0$, i.e. only parcels touching each other, however parcels which are separated by thin boundaries, like rivers or roads, require a value of τ greater than 0 to be considered as neighbors. We consider that “*isolated*” parcels, i.e. parcels that do not have neighbors in the range τ , do not fall within the scope of Pr_1 : access to these parcels is granted automatically.

Figure 3.1 shows a graph for Pr_1 representing part of the cadastral database, where parcel 1 touches $\{2, 3, 8\}$, parcel 4 touches $\{5\}$, parcel 7 touches $\{3, 5, 6, 8\}$, etc. Notice that every parcel belongs to its proper zone and to every zone formed by every neighboring parcel.

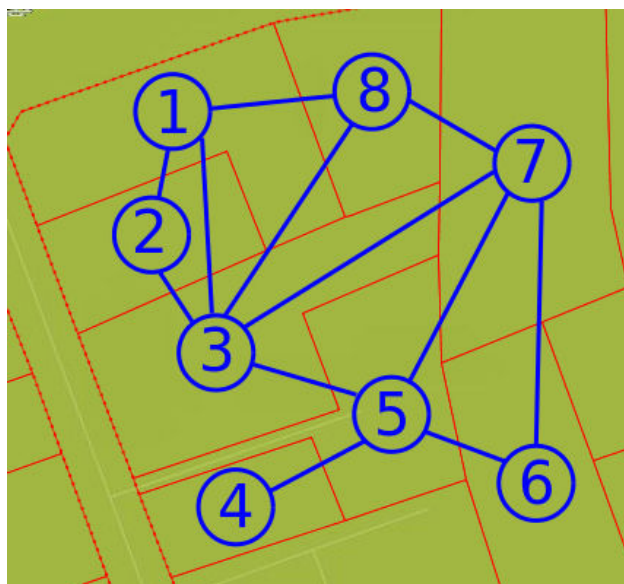


Figure 3.1: A graph for Pr_1

Definition 5 [*Dominant Zone*] *The dominant zone of a parcel p , dom_p , is the zone containing p having the highest cardinality. A parcel can have multiple dominant zones.*

The zone with the highest cardinality is the zone that contains the greatest number of parcels. Let $Dom = \{dom_i : i = 1, 2 \dots M\}$ be the set of all dominant zones in the database; $|dom_i| = N_i$. A user has the right to know the ownership of any parcel belonging to any dominant zone in the database.

The **Aggregation Control Property** is: for all dom_i , the number of disclosed parcels for any user, k_i , should always be strictly lower than N_i . Satisfying the aggregation control property, namely preventing a user from accessing all parcels in a dominant zone, implies the satisfaction of the security policy and effectively preventing this user from acquiring the knowledge of all owners in any region of any size.

Enforcing QBA control is simple: when a user requests a parcel p_i , the algorithm should make sure that the number of disclosed parcels is strictly lower than k_i for the dominant zone of the requested parcel, dom_i , and all dominant zones containing it, i.e. dominant zones of its direct neighbors containing the requested parcel. If this condition is satisfied, access is granted; otherwise, access is denied. In order to satisfy the security policy: for every dominant zone dom_i , k_i must satisfy: $0 < k_i < N_i$.

This is sufficient if we consider a single user accessing the cadastral database in isolation. Let us take an example showing why this condition is not sufficient if we consider multiple users: if two users are accessing a dominant zone where $k_i = N_i - 1$, none of them can get the ownership information of all parcels in that dominant zone, however, they could collaborate and combine their knowledge to bypass the limit k_i . Therefore $0 < k_i < N_i$ is a necessary but not sufficient condition in real-world applications where collaborating users form an actual threat to the security of the application.

This collaboration is called “*collusion*.” The Merriam-Webster online dictionary defines collusion¹ as «*[a] secret agreement or cooperation especially for an illegal or deceitful purpose.*» In our context, the illegal or deceitful purpose is to access a complete dominant zone. Therefore a collusion happens when x users secretly agree or cooperate to access a given dominant zone. An important property that should be satisfied by QBA control is collusion resistance.

Definition 6 [x -collusion] *We say that x users collude to reconstruct all entries in a dominant zone if the union of accessed parcels by those x users covers the complete dominant zone.*

A QBA control mechanism is x -collusion resistant if x or fewer users cannot reconstruct a complete dominant zone. To achieve x -collusion resistance the **Aggregation Control Property** should be extended to:

$$k_i = \begin{cases} \lceil N_i/x \rceil - 1 & \text{if } N_i > x > 1 \\ 1 & \text{otherwise} \end{cases} \quad (3.1)$$

This way, x users are guaranteed to never collude and reconstruct a complete dominant zone even if those x users accessed disjoint subsets of dom_i .

Now we should analyze x -collusion resistant QBA control. In fact, we should evaluate the effect of the variation in the size of dominant zones. Since we proposed a solution to achieve x -collusion resistance that relies on N_i , and N_i is variable due to the dynamic nature of the database (deletions, insertions, and divisions of existing parcels), then we should see which values would vary with respect to N_i . At any given moment x users or fewer can never collude to get the complete set of records, therefore k_i should vary with N_i .

1. If k_i increases then users have access to more parcels. This means that users might collude to reconstruct the previous dominant zone dom_i before it expanded. From a security point of view this means that the previous zone has

¹<http://www.merriam-webster.com/dictionary/collusion>

been “*declassified*.” If we wish to avoid this situation then the only solution would be enforcing smaller values of k_i :

$$k_i = \begin{cases} \lceil N_i/x \rceil - \alpha & \text{if } \lceil N_i/x \rceil > \alpha > 0 \\ 1 & \text{otherwise} \end{cases} \quad (3.2)$$

With α carefully chosen, expanding dominant zones dom_i do not allow colluding users to reconstruct today information that was considered as sensitive yesterday.

2. In the case where the value of k_i decreases to k'_i , then users who have already accessed more than k'_i parcels might collude to reconstruct the new dominant zone dom'_i . Here also, computing smaller values of k_i (i.e. choosing a proper value for α) would eliminate this security threat.

Now let us consider the set of all M dominant zones under x -collusion resistance. If x has the same value for all dominant zones, then x or fewer users are guaranteed not to collude on all M zones. If the security administrator sets for zones $dom_1, dom_2 \dots dom_M$ different values $x_1, x_2 \dots x_M$ then a coalition of x_c users, where $\min(x_1, x_2 \dots x_M) < x_c < \max(x_1, x_2 \dots x_M)$ can collude to reconstruct all dominant zones with x_b -collusion resistance such as $x_b \leq x_c$. Therefore, we recommend setting a single value of x -collusion resistance to all zones.

The drawback of x -collusion resistance is that it assumes that all users are potential colluders on all dominant zones. In practice this assumption is somewhat too strong and may lead the QBA control mechanism to detect too many false positives. Another type of collusion resistance is needed where a user is assumed to be a potential colluder if his/her querying behavior is suspicious. This new type should take into account the main idea behind Pr_1 , while relaxing the assumption on colluding users: recall that Pr_1 states that «*a user cannot get the list of all owners in*

a geographical region.» Therefore, a group of users should be considered as potential colluders if they are trying to attack a region, the more general concept of a zone.

Definition 7 [*z-region*] a *z-region* of a parcel p is formed by dom_p and the set of dominant zones of neighbors of p of degree $\leq z$.

Definition 8 [*(x, y, z)-collusion*] We say that x users collude to reconstruct y dominant zones in a *z-region* if the union of accessed parcels by those x users covers those y complete dominant zones.

To achieve (x, y, z) -collusion resistance, the **Aggregation Control Property** becomes: for any *z-region* R_i , where $|R_i| = M_i$:

1. A user is prohibited from querying more than k_i parcels in y dominant zones of R_i such as

$$k_i = \begin{cases} N_i - \alpha & \text{if } N_i > \alpha > 0 \\ 1 & \text{otherwise} \end{cases} \quad (3.3)$$

2. It enforces x -collusion resistance on the remaining $M_i - y$ dominant zones of R_i according to equation 3.2.

The idea behind (x, y, z) -collusion resistance is that as long as users cannot reconstruct more than y dominant zones in a given region then they should not be considered as colluders. As soon as these users can reconstruct y dominant zones in a given region then the x -collusion resistance scheme should be applied on the remaining $M_i - y$ dominant zones.

To support (x, y, z) -collusion resistance, k should be split into 2 variables: k_h (read K HIGH) and k_l (read K LOW). A user has the right to access up to k_h entries in y dominant zones in any *z-region* R_i of a parcel, after which s/he is considered a potential colluder. For the remaining $M_i - y$ dominant zones s/he has the right to access k_l entries, where $k_h > k_l$. The security administrator sets k_h according to equation 3.3, and k_l is determined by the required level of x -collusion resistance as defined in equation 3.2.

3.4 Enforcing Pr_2

The basic idea for enforcing Pr_2 is to use the scheme we developed for Pr_1 in the previous section by only modifying the graph definition as follows: two parcels are considered neighbors in the graph if they belong to the same owner. In such a graph, vertices belonging to the same owner are all interconnected, forming a complete graph. Formally, $G(E) = \{(p, q) : p, q \in G(V), p \neq q, o_p \cap o_q \neq \emptyset\}$. For instance, Figure 3.2 shows a graph for Pr_2 representing part of a cadastral database (the same part as in Figure 3.1), where parcels $\{1, 2, 8\}$ are owned by Joe, $\{4, 5, 6, 7\}$ are owned by Elissa, and $\{3, 5\}$ are owned by Lucy. Notice that parcel 5 has two owners, namely Elissa and Lucy.

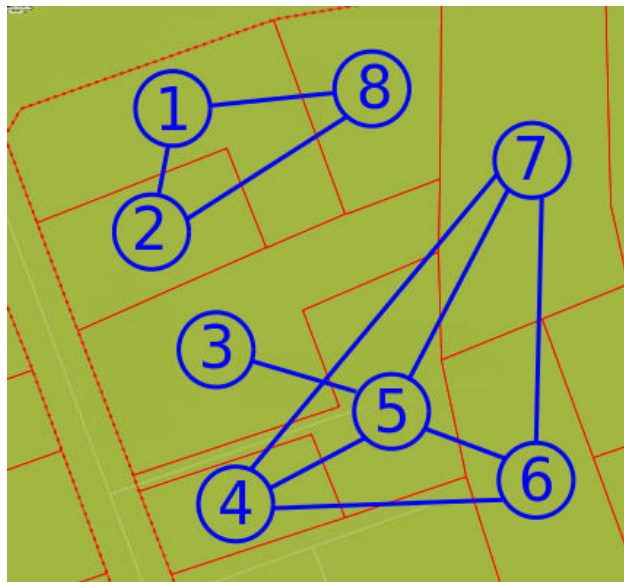


Figure 3.2: A graph for Pr_2

It is clear that a zone, as presented in Definition 4, depends only on the graph: for Pr_1 , the zone of a parcel p is p and the set of parcels touching, or located at a given distance from p ; for Pr_2 , the zone of a parcel p is p and the set of parcels owned by the same legal entity. The dominant zone of a parcel p is the zone containing p having the highest cardinality, i.e. the owner that owns the highest number of parcels.

x -collusion resistance also holds. The **Aggregation Control Property** should conform with equations 3.2. To achieve (x, y, z) -collusion resistance, we define a distance function $dist_{social}$ as follows:

$$dist_{social} : V^2 \rightarrow \mathbb{N} \quad (3.4)$$

$dist_{social}$ returns the smallest social distance between the owners of 2 parcels according to some social relationship (e.g. father, grand-child, etc.). This distance function is essential to the new definition of a z -region in Pr_2 :

Definition 9 [z -region] *Is defined as a subset of the database where the distance $dist_{social}$ between any two parcels belonging to the subset is lower than z . Formally, a set V' of parcels belongs to a z -region if, and only if,*

$$v_m, v_n \in V', v_m \neq v_n, 1 \leq dist_{social}(v_m, v_n) \leq z \quad (3.5)$$

Similarly to Pr_1 , and in order to support (x, y, z) -collusion resistance, k should be split into 2 variables: k_h (read K HIGH) and k_l (read K LOW). A user has the right to access up to k_h entries in y zones in any z -region R_i of a parcel, after which s/he is considered a potential colluder. For the remaining $M_i - y$ zones s/he has the right to access k_l entries, where $k_h > k_l$. The security administrator sets k_h according to equation 3.3 and k_l is determined by the required level of x -collusion resistance as defined in equation 3.2. Note that we consider that “isolated” parcels, i.e. a parcel belonging to a single owner who himself does not own other parcels, do not fall within the scope of Pr_2 : access to these parcels is granted automatically.

4

Implementing the Security Policy

In this chapter we will present implementation details of the model we developed in the previous chapter for Pr_1 and Pr_2 , in Sections 4.1 and 4.2 respectively.

Our solution is described in the relational model, facilitating the integration with the existing cadastral database of French Polynesia. For every prohibition, we will show the database schema we are considering, in addition to the views that should be developed to ensure QBA enforcement. These views are not actually implemented in our prototype and should not be implemented in a production environment for performance reasons as we will show in Chapter 8. However, we decided to include them since they provide unambiguous semantics of the actual database schema we use. Additionally, the QBA enforcement algorithm—expressed in relational algebra—for each prohibition is included.

These schemas and enforcement algorithms are used as the basis of our prototype described in Chapter 7. The actual SQL enforcement script used by our prototype, which is based on the enforcement algorithm, is presented in Appendix C.

Figure 4.1 shows the entity-relationship diagram of the database we are considering. Listing 4.1 shows the corresponding tables (underlined attributes represent primary keys and those prefixed with # represent foreign keys).

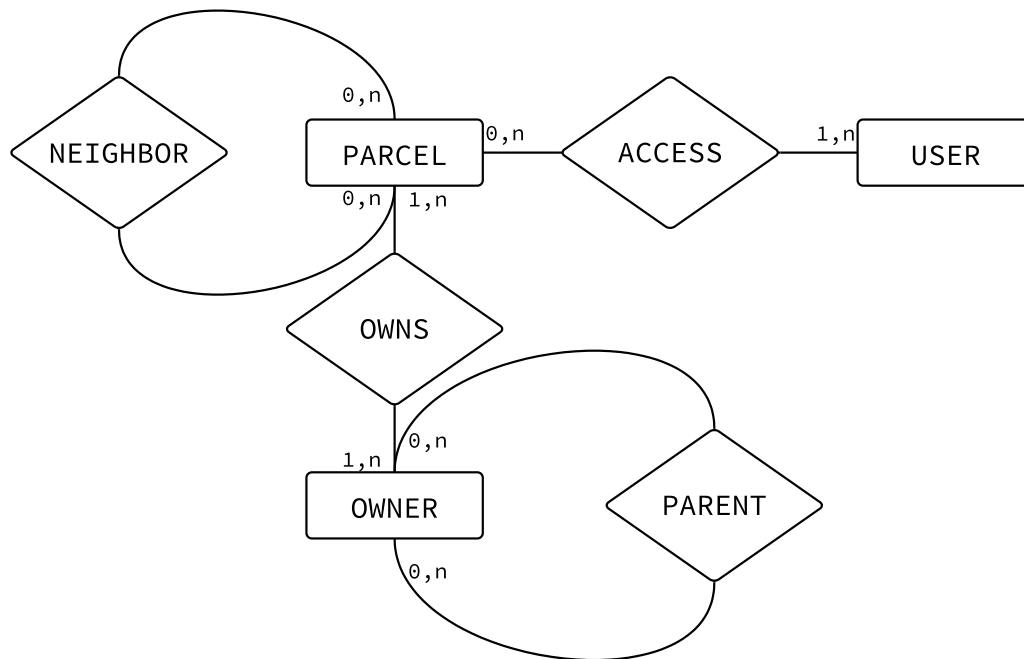


Figure 4.1: Entity-Relationship Diagram of the Database

Listing 4.1: Database Tables Corresponding to the ERD of Figure 4.1

```

1 PARCEL (PID ,GEOMETRY )
2 NEIGHBOR (#PID1 ,#PID2 )
3 ACCESS (#PID ,#UID ,TIMESTAMP)
4 USER (UID ,NAME )
5 OWNER (OID ,NAME )
6 OWNS (#PID ,#OID )
7 PARENT (#OID1 ,#OID1 )

```

4.1 Implementing Pr_1

The views of Listing 4.2 are derived from the tables of Listing 4.1 (we are considering that $z = 2$, $\alpha = 1$). Let us comment it line by line:

1-4 1-REGION represents 1-regions, i.e. zones. Each zone is identified by a parcel (PID1) which is linked to itself and its neighbors (PID2).

6-10 1-REGION-CPT computes for every 1-region its k_l and k_r . Notice that for k_l we set α to 1.

12-19 DOMINANT-ZONE returns the set of dominant zones of every parcel. For each parcel PID1 it gives the set of ID (PID2) identifying zones which are the dominant zones of parcel PID1.

21-25 2-NEIGHBOR returns 2nd degree neighbors of a parcel (excluding 1st degree neighbors). Notice that we restricted the view to $z = 2$. We do the same thing for 2-REGION and 2-REGION-YDISCLOSED as we will show later.

27-30 2-REGION returns the set of 1-regions of 2nd degree neighbors.

32-40 1-REGION-DISCLOSED returns, for every 1-region, the number of **DISCLOSED** parcels for a given user.

42-49 2-REGION-YDISCLOSED returns **YDISCLOSED**, the number of 2-regions where DISCLOSED is greater than k_l .

51-55 ZONE-USER represents user access history on every zone. QBA Enforcement should always make sure that DISCLOSED is less than or equal to KH and YDISCLOSED is less than y (of (x, y, z) -collusion resistance).

Listing 4.2: Views for Pr₁

```

1 CREATE VIEW 1-REGION AS
2   SELECT PID AS PID1, PID AS PID2
3   FROM   PARCEL
4   UNION (SELECT PID1, PID2 FROM NEIGHBOR);
5
6 CREATE VIEW 1-REGION-CPT AS
7   SELECT P.PID, (COUNT(*)/x)-1 AS KL, COUNT(*)-1 AS KH
8   FROM   PARCEL P, 1-REGION R
9   WHERE  P.PID = R.PID1
10  GROUP BY P.PID;
11
12 CREATE VIEW DOMINANT-ZONE AS
13   SELECT 1R.PID1, 1R.PID2
14   FROM   1-REGION 1R, 1-REGION-CPT 1RC
15   WHERE 1R.PID2 = 1RC.PID
16   AND   1RC.KH = (SELECT MAX(1RC2.KH)
17                  FROM   1-REGION 1R2, 1-REGION-CPT 1RC2
18                  WHERE  1R2.PID1 = 1R.PID1
19                  AND   1R2.PID2 = 1RC2.PID);
20
21 CREATE VIEW 2-NEIGHBOR AS
22   SELECT N1.PID1, N2.PID2
23   FROM   NEIGHBOR N1, NEIGHBOR N2
24   WHERE  N1.PID2 = N2.PID1
25   AND   N1.PID1 <> N2.PID2;
26
27 CREATE VIEW 2-REGION AS
28   SELECT PID1,PID2
29   FROM   1-REGION
30   UNION (SELECT PID1, PID2 FROM 2-NEIGHBOR);
31
32 CREATE VIEW 1-REGION-DISCLOSED AS
33   SELECT DZ.PID1, A.UID, COUNT(*) AS DISCLOSED
34   FROM   DOMINANT-ZONE DZ, ACCESS A
35   WHERE  DZ.PID2 = A.PID
36   GROUP BY DZ.PID, A.UID
37   UNION
38   SELECT P1.PID, U.UID, 0
39   FROM   PARCEL P1, USER U
40   WHERE (P1.PID, U.UID) NOT IN (SELECT PID, UID FROM ACCESS);
41
42 CREATE VIEW 2-REGION-YDISCLOSED AS
43   SELECT 2R.PID1, 1RD.UID, COUNT(*) AS YDISCLOSED
44   FROM   2-REGION 2R, 1-REGION-DISCLOSED 1RD
45   WHERE  2R.PID2 = 1RD.PID
46   AND   1RD.DISCLOSED > (SELECT KL
47                          FROM   1-REGION-CPT

```

```

48         WHERE PID = 1RD.PID)
49     GROUP BY 2R.PID1, 1RD.UID;
50
51 CREATE VIEW ZONE-USER AS
52     SELECT 1RD.PID, 1RD.UID, DISCLOSED, YDISCLOSED
53     FROM 1-REGION-DISCLOSED 1RD, 2-REGION-YDISCLOSED 2RD
54     WHERE 1RD.PID = 2RD.PID
55     AND 1RD.UID = 2RD.UID;

```

However we did not implement any of these views as such for performance reasons. In fact, in Chapter 8, Section 8.2.2 on page 110, we show that computing dominant zones on-the-fly alone, as opposed to storing them in their own relation, introduces a significant performance hit. We included them here since they give unambiguous semantics of the set of relations listed in Listing 4.3.

Relations in Listing 4.3 are used and updated by our QBA control algorithm (Algorithm 2). The main modification lies in the replacement of ACCESS by PARCEL-USER and ZONE-USER: the former records user access history on the parcel level, and the latter records that history on the zone and region level. We also added DOMINANT-ZONE that records for every parcel, the set of its dominant zones.

Listing 4.3: Actual Database Relations for Pr_1

1	PARCEL	(<u>PID</u> , GEOMETRY , KH , KL)
2	NEIGHBOR	(#PID1 , #PID2)
3	DOMINANT-ZONE	(#PID1 , #PID2)
4	USER	(<u>UID</u> , NAME)
5	PARCEL-USER	(#PID , #UID , TIMESTAMP)
6	ZONE-USER	(#PID , #UID , DISCLOSED , YDISCLOSED)
7	OWNER	(<u>OID</u> , NAME)
8	OWNS	(#PID , #OID)

Algorithm 2 is the one used to enforce Pr_1 . As a matter of fact, this algorithm enforces (x, y, z) -collusion resistance; if we eliminate lines 11 to 18, the algorithm ensures x -collusion resistance only (and in this case, KH is set according to equation 3.1 or 3.2). The algorithm does the following (every item in the list explains a corresponding line):

- 1 `result` is the set of owners of the requested parcel retrieved from the user's access history.
- 3–4 If `result` is not empty, i.e. the user has already queried the requested parcel, it is returned immediately.
- 5 `requestedZones` is the zone of the requested parcel (see Definition 4).
- 6 `requestedDZ` is the set of dominant zones containing the requested parcel.
- 7 `maxedZones` is the set of dominant zones falling in the zone of the requested parcel, where the user have reached the limit KH .
- 9–10 If `maxedZones` is positive, i.e. the disclosure of the owner of the requested parcel will give the user the knowledge of more than k_{i_h} out of N_i parcels in any dominant zone, access is denied, and the empty set is returned.
- 11 `potential` is the set of requested parcels where the user has reached the lower limit KL ; they represent dominant zones that would potentially, in case access was granted, be counted among the allowed y parcels.
- 13–14 For every potential parcel p , perform a breadth-first traversal (BFS), where p is the root node, stop at depth z , and increment the value of `YDISCLOSED` for all visited nodes.
- 16 We count the number of regions that have `YDISCLOSED` greater than y and store it in `maxedRegions`.
- 17–18 If the number of `maxedRegions` is grater than 0, i.e. the disclosure of the owner of the current parcel would give the user the knowledge of more than k_l parcels in more than y zones, access is denied, all operations are rolled-back, and the empty set is returned.

- 19** At this stage the user has never exceeded KH parcels in the zone of the requested parcel, and never exceeded y in any region, therefore access is granted, and DISCLOSED of all parcels should be incremented. Entries in ZONE -USER and PARCEL -USER are inserted or updated as necessary.
- 20** Return the set of owners of the requested parcel.

Algorithm 2: QBA Control Enforcement Algorithm for Pr_1

```

Input: parcelID, userID, y, z
Output: owners
1 result  $\leftarrow \pi_{NAME} (\sigma_{(PID=parcelID) \wedge (UID=userID)} (OWNER \bowtie OWNS \bowtie$ 
2 PARCEL-USER))
3 if result is not  $\emptyset$  then
4   return result
5 requestedZones  $\leftarrow (\pi_{PIDN} \sigma_{PID=ParcelID} NEIGHBOR) \cup \{ParcelID\}$ 
6 requestedDZ  $\leftarrow (\pi_{PIDN} DOMINANT-ZONE) \cap$  requestedZones
7 maxedZones  $\leftarrow |\sigma_{(UID=userID) \wedge (DISCLOSED=KH)} (requestedDZ \bowtie$ 
8 ZONE-USER  $\bowtie$  PARCEL)|
9 if maxedZones > 0 then
10  return  $\emptyset$ 
11 potential  $\leftarrow \pi_{PID} (\sigma_{(UID=userID) \wedge (DISCLOSED=KH)} (requestedDZ \bowtie$ 
12 ZONE-USER  $\bowtie$  PARCEL))
13 for all p  $\in$  potential do
14   IncrementYDisclosed(userID, BFS(p, z))
15 end
16 maxedRegions  $\leftarrow |\sigma_{(UID=userID) \wedge (YDISCLOSED > y)} ZONE-USER |$ 
17 if maxedRegions > 0 then
18   Rollback and return  $\emptyset$ 
19 Update(parcelID, userID)
20 return  $\pi_{NAME} (\sigma_{PID=parcelID} (OWNER \bowtie OWNS))$ 

```

4.2 Implementing Pr_2

Similar to what we presented in Section 4.2, Listing 4.4 is derived from the database of Listing 4.1, with a single modification on ACCESS that becomes ACCESS(OID, PID). In the following we explain the views of Listing 4.4 (we are considering that $z = 2, \alpha = 1$):

1-4 OWNER-CPT returns for every owner its k_l and k_h .

6-14 DOMINANT-ZONE returns the set of dominant zones (owners) for every parcel. For each parcel PID it gives the set of ID (OID) identifying zones (owners) which are the dominant zones of parcel PID.

16-19 FAMILY returns 1st relatives of an owner.

21-24 2-PARENT returns 2nd degree neighbors of a parcel (excluding 1st degree relatives).

26-29 LARGE-FAMILY returns the set of 1st and 2nd degree relatives.

31-39 OWNER-DISCLOSED returns, for every owner, the number of **DISCLOSED** parcels for a given user.

41-48 LARGE-FAMILY-DISCLOSED returns **YDISCLOSED**, the number of owners in the large family where DISCLOSED is greater than k_l .

50-53 OWNER-USER represents user access history on every zone. QBA Enforcement should always make sure that DISCLOSED is less than or equal to KH and YDISCLOSED is less than y (of (x, y, z) -collusion resistance).

Listing 4.4: Views for Pr₂

```

1 CREATE VIEW OWNER-CPT AS
2   SELECT  OID, (COUNT(*)/x)-1 AS KL, COUNT(*)-1 AS KH
3   FROM    OWNS
4   GROUP BY  OID;
5
6 CREATE VIEW DOMINANT-ZONE AS
7   SELECT  O1.PID, O1.OID
8   FROM    OWNS O1, OWNER-CPT O2
9   WHERE   O1.OID = O2.OID
10  AND     O2.KH = (SELECT MAX(O3.KH)
11              FROM  OWNER-CPT O3
12              WHERE O3.OID IN (SELECT O4.OID
13                          FROM  OWNS O4
14                          WHERE O4.PID = O1.PID));
15
16 CREATE VIEW FAMILY AS
17   SELECT  OID AS OID1, OID AS OID2
18   FROM    OWNER
19   UNION  (SELECT OID1,OID2 FROM PARENT);
20
21 CREATE VIEW 2-PARENT AS
22   SELECT  P1.OID1, P2.OID2
23   FROM    PARENT P1, PARENT P2
24   WHERE   P1.OID2=P2.OID1 AND P1.OID1 <> P2.OID2;
25
26 CREATE VIEW LARGE-FAMILY AS
27   SELECT  OID1,OID2
28   FROM    FAMILY
29   UNION  (SELECT OID1, OID2 FROM 2-PARENT);
30
31 CREATE VIEW OWNER-DISCLOSED AS
32   SELECT  O.OID, A.UID, COUNT(*) AS DISCLOSED
33   FROM    DOMINANT-ZONE O, ACCESS A
34   WHERE   O.PID=A.PID
35   GROUP BY O.OID, A.UID
36   UNION
37   SELECT  O1.OID, U.UID, 0
38   FROM    OWNER O1, USER U
39   WHERE   (O1.OID, U.UID) NOT IN (SELECT OID, UID FROM ACCESS);
40
41 CREATE VIEW LARGE-FAMILY-YDISCLOSED AS
42   SELECT  LF.OID1, OD.UID, COUNT(*) AS YDISCLOSED
43   FROM    LARGE-FAMILY LF, OWNER-DISCLOSED OD
44   WHERE   LF.OID2=OD.OID
45   AND     OD.DISCLOSED > (SELECT KL
46              FROM  OWNER-CPT
47              WHERE  OID=OD.OID)

```

```

48     GROUP BY LF.OID1, OD.UID;
49
50 CREATE VIEW OWNER-USER AS
51     SELECT OD.OID, OD.UID, DISCLOSED, YDISCLOSED
52     FROM   OWNER-DISCLOSED OD, LARGE-FAMILY-YDISCLOSED LF
53     WHERE  OD.OID=LF.OID AND OD.UID=LF.UID;

```

However, for the same reasons mentioned in the previous section, we opted to use the schema shown in Listing 4.5. The main modification lies in the modification of OWNER and the creation of OWNER-USER. We also added DOMINANT-ZONE that records for every parcel, the set of dominant zones.

Listing 4.5: Actual Database Relations for Pr_2

```

1 PARCEL      (PID      ,GEOMETRY      )
2 NEIGHBOR    (#PID1   ,#PID2      )
3 DOMINANT-ZONE (#PID    ,#OID      )
4 USER        (UID      ,NAME      )
5 OWNER-USER  (#OID    ,#UID    ,DISCLOSED ,YDISCLOSED)
6 OWNER       (OID      ,NAME    ,KL      ,KH      )
7 OWNS        (#PID    ,#OID      )

```

Algorithm 3 is the one used to enforce Pr_2 . As a matter of fact, this algorithm enforces (x, y, z) -collusion resistance; if we eliminate lines 10 to 17, the algorithm ensures x -collusion resistance only (and in this case, KH is set according to equation 3.1 or 3.2. The algorithm does the following (every item in the list explains a corresponding line):

- 1 `result` is the set of owners of the requested parcel retrieved from the user's access history.
- 2-3 If `result` is not empty, i.e. the user has already queried the requested parcel, it is returned immediately.
- 4 `requestedOwners` is the set of owners of the requested parcel.
- 6 `maxedZones` is the number of owners (among `requestedOwners`) where the user has reached the limit KH.

- 8–9** If `maxedZones` is positive, i.e. the disclosure of the owner of the requested parcel will give the user the knowledge of more than k_{ih} out of N_i parcels in any zone, access is denied, and the empty set is returned.
- 10** `potential` is the set of requested zones where the user has reached the lower limit `KL`; they represent the zones that would potentially, in case access was granted, be counted among the allowed `y` parcels.
- 12–13** For every potential parcel p , perform a breadth-first traversal (BFS), where p is the root node, stop at depth z , and increment the value of `YDISCLOSED` for all visited nodes.
- 15** We count the number of regions that have `YDISCLOSED` greater than `y` and store it in `maxedRegions`.
- 16–17** If the number of `maxedRegions` is greater than 0, i.e. the disclosure of the owner of the current parcel would give the user the knowledge of more than `KL` parcels in more than `y` zones, access is denied, all operations are rolled-back, and the empty set is returned.
- 18** At this stage the user has never exceeded `KH` parcels in the zone of the requested parcel, and never exceeded `y` in any region, therefore access is granted, and `DISCLOSED` of all zones (owners in this case) should be incremented. Entries in `OWNER-USER` and `PARCEL-USER` are inserted or updated as necessary.
- 19** Return the set of owners of the requested parcel.

Algorithm 3: QBA Control Enforcement Algorithm for Pr_2

```

Input: parcelID, userID, y, z
Output: Owners
1 result  $\leftarrow \pi_{NAME} (\sigma_{(PID=parcelID) \wedge (UID=userID)} (OWNS \bowtie PARCEL-USER))$ 
2 if result is not  $\emptyset$  then
3   return Result
4 requestedOwners  $\leftarrow \pi_{OID} \sigma_{PID=parcelID} OWNS \cap$ 
5    $\pi_{OID} DOMINANT-ZONE$ 
6 maxedZones  $\leftarrow |\sigma_{(UID=userID) \wedge (DISCLOSED=KH)} (\text{requestedOwners} \bowtie$ 
7    $OWNER \bowtie OWNS \bowtie PARCEL-USER) |$ 
8 if maxedZones > 0 then
9   return  $\emptyset$ 
10 potential  $\leftarrow \pi_{PID} (\sigma_{(UID=userID) \wedge (DISCLOSED=KH)} (\text{requestedOwners} \bowtie$ 
11    $OWNER \bowtie OWNS \bowtie PARCEL-USER))$ 
12 for all p  $\in$  potential do
13   IncrementYDisclosed(userID, BFS(p, z))
14 end
15 maxedRegions  $\leftarrow |\sigma_{(UID=userID) \wedge (YDISCLOSED > y)} OWNER-USER |$ 
16 if maxedRegions > 0 then
17   Rollback and return  $\emptyset$ 
18 Update(parcelID, userID)
19 return  $\pi_{NAME} (\sigma_{PID=parcelID} (OWNER \bowtie OWNS))$ 

```

5

Additional Aspects

In this chapter we present additional aspects of the cadastral application that affects QBA control. Indeed, we will show how to handle cadastral updates in Section 5.1. Cadastral updates are also called mutations and they include buying and selling of a parcel (transferring ownership from one legal entity to another), merging multiple neighboring parcels into a single one, and splitting a parcel to multiple ones. After performing an update, the topology of the graphs used in Pr_1 and Pr_2 change, which will affect all counters (e.g. k_l and k_h). QBA control should ensure equal rights of access for all users, which means that it should avoid—as much as possible—blocking users or allowing security compromises due to the change in the size of dominant zones. A detailed discussion on how to handle user access history in each prohibition, for every mutation operation, is presented.

Afterwards we will show how to gradually reset access to the database in Section 5.2. In fact, after a given period of time, and due to repetitive querying from a user, s/he could be blocked on a set of accessible parcels, which requires a resetting strategy to ensure that the database is useful for its users.

Additionally, we will discuss potential inference channels that may arise from QBA enforcement in Section 5.3. These channels are especially problematic for

Pr_2 , where a user can infer that s/he stumbled upon a parcel belonging to a target owner from a denial of access and background knowledge.

We will talk about authentication in Section 5.4. The security of QBA control relies heavily on how accurately we could relate the virtual identity of a user to his/her physical one. The circumvention of the collusion resistance scheme we are employing depends on how easily an attacker can create virtual identities in the system (which is known in the literature as “*The Sybil attack*” [Dou02]).

Finally, we will show how to select different parameters of our model in Section 5.5. As a matter of fact, our model relies on the graph; we will restrict the discussion to Pr_1 for reasons that will be clear in the next chapter, specifically in Section 6.3 on page 79. The topology of the neighborhood graph is largely affected by the geographic, economic, and historic (and other) factors, therefore, these parameters should be selected after an analysis of the database. We will use the cadastral database of Maupiti as an illustrating example to show how to calibrate the model’s parameters.

Maupiti is a small coral atoll located to the west of the Leeward Islands (*Îles Sous-le-vent; Fenua Raro Mata’i*) in French Polynesia, hosting around 1200 inhabitants. The cadastral database of Maupiti was provided by the real-estate service of French Polynesia.

In this chapter only we will use both terms “*zone*” and “*dominant zone*” interchangeably.

5.1 Handling Cadastral Updates

Four cadastral operations (called mutations) are performed daily on the database:

- Buy and Sell: a parcel's ownership is transferred from its original owner to a new person, affecting the topology of the graph in Pr_2 only;
- Merge and Split: two or more parcels are merged (split) into a single parcel (multiple parcels), affecting the topology of the graph in Pr_1 and Pr_2 .

In the following, we will show how to handle mutations for Pr_1 and Pr_2 (Sections [5.1.1](#) and [5.1.2](#) respectively).

5.1.1 Mutations in Pr_1

The first operation we want to address is a Buy/Sell of a parcel p . It is an operation that changes the proprietary of p , therefore all users should have the opportunity of accessing this parcel and knowing its new owner. Intuitively, the solution should be the erasure of all access history to guarantee equal access to all users. But let us take a look at the options we have:

1. Erase user access history of p , in this case:
 - (a) Users who have not queried p before a Buy/Sell will not be affected, whether they were blocked on any dominant zone containing p or not.
 - (b) Users who queried p before a Buy/Sell have now the right to choose to re-query p or another parcel in the same zone.
 - i) If they could have accessed other zones containing p before the Buy/Sell, then there is nothing to worry about, however
 - ii) If they were blocked (access denied) on any zone containing p , this erasure might give him access to information that was previously "*classified.*"

2. Keep user access history of p : nothing changes for any user, whether s/he queried p s/he has been blocked on a zone containing p (before the Buy/Sell).

Notice that erasing access history does not only raise a security issue (Point 1(b)ii above), but it is computationally costly too. If we were to erase the history, breadth-first traversal (BFS) should have been incurred multiple times per user in that list. Therefore the best strategy is to keep access history of a parcel that have been bought/sold.

Now we should examine merging and splitting. Merging requires all parcels involved in a merger to form a continuous geographical region: every parcel should touch at least one other parcel. Splitting does not have this requirement. However in both cases, zones that contained old parcels will change, particularly in size (bigger, smaller or keep their size; we should remind you that all users should have equal right of access to the new information after merger/split).

We can merge/split access history, but this is problematic when there are zones, post-merge and post-split, that get smaller in size: DISCLOSED and YDISCLOSED might exceed allowed limits (namely k_i and y , respectively), which requires special handling, per-user; in other words, not all users will have equal rights of access. Erasing access history, i.e. removing access history of merged and split parcels, is more convenient and does not induce that issue; therefore we argue for it. It is worth mentioning that merging/splitting access history and erasing it induce another security issue in one special case: when zones, post-merge and post-split, become bigger, users might gain access to information that was previously “classified.” See Point 1 in Section 3.3 on page 41 for more details.

Algorithm 4 shows how the update algorithm should work. It takes 2 lists: `oldParcels` and `newParcels`. For mergers, `newParcels` is a single item; for splits, `oldParcels` is a single item. We chose to write down a single algorithm for both for brevity. The algorithm works as follows:

- 2 – 3 Delete all traces of user access history in PARCEL-USER, ZONE-USER, ZONE and NEIGHBOR.
- 4 – 5 Add new parcels to NEIGHBOR.
- 9 – 12 Update ZONE with new dominant zones.
- 14 – 16 Recalculate counters, i.e. KH, KL, DISCLOSED and YDISCLOSED. This operation requires a Breadth-First traversal in order to correctly calculate YDISCLOSED.

Algorithm 4: Update Algorithm for Merge/Split (Pr_1)

```

Input: oldParcels, newParcels
1 neighbors = GetNeighborsOf(oldParcels)
2 DeleteFrom_PARCEL-USER_AND_ZONE-USER(oldParcels)
3 DeleteFrom_DOMINANT-ZONE_And_NEIGHBOR(oldParcels)
4 for all n ∈ newParcels do
5     InsertInto_NEIGHBOR(n, GetNeighborsOf(n))
6 end
7 affectedParcels = newParcels ∪ neighbors
8 affectedZones = ∅
9 for all p ∈ affectedParcels do
10    dz = GetListOfDominantZones(p)
11    affectedZones = affectedZones ∪ dz
12    Update_DOMINANT-ZONE(dz, p)
13 end
14 for all z ∈ affectedZones do
15    for all u ∈ UsersOf(z) do
16        RecalculateCounters(z,u)
17    end
18 end

```

5.1.2 Mutations in Pr_2

Mutation operations affect Pr_2 very differently. In fact buying, selling, merging and splitting are all equivalent. Ownership of a parcel will be transferred from one person to itself (i.e. in the case where the resulting owners of the merger/split are the same owners of old parcels) or to other owners, and in both cases, zones affected by these mutations will get bigger, smaller or keep their sizes. And in all of those cases, the best solution is to erase parcel access history, for the same reasons presented for merge/split for Pr_1 in Section 5.1.1.

5.2 Resetting Access

Another important problem of QBA enforcement is the fact that after a given period of time, when users consume k entries from a zone, they become blocked on those k entries (as depicted in Figure 8.1 on page 99) and the database itself becomes of no useful value in any future interaction¹. Therefore, an appropriate resetting should be done so that users can still use the cadastral application. For instance, if a user was blocked in a zone on k_i out of N_i parcels and 2 or 3 years later, s/he decides to come back and query some parcels in the same zone s/he will still be blocked although a long period of time has passed and this user has a legitimate need of the requested information.

By removing previously accessed parcels from the user's history (from PARCEL-USER), the user gains the ability to query other parcels in zones that would normally be blocked.

For this reason, we made sure that PARCEL-USER contains a TIMESTAMP attribute to record access date. The simplest resetting scheme would use a global timer that ticks every T units of time, and removes t parcel from the history of every user if they were accessed more than T units ago. On every timer tick, for every user $u \in \text{USER}$:

¹This observation was also found in inference control, [e.g. CW05]

1. TBR = a subset of PARCEL -USER where every entry belongs to u and $NOW() - \text{TIMESTAMP} > T$. This is the list of parcels To Be Removed.
 - (a) TBR is in ascending order (i.e. from oldest to newest), and
 - (b) We are only interested in the top t entries.
2. For every $p \in \text{TBR}$
 - (a) Remove p from PARCEL -USER
 - (b) DISCLOSED of every zone from ZONE -USER containing p is decremented.
 - (c) If DISCLOSED becomes less than y , then:
 - i. Perform a breadth-first traversal (BFS) rooted at p with maximal depth z , and
 - ii. Decrement YDISCLOSED in every region visited by BFS (in 2(c)i).

Note that we are proposing a gradual resetting scheme where, eventually, the possibility of accessing any parcel in the database can be obtained given that the user is rarely accessing the database (i.e. resetting all access to all parcels). The choice of the value of the threshold T is of utmost importance from a security perspective. Big values (e.g. 3 years) might put in question the utility of the resetting scheme, and small values (e.g. 1 hour) put in question the utility of the whole QBA control mechanism. The security administrator should take into account how frequently the cadastral database is accessed. Big values of T lead to more data confidentiality, while small values lead to greater data availability. The same argument, although reversed, goes for t , the number of parcels to be released on every timer tick: big values of t lead to more data availability, while small values lead to more confidentiality.

Other strategies could exist. For instance, there might be a need to penalize users who insist on querying parcels that are already blocked, but the penalty should be attributed during QBA control. That is, if a user tried to access a parcel, where access is denied for x -collusion resistance or (x, y, z) -collusion resistance violation then the release of all neighboring parcels could be postponed for another timer tick. This can be achieved by the following: if a user was denied access from a parcel for x -collusion resistance or (x, y, z) -collusion resistance violation, then for every parcel p neighboring the requested parcel, also belonging to PARCEL-USER:

1. if $p.TIMESTAMP < \text{next scheduled timer tick}$, then add T units of time,
2. else, do nothing.

5.3 Inference Channels

5.3.1 Potential Inference Channels from external knowledge

Most of the people using the cadastral application do have some external knowledge. Very often, they know the owner's name of some parcels from their neighborhood or their village or their family. Because of this external knowledge, users can break Pr_1 or Pr_2 without being detected by the inference control mechanisms. Dealing with external knowledge is theoretically impossible since it is simply impossible to know what a given user knows. However, the security administrator can roughly estimate the average level of users' external knowledge. This estimation is expressed in the parameter β of equations 5.1 and 5.2 which are modifications of equations 3.2 and 3.3.

$$k_l = \begin{cases} \lceil N_i/x \rceil - (\alpha + \beta) & \text{if } \lceil N_i/x \rceil > (\alpha + \beta) > 0 \\ 1 & \text{otherwise} \end{cases} \quad (5.1)$$

$$k_h = \begin{cases} N_i - (\alpha + \beta) & \text{if } N_i > (\alpha + \beta) > 0 \\ 1 & \text{otherwise} \end{cases} \quad (5.2)$$

$\beta = 0$ means the users are assumed to have no external knowledge whereas a $\beta > 0$ means the users are assumed to have some external knowledge.

5.3.2 Potential Inference Channels from Denial of Access in Pr_1

In the framework of multilevel database, Sandhu and Jajodia [SJ92] underlined the fact that a denial of access provides the user with the information that the data s/he is trying to access is highly classified. In the context of our application, if a user is denied an access then s/he can conclude that s/he is about to break prohibition Pr_1 . If the user is trying to break Pr_1 then s/he actually does not learn much from the denial of access. From the parcels s/he has accessed before the denial of access, s/he can simply verify that s/he has queried too many parcels within a given region.

5.3.3 Potential Inference Channels from Denial of Access in Pr_2

First of all, we should note that in order to be successful, an attacker trying to break Pr_2 should already know the approximate location of all the target entity's parcels. Without this external knowledge, the attacker would need to randomly select parcels from the entire database which is of course infeasible.

Nonetheless, we consider it as a probable attack and we shall address it. Let us assume that Bob already knows the approximate location of all Alice's parcels. We also assume that after several queries, Bob has identified several parcels belonging to Alice. If Bob is denied access to an additional parcel then he can reasonably deduce that this parcel belongs to Alice. Returning "*access denied*" can even be seen as worse than returning Alice's name since it informs Bob that he has found the last parcel in Alice's list of parcels, if we consider $\beta = 0$.

One possible solution to prevent Bob from deducing that he has found the last parcel in Alice's parcels list is to increase the value of β . In that case, Bob would be denied access to Alice's parcels before finding the last parcel. However, there is no

solution to prevent Bob from deducing from a denial of access that he has found a parcel belonging to Alice.

Another possible solution would be to return a cover story instead of denying access. A cover story is a lie introduced in the database in order to hide the existence of a sensitive data [CG99]. Cover stories have mainly been used in the framework of military multilevel databases [Den+88]. In our cadastral application, using a cover story would mean returning a fake owner for a given parcel. This solution is of course unacceptable for an official online public cadastral application where answers to query have a legal value and have, therefore, to be trusted.

We propose another solution: we deny access to the remaining parcel and all its geographical neighbors. In the same example of Alice and Bob, when Bob reaches the limit k_l , we deny access to the remaining parcel, namely p , and all parcels of its geographical zone (as defined for Pr_1). This can be achieved by adding a special flag associated to every parcel in the database and read during QBA control. When Bob reaches k_l in any dominant zone, QBA control should set this flag to true to the remaining parcel p and its geographical neighbors. Subsequently, when Bob tries to access p or any of its geographical neighbors, access should be immediately denied. This flag should be the first thing checked by QBA control.

This way, we increase the confusion for Bob, thus lowering his confidence in the inference by denial of access from $1/1$ (the case where only the remaining parcel is blocked) to $1/n$, where n is the number of parcels in the (Pr_2) zone of p . This confidence can even be lowered by increasing the number of blocked parcels by including 2^{nd} degree neighbors of p too.

5.4 Authentication

The security of QBA control depends on the ability of the system implementing it to directly relate real identity of users to their virtual ones, and restrict their ability to create multiple users in it. All philosophical debates on what constitutes a person's identity aside, we consider that the real identity of a user is his physical identity.

The first required step is “strong” authentication. O’Gorman [OG03] defines user authentication as «[...] *the process of verifying the validity of a claimed user.*» Users’ identity is verified, by a machine, using one of the following types of authenticators:

Knowledge-based – by proving “*what you know*” to the authentication service.

This includes PIN numbers, the name of the highschool you attended, correctly identifying names of friends on a picture². The most famous example of authenticators that are knowledge based are passwords. This type of authenticators rely on the secrecy or the obscurity of the information required for authentication. There are numerous issues with this type of authenticators (especially passwords [Sch08; Goo13]). Users share passwords, write them down, they are vulnerable to dictionary attacks, etc.

Object-based – by proving “*what you have*” physically, such as a security tokens [Mat03], which are pieces of hardware: bankcards, smartcards, one-time passcodes, etc. These devices can be used standalone or as mixed with other types of authentication.

ID-based – by proving “*who you are*”, this includes credit cards, passports, diplomas, etc. Biometrics [Way+05], such as fingerprints and voiceprints, fall into this category too.

²Facebook occasionally uses this type of authentication to verify that the logged-in user is actually who he claims to be, especially in cases where s/he logs in from a different country than the last time s/he did.

Usually, multiple types of authenticators are used in conjunction³, such as time-synchronous on-time passcodes with passwords, and two-factor authentication [Sch05].

«One can have, some claim, as many electronic personas as one has time and energy to create» [Don99]. Authentication is used to prove that the user is what he claims to be, but for QBA control, it is important to make sure that a physical user has one virtual user in the system—or at least minimize the number of virtual users s/he could create. Situations where users can forge multiple identities to circumvent the QBA control should be avoided, e.g. authentication by email and password. This is known in the literature as “The Sybil Attack” [Dou02]. Sybil attacks were discussed by Douceur in the context of peer-to-peer networks [DH06] where users can forge multiple nodes in the system, but they appear in a lot of application domains, such as ad-hoc networks, cash economies, reputation systems, etc. There is no general solution [LSM06] for such an attack, and every case should be considered separately. Among the solutions proposed in the literature, we find social networks [Yu11], trusted certifications, trusted devices, and others.

The choice of the authentication mechanism largely depends on the target audience: *could they afford biometric authenticators? Are smart-cards convenient for the desired user-experience? Can the database operator, legally speaking, hold its users' IDs? etc.* Therefore, the system developers should take into account multiple factors in order to achieve as much as possible the goal that we stated at the beginning of this section: *how to tie the user's virtual identity to its physical one?*

³Which means they need to be all validated. Any failure in any step means a failure in authentication.

5.5 Choosing the Model's Parameters

The responsibility of setting the values of the model's parameters falls on the security administrator. We have already discussed other parameters and their significance. α is used to anticipate variations in the size of dominant zones. β is used to minimize the effect of inferences arising from QBA control itself due to users' *a priori* knowledge. β figures in two equations: 5.1 and 5.2, for k_l and k_h respectively. It should hold the same value in both cases. Parameters of the resetting scheme t and T should be calibrated depending on the expected traffic.

The parameters x , y and z cannot be assigned arbitrary values. We will limit our discussion to Pr_1 for reasons we mentioned in the previous section. For instance, x defines the level of collusion resistance per dominant zone, therefore, for a given parcel p , x should always be strictly smaller than $|dom_p|$ ⁴. In Section 3.3 we argued that x should have the same value for all dominant zones.

Let us take the example of Figure 8.5 on page 104, and let us consider, for the sake of argument, that this is our complete cadastral database. We have two dominant zones: $zone_{p_2}$ of size 3, and $zone_{p_5}$ of size 5. If we want this database to be 4-collusion resistant, then for $zone_{p_2}$, $k_l = 1$ and $k_h = 2$; for $zone_{p_5}$, $k_l = 1$ and $k_h = 4$. Notice that this is the highest level (limit) of collusion resistance that could be attained on $zone_{p_2}$: we either give access to 1 parcel out of 3, or we deny access completely, which is not a desirable outcome for this specific application. Therefore, the best solution is to fix x for the whole database: for any parcel p where $|dom_p| \leq x$, $k_l = 1$, achieving $(|dom_p| - 1)$ -collusion resistance; the remaining dominant zones will be x -collusion resistant.

Figure 5.1 on page 72 shows the cumulative distribution of the sizes of dominant zones in the island of Maupiti. The value of x could be less than or equal to the average of dominant zone sizes in the database. It could even be set to

⁴If $x = |dom_p|$, then any user would have access to 1 parcel in dom_p at most. A coalition of $|dom_p|$ users can recover dom_p

the value of the mode ⁵ (or a value in between). If x was set to the average, then 66.41% of dominant zones will be pushed to their limits. If it was equal to the mode, then 32.82% of dominant zones will be pushed to their limits in terms of collusion resistance. Therefore, the security administrator should perform such an analysis for his cadastral database. If s/he sets x too high (e.g. 10 for the cadaster of Maupiti), then the majority of dominant zones will not be effectively x -collusion resistance as s/he desires (e.g. 91.28% for Maupiti).

Here, we can see that the adoption of dominant zones does not affect this decision, when compared to zones only. Indeed, the minimal size of a dominant zone or a zone in Figure 8.4a on page 103 is 2. The mode for both of them is 3. However the average for dominant zones is 5.94, while it is 5.55 for zones. This clearly shows that dominant zones did not significantly change the minimum, median or average size of “active zones”, even though it reduced the number of “active zones” significantly (around 60% reduction as mentioned in Section 8.1.2, page 102).

If we consider (x, y, z) -collusion resistance, then y is the number of parcels that are not under collusion resistance in a z -region, therefore for a given parcel p , $y < |z\text{-region}_p|$. Theoretically, y could be distinct for every z -region, e.g. a third of a z -region, or set uniformly, i.e. y is constant. There is no implication on the security of the application. However, distinctive y values means that it should be calculated for every dominant zone, which means Breadth-First traversal should be applied to every dominant zone, especially after a mutation operation, and the resulting value should be stored in the database alongside every dominant zone: performance and storage hits are inevitable. Therefore, setting a global value for y is a more reasonable choice.

⁵The mode is the value that appears most in the dataset; the zone size that appears most in the database, in our case.

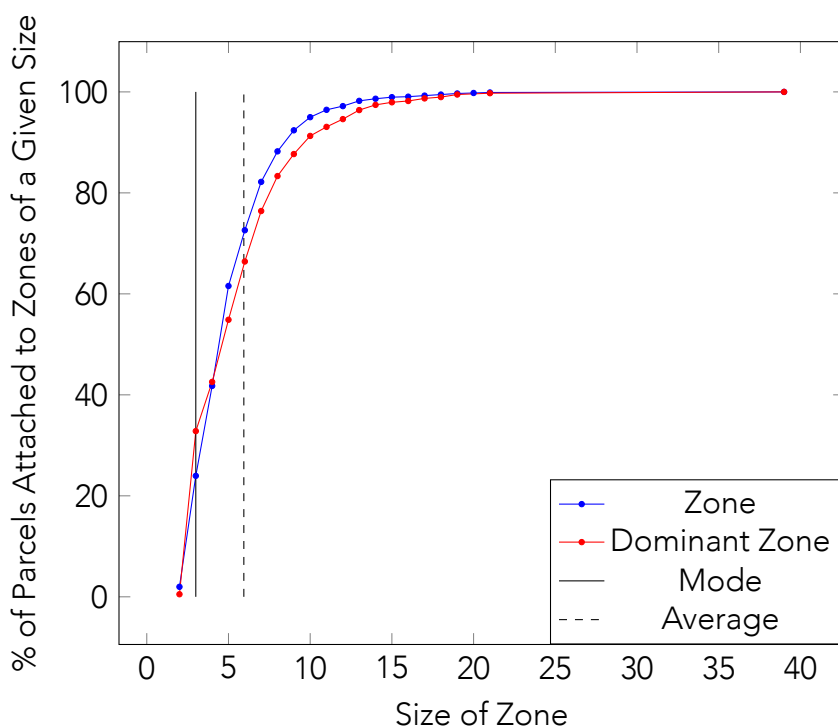


Figure 5.1: Cumulative distribution of parcels attached to zones of different sizes

Practically, γ can be lower than or equal to the average number of dominant zones (or mode) in a z -region. Here, dominant zones show an advantage. Figure 8.9 on page 113 shows that the average number of parcels in a z -region, for different values of z , change less drastically for dominant zones than for zones. This advantage is even clearer if we consider the maximum number of parcels that could occur in a z -region. Indeed, the slope of maximum parcels in a dominant zones is close to that of their average, and almost as smooth; the slope of maximum parcels in zones, on the other hand, is very steep and jumps drastically especially for low values of z .

As for z , the security administrator should keep in mind that, in addition to its function in determining the balance between data availability and its confidentiality, it determines the depth in the Breadth-First traversal, which has a runtime complexity of $O(b^z)$, where b is the branching factor (or the average number of neighbors per parcel). Therefore, z should be > 1 , and an assessment of available

computational resources should be taken into account to achieve the desired and most practical results. Here also, dominant zones present a significant advantage over zones, in terms of runtime, as shown in Figure 8.8 (page 112) and discussed in Section 8.2.2 (page 110).

Nevertheless, the security administrator and decision makers on this matter should test different values—while taking into account these recommendations—to see for themselves the results of different tunings and different combinations. In fact, the topology of the neighborhood graph changes from one cadastral database to another, and it is mainly related to the geography of the place in question. It could also be affected by economical or social factors. The graph of an ancient and continuously lived city like Byblos⁶ differs significantly from that of a modern one like New York City, or a remote island in the pacific like Maupiti. The topology will directly affect the availability of cadastral data, which requires human intervention and judgement to get the most desired results.

⁶The city of Jubayl in modern-day Lebanon, first occupied between 8800 and 7000 BC.

6

Application to the Cadaster of French Polynesia

Our work is applicable to any cadastral database that requires QBA control. This chapter discusses the application of QBA control to the case of French Polynesia.

We will first introduce the workflow desired by the real-estate service (Section 6.1) and its implication on authentication (Section 6.2) and the security of the application. In fact, the real-estate services wishes to replicate—as much as possible—the current physical workflow of cadastral excerpt demands, and make it available through the Internet. This requirement implies the abandonment of any “*strong*” authentication method; IP-based authentication is considered sufficient for them.

Afterwards we will show what prohibition should be implemented, and why (Section 6.3). *Should we implement Pr_1 only? Can we implement both Pr_1 and Pr_2 ?*

Finally we will briefly discuss other security aspects that should be taken into account for a production environment (Section 6.4).

6.1 Desired Workflow

Currently, in order to acquire information about any given parcel, a citizen of French Polynesia needs to visit the facilities of the real-estate service of French Polynesia. There, s/he will stand in a queue waiting for her/his turn, and then s/he will meet an employee who will receive the citizen's query. The citizen needs to provide the requested parcel's ID, or its address. Moreover, s/he can query multiple parcels at the same time. The citizen needs not to provide any identification (no driver's license, nor passport, etc).

Once provided with the parcel's ID or its address, the employee will perform a check on the query itself, the number of requested parcels and the rate at which the citizen has been issuing queries:

1. *Is the requested information classified?* (e.g. owned by the military, the president, etc.)
2. *Is the citizen requesting a lot of parcels?* (e.g. the owners of a complete neighborhood)
3. *Has the citizen been asking for cadastral excerpts regularly and in a suspicious manner?*

Obviously, the employee is enforcing an internal policy constraining citizens' requests. If the employee accepts the request, the citizen must pay a fee before getting the excerpt of the requested parcel(s).

There are two main issues with this workflow:

1. Citizens should be physically present at the real-estate service. This is especially problematic in countries such as French Polynesia that are formed uniquely by archipelagos (118 islands and atolls with an Exclusive Economic Zone (EEZ) of over 5 million km². In comparison, Metropolitan France's EEZ is around 330 thousand km² only).

2. Employees enforcing the service's internal policy are themselves human, therefore error-prone. Moreover, there is not a single employee, and they do change with time.

The real-estate service wishes to make the cadastral database available online, making it easier for citizens to acquire excerpts of parcels, while adapting the original workflow as follows:

1. A user is presented with a mapping interface where s/he has the option to select a single parcel.
2. Once selected, the user has the option to "*preview*" the parcel's ownership information, as long as this "*preview*" does not violate the service's policies (namely Pr_1 and Pr_2).
3. If the preview was successful, the user can either cancel his order or proceed and place the order for the excerpt where s/he is required to pay a predefined fee.
4. If the preview was not successful—due to the violation of either Pr_1 , Pr_2 , or both—the user can still proceed and place the order for the excerpt and pay the required fee.

This "*preview*" feature acts as a guard for the user her/himself: online data can be out of date or incorrect. Therefore, s/he can profit from this feature and withhold from paying any amount of money if s/he judges that online information is not accurate. Pr_1 and Pr_2 are required to limit the abuse of this feature.

6.2 Authentication

Our model is secure with “*strong*” authentication. However QBA control in the context of this cadastral application is only preventive as we previously showed.

The service explicitly mentioned that any form of “*strong*” authentication is unnecessary and might discourage users from using the service, especially that:

- Access to the internet on small islands is available uniquely through municipalities, and users are not necessarily tech-savvy.
- They want to replicate the current workflow found at their offices, and they want to keep no record that identifies the user explicitly, just like the physical process.

Users, for such workflows, can be authenticated with their IP addresses, which seems to be sufficient—from the service’s point of view—to enforce QBA and manage collusions. It follows that collusion resistance is also meant to prevent users from constantly changing their IP addresses (e.g. disconnecting their ADSL modem then reconnecting it) to circumvent QBA control. A tool to deter casual attackers, not serious ones.

The goal is not anonymity. Indeed, users of the cadastral application should be traceable on the online application using indirect identifiers (quasi-identifiers):

- The cadastral database holds information about people. Abusers of the application should be traceable in case tracing is needed (e.g. court order on legal action).
- The real-estate service does not have the right to ask for identification when a person asks for a cadastral excerpt—at their facilities or online—but they keep security cameras on at their offices, and employees and other people in the building can act as eye witnesses that could possibly re-identify a person if it is needed.

6.3 Pr_1 , Pr_2 or Both?

The security policy as defined by the real-estate service states that both Pr_1 and Pr_2 should be applied in conjunction.

Pr_1 is applicable directly to the whole database. Since French Polynesia is constituted of islands, the real-estate service has the advantage of analyzing and fine tuning QBA control for every island if it wishes to do so. Although tedious, the choice of parameters x , y and z (and α , t and T) for (x, y, z) -collusion resistance can be done independently for every island, taking into consideration the nature of every island¹, its economic and social importance², etc.

However, Pr_2 is problematic. We cannot know the number of parcels owned by multiple legal entities. In fact, parcels with multiple owners are registered as if they have a single owner. Ownership information in the database is not, currently, in a format that distinguishes and/or groups legal entities in a meaningful and consistent manner. For example, a married couple where each one owns a parcel outside marriage and share the ownership of a third will be identified in the database as 3 separate owners, with no links to tie them. This is not the case for the cadaster of France, for example, where every person is registered separately and relationships between people is present. If we take the same example of the married couple, in France, they would be identified as 2 separate owners—instead of 3—where everyone owns a parcel separately and they both share the ownership of a third.

Even if the real-estate service wishes to implement Pr_2 on the current database, the best level of collusion resistance that could be achieved is x -collusion resistance, because of the second reason we previously mentioned. Currently, there is no social graph in the cadastral database of French Polynesia, which is a prerequisite to (x, y, z) -collusion resistance.

¹*Is it an island? An islet? A reef islet?*

²Economic and social importance can be used as general indicators of expected traffic

6.4 Server-side Security

Many of the algorithms developed for the prototype, that we will introduce in the next chapter, are not directly portable for a production-ready environment, therefore, optimizations specific to the production environment are required.

The desired application is a web application, which means that a lot of attention should be given to server-side security. Li and Xue [LX14] give a comprehensive survey of server-side approaches to securing web applications, from which we mention:

1. Input validation, to prevent SQL injection and cross-site scripting (XSS) attacks.
2. Session management, to prevent session hijacking.
3. Static and dynamic analysis, to address different types of vulnerabilities like logic flaws, and access control and workflow violations.

Additionally, the final product should ensure that all exposed server-side APIs guarantee the “*point-and-click*” behavior of the application: every API call from the client should be fully processed before accepting and/or processing the next one. This can be achieved, for example, with stateful server-side sessions using a separate queue for every user.

7

The Prototype

In this chapter we will show how we implemented our prototype, which is a showcase of our algorithm: from graph generation to the enforcement of the security policy. Please bear in mind that our prototype is only a “*proof-of-concept*” for QBA control itself, not a prototype of the production application desired by the real-estate service.

Before we talk about different components of the prototype, we should mention that the computer science department has provided us with a database to test our model and algorithms (that of the island of Maupiti). However, this database was stored in a proprietary format (ESRI GDB, which uses `shapefiles` [98], the vector data format from ESRI). We converted this database to an open format, namely GeoJSON¹, which allows us to work with free and/or open-source tools easily and facilitates debugging.

We have implemented Pr_1 only for two main reasons:

1. All owners in the cadastral database of Maupiti own a single parcel.
2. Information about families is not currently available in a format that allows direct analysis regarding the advantages and disadvantages of dominant zones for Pr_2 .

¹<http://geojson.org/geojson-spec.html>

The prototype should be normally accessible from <http://webgis.upf.pf:8080>. The database we received from the GIS department of French Polynesia contains original information about Maupiti's parcel owners. Due to confidentiality agreements, we eliminated every trace of information that could relate to the original owners: all names were deleted and every parcel has one fake owner.

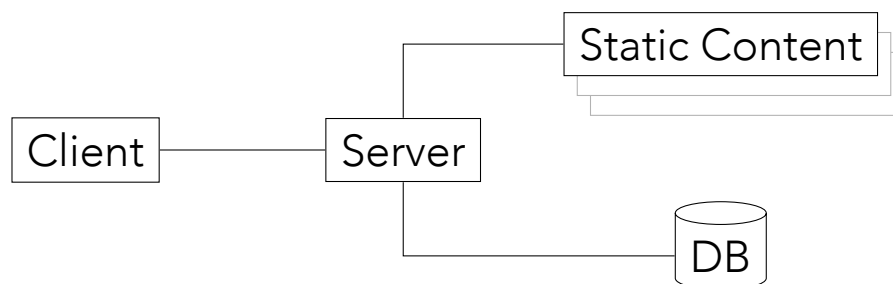


Figure 7.1: Prototype Overview

Figure 7.1 shows an overview of our prototype. We have 4 main components:

Static Content: `html` and `js` files to be consumed by the client. These files make up the visual interface and geographic data (in `GeoJSON`).

Database: Containing all information about parcels, owners, users, access history, etc. The enforcement of the security policy happens at this level (via `PL/SQL`). The enforcement algorithm decides whether the user is eligible to read the requested information, or not.

Server: A custom `HTTP` server delivering static content to the client and a web API used by the client to access ownership information from the database. The latter acts as a bridge between the client and the policy enforcement script in the database.

Client: A browser that loads all `html` and `js` from the server (with some additional APIs like `jquery`² and `leaflet`³) displaying a mapping interface for users. Users can click on a parcel to obtain its respective owner. This is

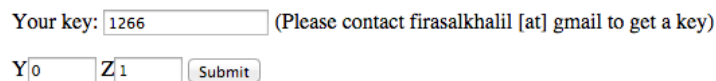
²A cross-browser library used mainly for asynchronous `HTTP` requests. <https://jquery.org>

³A library used to display parcels in a mapping interface. <http://leafletjs.com>

done through an asynchronous HTTP request to the server, that itself calls the enforcement algorithm stored in the database.

In the following sections, we will rapidly describe the user interface (Section 7.1). Afterwards, we will present a primer on R-Tree (Section 7.2.1) which is necessary to understand graph generation (Section 7.2.2; a hidden component and an imperative step required by the database and the client) then the database (Section 7.3), the server (Section 7.4), and finally the client (Section 7.5).

7.1 User Interface



Your key: (Please contact firasalkhalil [at] gmail to get a key)

Y Z

Figure 7.2: Initial Screen

The user is presented with a simple interface where s/he is required to enter her/his key and select the values for y and z (from our model), as shown in Figure 7.2. The key is there because the prototype is online and we want to prevent abuse by crawlers or amateur tinkers (in fact, the key is hardcoded in the source of the server; no sophisticated solutions were employed).

Once the form is submitted, the user is presented with two instances of our map previewer, simulating two distinct users as shown in Figure 7.3.

If we zoom in a little bit we can see parcels with more details. Parcels are colored in blue and their borders are marked with a white dashed line, as shown in Figure 7.4.

The user can choose, from the upper right corner, to display the neighborhood graph, as shown in Figure 7.5. We will call this graph the GeoJSON graph to distinguish it from the database graph. It is only used for display. It is very important for debugging.



Figure 7.3: Simulation of 2 Users



Figure 7.4: Part of the Cadastral Database of Maupiti

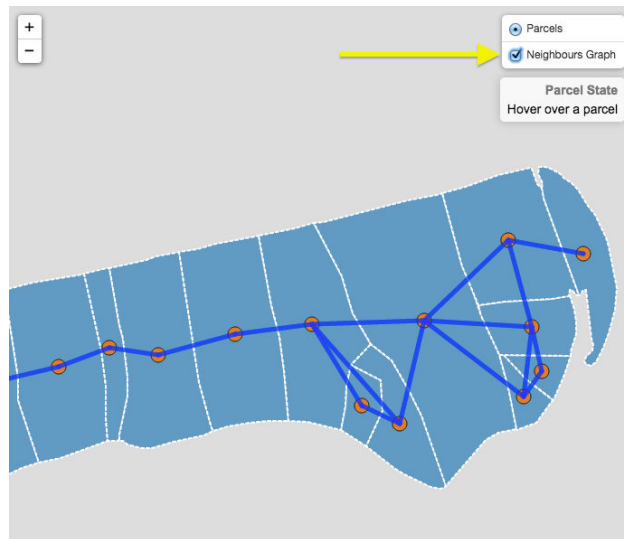


Figure 7.5: Part of the Cadastral Database of Maupiti with the GeoJSON Graph Plotted on Top

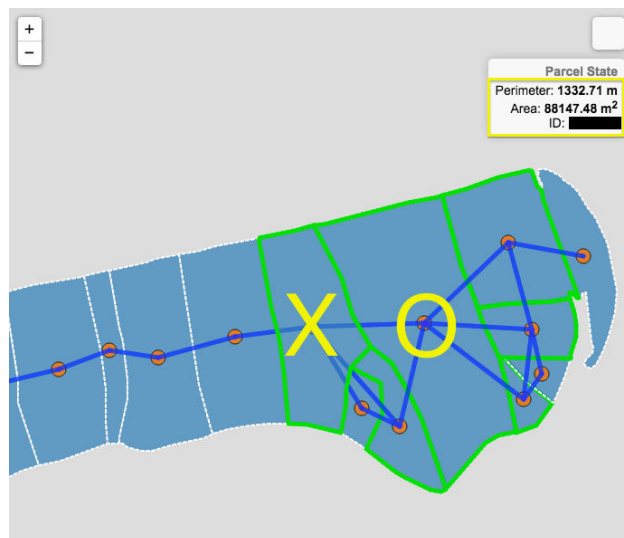


Figure 7.6: Part of the Cadastral Database of Maupiti Showing Information When Hovering over a Parcel

When a user hovers over a parcel, the dominant zone is highlighted in green, as shown in Figure 7.6. In this example, the user had the mouse over the parcel marked with X; its dominant zone is marked with O. Additional information about the parcel are shown in the upper right box (e.g. parcel ID, surface area). This information is also useful for debugging.

Of course, the GeoJSON graph and the dominant zone should not be displayed for the end user, but we remind you that this is a prototype made as a proof-of-concept and a tool to help explain the model and the algorithm.

7.2 Graph Generation

7.2.1 A Primer on R-tree

An R-tree [Gut84] is a data structure used to index multidimensional objects. For simplicity, we will restrict our explanation to 2D objects, i.e. polygons. R-tree is a balanced search tree. All leaf nodes are at the same height.

Before we continue our discussion on R-trees, we need to introduce *Minimum Bounding Rectangles*, MBR hereafter (also known as *bounding box* or *envelope*). A MBR is the smallest rectangle containing one or multiple polygons. It serves as an approximation of the position of a geometric feature. The details of MBR are out of the scope of this document, but the reader may refer to [Cal12] and [PT97] for more details on the subject.

To see how R-trees work, we will take the example of Figure 7.7 where we index simple polygons: rectangles. Rectangles R_8 to R_{19} (marked in red) are the ones that we wish to index. The key idea of the data structure is to group nearby objects and represent them with their MBR in the next higher level of the tree; the “R” in R-tree is for rectangle. Effectively, R_3 is the MBR of $\{R_8, R_9, R_{10}\}$, R_4 is the MBR of $\{R_{11}, R_{12}\}$, etc.

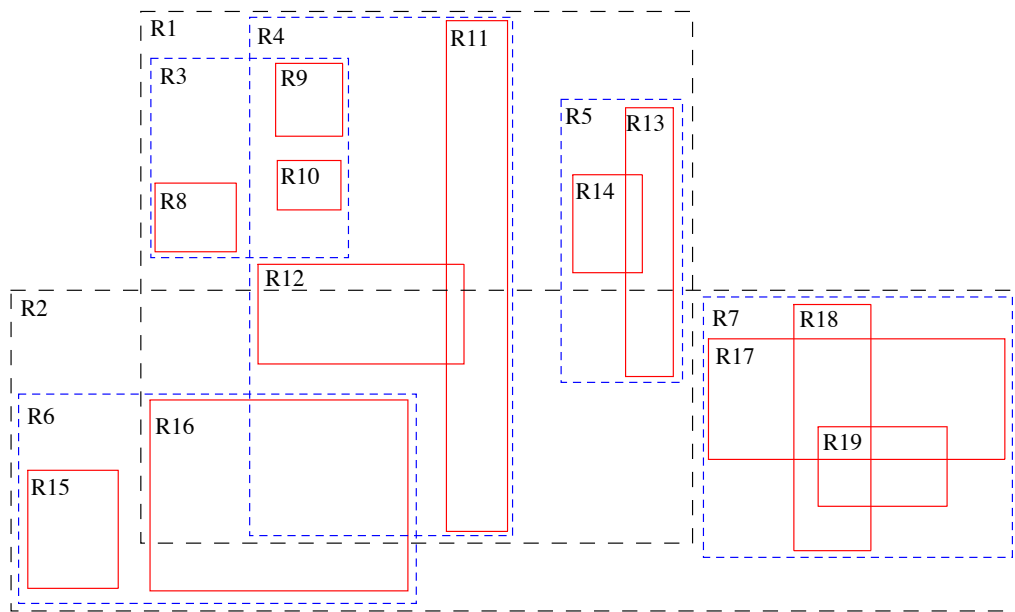


Figure 7.7: Example of R-tree for 2D Rectangles: Rectangle Visualization [Com10]

At the leaf level, each rectangle describes a single object; at higher levels (nodes), each rectangle describes the aggregation of an increasing number of objects. Figure 7.8 shows the tree structure of our R-tree index: our objects (R_8 to R_{19}) are stored at the leaf level, and MBRs occupy nodes (including the root).

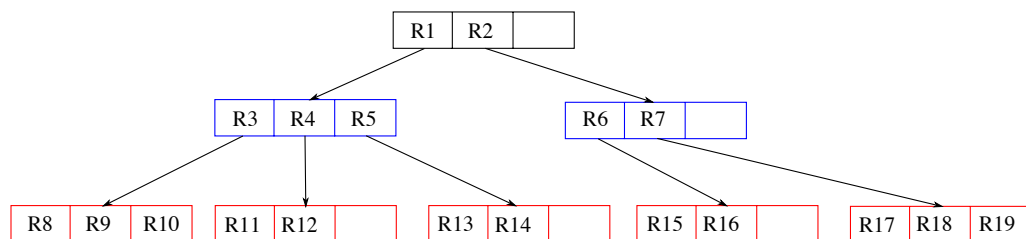


Figure 7.8: Example of R-tree for 2D Rectangles: Data Structure Visualization [Com10]

Since all objects lie within a bounding rectangle, a query that does not intersect the bounding rectangle also cannot intersect any of the contained objects. This property is important for us when we want to build our neighborhood graph. In fact, the first thing we should do—as we will see in the next section—is to create MBRs for all parcels. These MBRs are indexed in an R-tree (leaves). Then to detect all

neighbors of a given parcel p we simply have to get all MBRs (nodes) that intersect with it, discarding MBRs that don't, thus getting an initial approximation of touching parcels. To get the exact set of parcels that touch p , we use the euclidean distance function between p and all parcels in the approximation set (the euclidean distance is used on polygons themselves, not their MBRs). Therefore, R-trees allow us to avoid testing for intersection between p and every single other parcels in the database, especially if we consider that we need to repeat this operation to every single parcel.

To detect the neighbors that are separated by a distance τ , we should index bigger MBRs for polygons before generating the approximation set: For every polygon p

1. Create its MBR, mbr .
2. Calculate the desired new width ⁴ $w = width_{mbr} + (\tau/2)$.
3. Scale mbr with the factor ⁵ $f = w/width_{mbr} = 1 + \tau/(2 \times width_{mbr})$.
4. Index mbr .

Then we can proceed as normal from the approximation set and detect polygons separated by an euclidean distance τ . Notice that if $\tau = 0$, i.e. we want to detect only touching parcels, then $f = 1$, i.e. no scaling is need. This method of graph generation may not be optimal, however it is not of central interest or impact on our research on QBA, therefore it falls out of the scope of the thesis. Future research could address this specific issue by comparing this method to alternative ones—if there are any—in terms of performance, storage, and accuracy.

⁴Since every polygon will get a bigger MBR, it suffices to extend the reach of each mbr with half the value of τ .

⁵The scaling factor f could be calculated with the height, too.

7.2.2 Generation Method

The first step in the implementation was to convert ShapeFiles stored in the GDB to GeoJSON. Afterwards, we need to generate:

1. A GeoJSON graph for display, as shown in Figure 7.5. In fact, every node is the centroid of the parcel, represented by a GeoJSON point, and edges are LineStrings. This graph is for display only, but helps in debugging.
2. A graph for the database. It can be stored in multiple ways: edge list, adjacency list, adjacency matrix, etc. We use edge lists for in-memory representation, to create the graph (because it is more efficient in terms of storage). Then we use the edge list to generate an adjacency list for storage inside the database (for efficiency in the relational model). This dichotomy in graph representation is due to the fact that we had an early implementation of the model that did not use a relational database.

An edge list is a simple list of all edges in a graph. Let us take the example of Figure 3.1. The edge list of the graph is

```
1,2   1,3   1,8   2,3   3,5   3,7
3,8   4,5   5,6   5,7   6,7   7,8
```

An adjacency list is a graph representation where each vertex in the graph is associated with a list. Each list describes the set of neighbors of its vertex. The adjacency list of the graph in Figure 3.1 is

```
1->[2,3,8]   3->[1,2,5,7,8]   5->[3,4,6,7]   7->[3,5,6,8]
2->[1,3]     4->[5]           6->[5,7]      8->[1,3,7]
```

Indeed, we generate both graphs in a single pass. The logic of the algorithm can be simplified by the following:

1. Read the `GeoJSON` file in memory.
2. Generate two maps: `polygons` and `polygonsMBR` mapping `[parcelid]` -> `[polygon]` and `[parcelid]` -> `[mbr]`, respectively, from the loaded file. These maps are used later to detect neighboring parcels.
3. Put `polygonsMBR` in a spatial index `rtree`.
4. Create a `GraphBuilder` instance. This is a custom implementation of edge lists.
5. Create two empty sets: `vertices` and `edges` that will hold points and lines, respectively, and used later to generate the `GeoJSON` graph used for display.
6. For every polygon (parcel) `p`
 - (a) Add the centroid of `p` to `vertices`
 - (b) Apply `rtree.intersects(...)` on `p` to get the set of intersecting polygons and the set of lines connecting the centroid of `p` and the centroids of every neighboring parcel.
 - (c) Add `p` and its neighbors to `GraphBuilder`.
 - (d) Add computed lines to `edges`
7. Print `vertices` and `edges` in `GeoJSON` format.
8. Print the edge list from `GraphBuilder`

Once the edge list is generated, we save it in a `csv` file, then we run a script that generates `SQL INSERT` statements (to be stored in the `NEIGHBOR` relation as an adjacency list).

7.3 The Database

We used PostgreSQL 9 for the database. The enforcement algorithm is written using PostgreSQL PL/SQL and it is a translation of Algorithm 2, with some modifications to include additional features, such as debugging information. Listing C.1 of Appendix C shows the actual code.

In this section, We will only mention how Breadth-First Search, the function `bfs`, is implemented. Actually we have a function that builds a SQL query that calls `bfs` multiple times: on all dominant zones of the requested `parcelid` and its neighbors where DISCLOSED is equal to KL (because YDISCLOSED might possibly surpass y). The `bfs` function—stripped from most of Postgres specific details—is shown in Listing 7.1. We use SQL CTE (Common Table Expressions).

Listing 7.1: BFS with SQL CTE (Common Table Expressions)

```
1 WITH RECURSIVE search_graph(root, depth) AS (  
2     SELECT parcelid, 0  
3     UNION  
4     SELECT nxt.pidn, sg.depth + 1  
5     FROM   neighbor nxt, search_graph sg  
6     WHERE  sg.root   = nxt.PID  
7     AND    sg.depth <= z  
8 )  
9 SELECT DISTINCT s.root, p.kl, p.kh  
10 FROM   search_graph s  
11 JOIN   parcel p ON p.pid = s.root;
```

7.4 The Server

The server is written entirely in Java 1.7 . It serves two main functions:

1. Serve the UI for the user, through static content (`html` and `js` files).
2. Serve the capacity to query the database, through HTTP APIs.

Normally, the server would be developed for an *Application Server* and coupled with an independent *Web Server* (mainly because each server targets a specific functionality; separation of concerns). Moreover, in a geographic application, like the cadaster, a GIS is almost inevitable. However, we decided to build our own server that serves both HTML content and business logic, for multiple reasons: more control over variables and reducing unnecessary complexity to name a few. We are developing a proof of concept of a model that should be as independent as possible from underlying technologies.

We use the Spark framework ⁶ in our prototype to deliver the following URLs:

Files :

`/index.html` The welcome page, that allows the user to select values for y and z for (x, y, z) -collusion resistance.

`/main.html` Presents a single mapping interface that plots parcels and the GeoJSON graph.

`/maupiti.js` Contains GeoJSON polygons representing parcels and GeoJSON points and lines representing the GeoJSON graph.

⁶<http://www.sparkjava.com/>

APIs : All API calls require a parameter `callback` because we are using JSONP (JSON with padding); this allows the client to call this API when the latter is served on `localhost`. Usually, JSONP is used in scenarios where clients request data from servers on different domains. Therefore, in a production environment, JSONP should be disabled if both the client and the HTTP API are served on the same domain. Additionally, all calls require a `key`—for reasons we detail in Section 7.1 on page 83.

/getuser Creates a number of users in the database and returns their identifiers in JSON. This call requires the following parameters:

- c** The number of users that should be created and returned.
- db** The database that should be accessed. This parameter is only useful for debugging, when we needed to access different databases and test different algorithms for research.

/owner Called when a user clicks on a parcel to read its owner. This call invokes the DB's QBA enforcement script. This call returns one of two responses (as JSON objects):

1. Access denied, with a reason. `reason` could be equal to "Access Denied" if the user has reached KH (i.e. when the database returns an empty result-set), or a list of parcel IDs that represent dominant zones where the user might exceed `y`.
2. Access granted: the list of the requested parcel's owners is returned.

u User ID; one of the IDs returned by `/getowner`.

y The selected value of `y`.

z The selected value of `z`.

p ID of the requested parcel.

db Same as `/getowner`.

7.5 The Client

The client, as it is clear by now, is a web application. We did our best to make it compatible with all modern browsers supporting HTML5. Figure 7.9 shows the main components of the web application (`/index.html`).

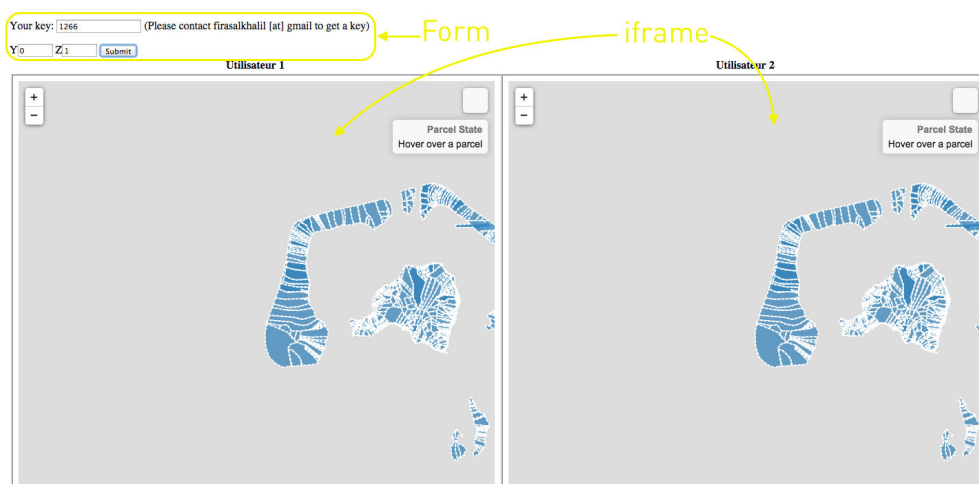


Figure 7.9: `index.html`

The application is made of a form that accepts the key, and values for y and z . Once the user submits the form:

1. An asynchronous call to `/getuser` is performed. In this case, we are asking to simulate 2 users. If the response from `/getuser` is not correct (e.g. invalid key) nothing happens.
2. If `/getuser` returned 2 users, then a table of 2 rows and 2 columns appears. Every cell (except the headers) contains an `iframe` pointing to `/main.html`. Every `iframe` uses one of the returned users' ID.

`/main.html` renders a mapping interface with data from `maupiti.js`. The actual mapping application is a `leaflet` library. Now the user can query parcels by simply clicking on a parcel.

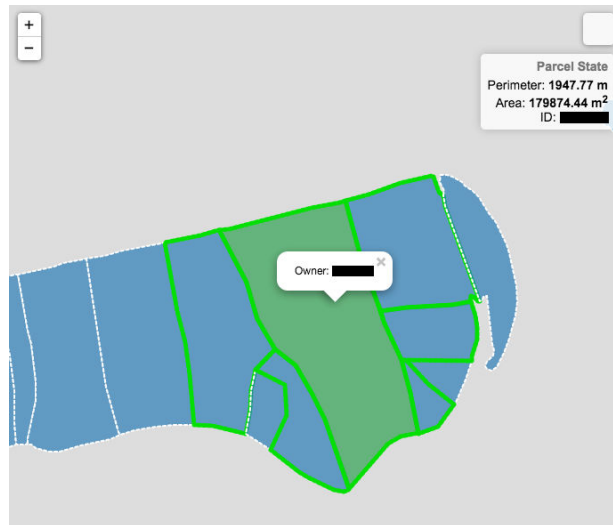


Figure 7.10: Access Granted

In case access was granted, a popup above the parcel containing ownership information is showed, and the parcel turns green (Figure 7.10).

If access was denied because the (simulated) user reached the limit KH in a given dominant zone, a popup saying “*Access Denied*” is shown above the requested parcel. Its color turns to red (Figure 7.11).



Figure 7.11: Access Denied: the Limit KH is Reached

If access was denied because the disclosure of ownership information of the requested parcel might cause one or multiple regions to surpass the allowed value of y , then a message saying “*Access Denied: ERROR Reached y*” is shown in a popup

above the selected parcel. Its color turns into red too. Moreover, borders of dominant zones where the value of y might surpass the limit turn into deep red too (Figure 7.12). The list of parcels is formed in the database during QBA control and transmitted to the client through `/owner`.

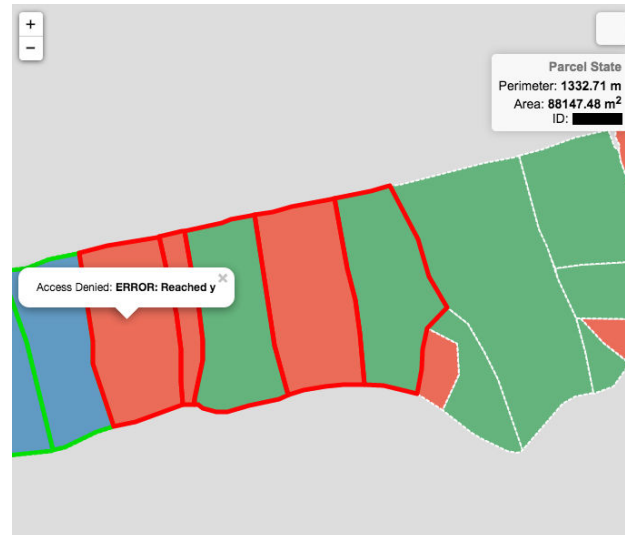


Figure 7.12: Access Denied: the Limit y is Reached and Borders of Dominant Zones that Risk Surpassing y are Highlighted

8

Benchmarks

In this chapter we present different benchmarks that show how dominant zones can achieve higher availability (Section 8.1) when compared to zones, and even outperform them (Section 8.2).

In fact, we tackled QBA problems in the cadastral database in an earlier work [AGC13]. Initially, we did not use dominant zones. QBA control was enforced on zones uniquely. We also defined different levels of collusion resistance (x -, (x, y) - and (x, y, z) -collusion resistance) to prevent users from colluding and bypassing Pr_1 and/or Pr_2 . Our approach to implement these levels required recording and tracking the query history of every user. This history was used to track collusions on the user level, i.e. maintaining lists of who is colluding with whom. This tracking required $O(\binom{U}{x})$ space to maintain the list of colluding users, while searching for a potential collusion on a single parcel level was an exhaustive search requiring $O(x^n)$ time, where U is the number of users in the system, n is the number of users who has accessed a parcel, and x is the value from x -, (x, y) - or (x, y, z) -collusion resistance. In addition, this implementation was described in terms of a graph database.

The work presented in [AGC14a] provided an alternative and more efficient implementation, using the same model, namely with zones only. In this second

implementation, we were not tracking any collusion in the first place. Indeed, we were defining the number of accessible parcels in a region (Pr_1) or belonging to a given family (Pr_2) beforehand and then simply counting the number of actually accessed parcels and making sure it does not exceed a given threshold. We also dropped a level of collusion resistance, namely (x, y) -collusion resistance, and changed some definitions in order to gain performance enhancements without compromising their security properties. Moreover, our solution was described in the relational model, facilitating the integration with the existing cadastral database of French Polynesia.

Dominant zones were introduced later [AGC14b] to achieve more availability while preserving the same level of confidentiality. Our goal is to ensure data availability for authorized users only, however, we want to allow them to get the possibility to know more data, using the new model, without violating the security policy.

To benchmark our QBA control methods, we used the cadastral database of the island of Maupiti. It contains 960 parcels.

8.1 Availability Benchmarks

8.1.1 Availability without Dominant Zones

QBA control relies on its collusion resistance scheme. The choice of x , y and z defines the number of available parcels per user. Figure 8.1 show the average number of parcels available for a user, when using zones only, for different values of y and z (x is set to 2). In fact, every point in this figure is the average accessible number of parcels per user: for every value of y and z we created 100 users and made them randomly traverse the complete database. As expected, increasing the value of y renders the data more available, while increasing z assures more confidentiality for cadastral data.

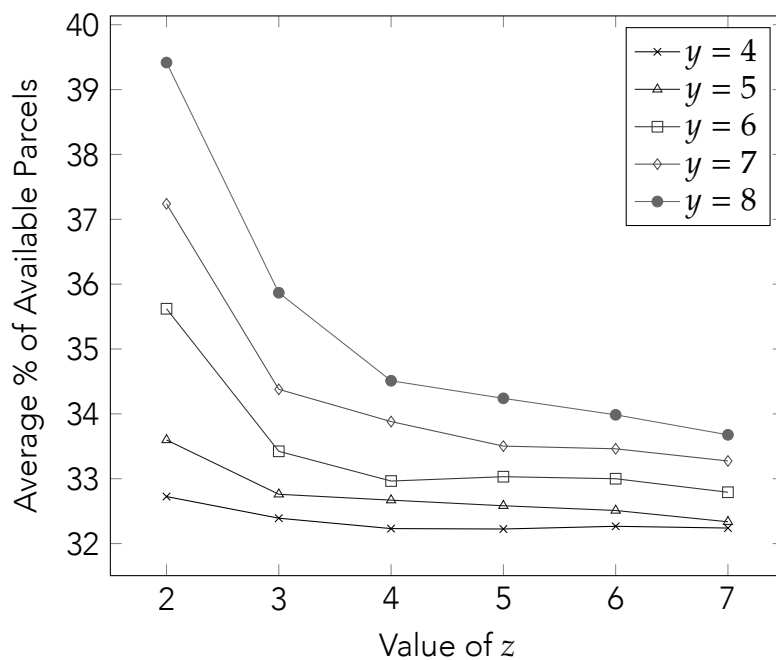


Figure 8.1: Availability for Different Values of y and z for Zones

Figure 8.1 shows that, even with relaxed security settings (high y and low z), the number of available parcels is very low. In order to achieve higher availability, the simplest solution would be to change the definition of a zone to reach 2nd

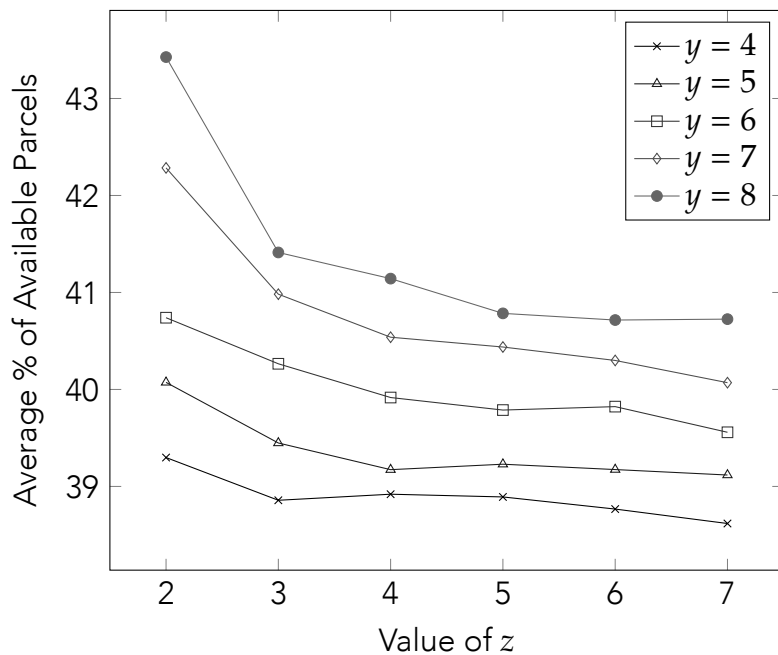


Figure 8.2: Availability for Different Values of y and z for 2-zones

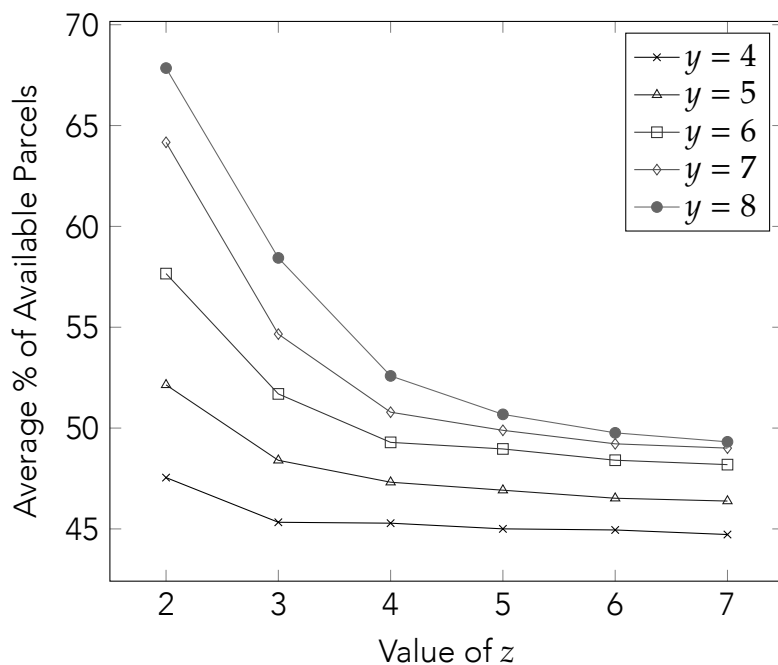


Figure 8.3: Availability for Different Values of y and z for Dominant Zones

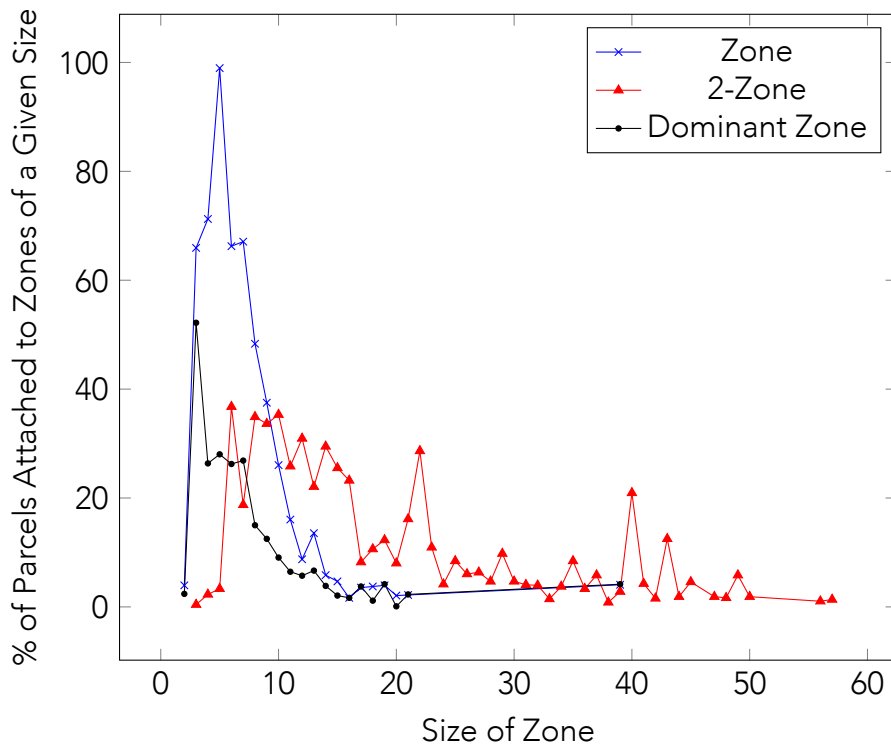
degree neighbors. We will call this new definition a 2-zone¹. This solution provides more availability as shown in Figure 8.2, however it presents a major drawback in processing time and storage. Indeed, in Maupiti's database, the relation storing parcel neighbors for 2-zones increased to 306.86% when compared to its size with normal zones of Definition 4. Another major drawback is the fact that the range of available parcels shrinks with 2-zones; we mean by availability range the difference between the lowest amount of available parcels to the greatest amount. In Figures 8.1, 8.2, and 8.3, the availability's range is 7.18% for zones (32.23% to 39.41%), while 2-zones reduced the range to 4.81% (38.62% to 43.43%); on the other hand, dominant zones increased availability's range to 23.13% (44.72% to 67.85%). This range of availability gives the security administrator more control and flexibility over the tradeoff between availability and confidentiality.

¹Recall that, according to Definition 7, a zone is a 1-region, and a 2-zone is effectively a 2-region. We decided to use the term 2-zone instead of 2-region to avoid confusion when thinking about x -collusion resistance that deals with zones solely, and (x, y, z) -collusion resistance that deals with zones and z -regions

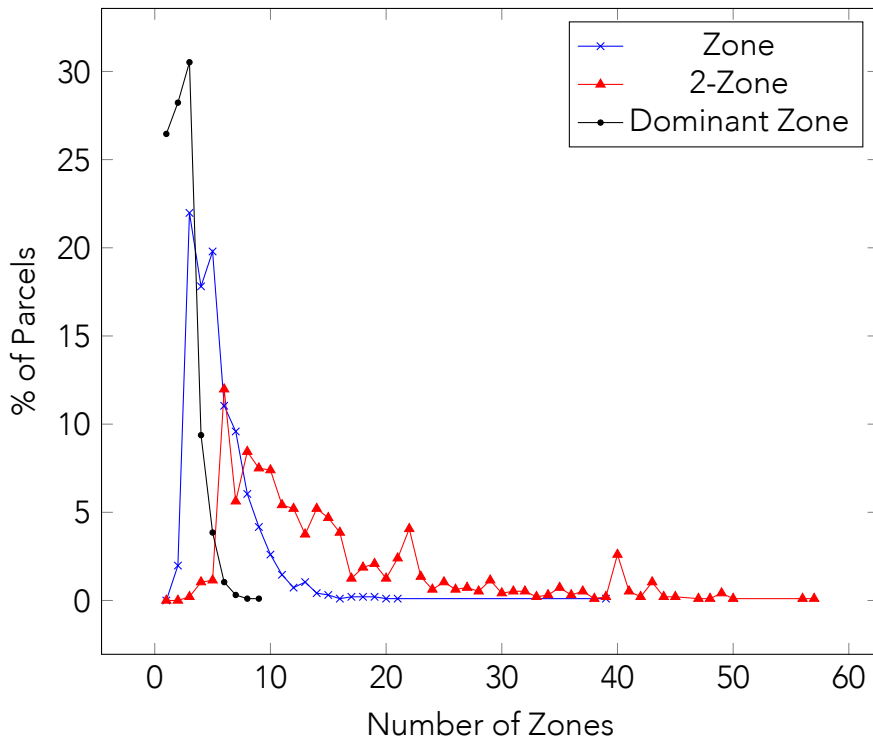
8.1.2 Availability with Dominant Zones

Let us take the example of Figure 8.5: we have 6 parcels $\{P_1, P_2, P_3, P_4, P_5, P_6\}$. According to Definition 4, the zone of P_1 , $zone_{P_1}$ is formed by P_1 and P_5 . Similarly, $zone_{P_2} = \{P_2, P_5, P_6\}$, $zone_{P_3} = \{P_3, P_5\}$, $zone_{P_4} = \{P_4, P_5\}$, $zone_{P_5} = \{P_1, P_2, P_3, P_4, P_5\}$, and $zone_{P_6} = \{P_2, P_6\}$. We suppose that we are not trying to achieve any level of collusion resistance. Every user has the right to access, in every zone, all parcels except 1 (see equation 5.1; $x = 1, \alpha + \beta = 1$). We consider a user, Alice, who has never queried any parcel. If Alice decides to access parcel P_5 , then access would be granted. However, access to $\{P_1, P_4, P_3\}$ will be automatically blocked because k_i is reached for all $zone_i, i = \{1, 4, 5\}$. Alice can finally query either P_6 or P_2 , thus acquiring the knowledge of 2 parcels out of 6. However, Alice could have queried these parcels in a different order: P_1, P_2, P_3, P_4 then P_6 , thus acquiring the knowledge of the owners of 5 out of 6 parcels. Notice that querying behavior changed drastically the number of accessible parcels; QBA control went from *very restrictive* to *very permissive*. Even if we try to apply x - or (x, y, z) -collusion resistance, the problem persists: these levels of collusion resistance change the quantity of accessible parcels, and actually render the QBA control enforcement stricter. This issue comes from the fact that we give all zones equal importance and we enforce collusion resistance on every single zone.

Figure 8.4a shows the distribution of parcels attached to zones of different sizes in Maupiti's database: every point represents the percentage of parcels in the database (ordinate) that are attached to a zone (resp. 2-zone and dominant zone) of a given size (abscissa). Let us take the example of Figure 8.5 to explain what we mean by attachment: if we are using zones, then P_2 is attached to 1 zone of size 5 ($zone_{P_5}$), 1 zone of size 3 ($zone_{P_2}$) and 1 zone of size 2 ($zone_{P_6}$); if we are considering 2-zones, then P_2 is attached to 3 2-zones of size 5 (2- $zone_{P_1}$, 2- $zone_{P_3}$ and 2- $zone_{P_4}$), 2 2-zones of size 6 (2- $zone_{P_2}$ and 2- $zone_{P_5}$), 1 2-zone of size 3 (2- $zone_{P_6}$); if we are considering dominant zones, then P_2 is attached to 1



(a) Distribution of Parcels Attached to zones of Different Sizes



(b) Distribution Showing How Many Parcels are Attached to How Many Zones

Figure 8.4: Comparison of zones, 2-zones and dominant zones

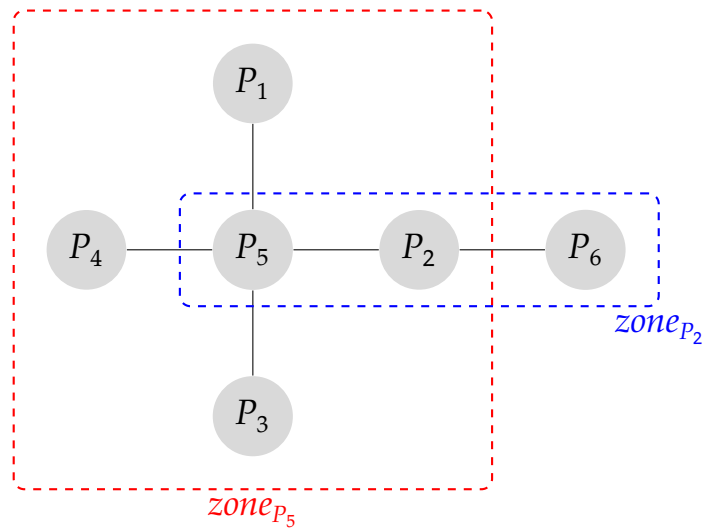


Figure 8.5: Example Graph Representing Parcels

dominant zone of size 5 ($dom_{P_2} = zone_{P_5}$). Notice that dominant zones reduce the number of attached parcels drastically: for instance, while 67.1%, 71.2%, and 98.9% of parcels are attached to zones of sizes 3, 4 and 5 respectively, dominant zones reduces these percentages to 52.2%, 26.4% and 28% respectively. On the other hand, 2-zones reduces these percentages but the number of 2-zones of different sizes is higher and 20% to 40% of parcels are attached to 12 2-zones of different sizes (bigger number of peaks in Figure 8.4a). This distribution explains why dominant zones provide more availability than zones and 2-zones: by giving priority on every parcel to the largest zone it is attached to, they reduce the number of parcels attached to zones of small sizes while keeping the sizes of (dominant) zones intact.

While Figure 8.4a shows how many parcels are attached to zones of different size, Figure 8.4b shows how many parcels are attached to how many zones, i.e. how many parcels are attached to 1 zone, 2, 3, ... (resp. 2-zones, dominant zones). Let us take the example of Figure 8.5: if we are using zones, then P_2 is attached 3 zones, namely $zone_{P_2}$, $zone_{P_5}$ and $zone_{P_6}$; if we're using 2-zones, then P_2 is attached to all 2-zones of the graph; if we're using dominant zones, then P_2 is attached to 2 dominant zones, namely $zone_{P_5}$ and $zone_{P_2}$ (because both contain P_2 and they

are dominant zones thus considered by the QBA control enforcement algorithm). Notice that, for dominant zones,

1. 85.2% of parcels are attached to 1, 2 and 3 dominant zones respectively, and
2. All parcels are attached to 1 to 9 dominant zones only, unlike zones (and 2-zones) that can be attached to 1 to 39 zones and 1 to 57 2-zones.

This distribution explains the fact that dominant zones provide a bigger range of availability: they reduce the number of zones that could influence the disclosure decision on a parcel, which, when combined with the fact that they reduce parcel attachment to zones of small sizes (Figure 8.4a), allows for a greater margin of flexibility when applying (x, y, z) -collusion resistance in QBA control.

The introduction of dominant zones has multiple advantages:

1. It produces less “*active zones*” thus improving execution time of the QBA control enforcement algorithm (390 dominant zones *vs* 960 zones for the island of Maupiti; around 60% reduction).
2. It provides more availability by giving importance to zones of big sizes.
3. It lowers the effect of the user’s querying behavior on the range of available parcels.

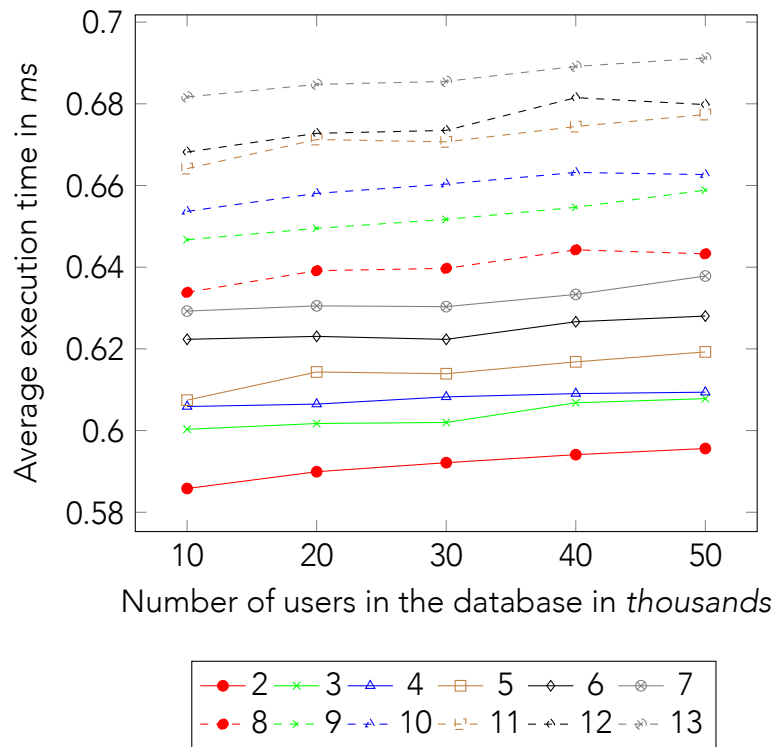
8.2 Performance Benchmarks

8.2.1 Performance of QBA Control with Zones

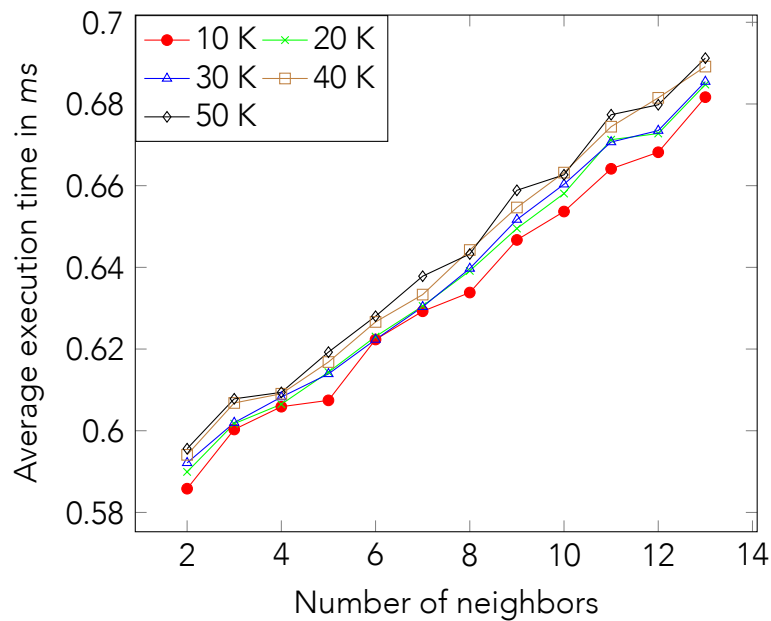
We are interested in the performance of our methods with respect to the number of users and the parameter z of (x, y, z) -collusion resistance only. Variation in the size of the PARCEL relation itself is not of a big interest since this table does not witness big variations in size over time, unlike the USER table that can go from hundreds to thousands of users. We considered zones only.

The benchmark was performed on a MacBook Air with: OS X 10.9, 1.8 GHz Intel Core i5, 4GB RAM, 120 GB SSD. We implemented 2 versions of the QBA enforcement algorithm (for x - and (x, y, z) -collusion resistance) in PostgreSQL 9.3. All QBA control algorithms were implemented with Postgres's PL/SQL. We created a script that sets up the database before any benchmark is run: it generates a given number of users (e.g. 1000) and simulates access to the cadastral database. In fact, we made sure that every user we register has accessed at least to one parcel per zone (parcels are selected at random). As such, every user has 960 entries in the ZONE-USER relation. We opted for this preliminary setup to simulate real access to the database, since access to 1 parcel affects the DISCLOSED value of all zones containing that parcel.

Afterwards, we selected a set of parcels to run our benchmarks on: every parcel is unique in terms of numbers of neighbors, i.e. we have 1 parcel with 2 neighbors, 1 with 3 neighbors, etc. so we have a total of 12 parcels. We hand-selected those parcels because running BFS on them will return a bigger set of results every time we increase the maximal depth z . In other words, these parcels allow us to test reliably the effect of increasing z for the (x, y, z) -collusion resistant algorithm without introducing outliers. Before we get to the final benchmark, we selected 100 users at random. We made sure that these users never accessed any of the



(a) Execution Time of x -collusion resistance as a Function of the Number of Users in the Database for Different Numbers of Neighbors per Parcel



(b) Execution Time of x -collusion resistance as a Function of Number of Neighbors per Parcel for Different Numbers of Users

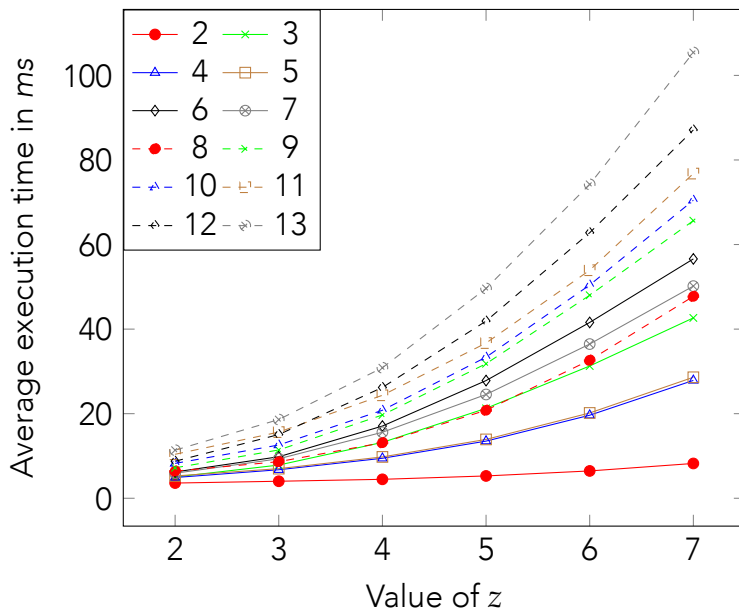
Figure 8.6: Performance Figures for x -collusion resistance

previously selected 12 parcels. Then we compared different execution times and we noticed that execution times were very similar too.

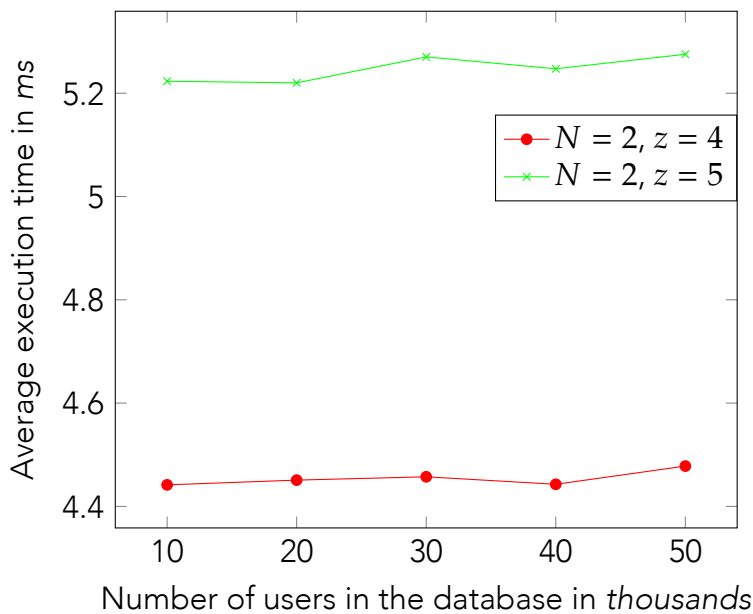
As for the final benchmark, we opted to test x - and (x, y, z) -collusion resistance with 10K, 20K, 30K, 40K and 50K users. We selected one user according to the previously mentioned criteria, and we run the x -collusion resistance for that user 10K times (10K times for 10K users, 20K users, etc.).

The runtime of the QBA enforcement algorithm with x -collusion resistance is affected by the number of users present in the database, i.e. it should grow linearly with the number of users, and this is what we see in Figure 8.6a. In the same manner Figure 8.6b shows a linear growth in execution time with respect to the number of neighbors per parcel, which was also predictable.

Similarly, we ran the algorithm for (x, y, z) -collusion resistance 10K times, with different values of z : 2 to 7 to show the effect of the choice of z . As for QBA enforcement with (x, y, z) -collusion resistance we see that it is running the Breadth First Search algorithm several times: $(deg(parcelid) + 1)$ times maximum. In our benchmarks, we made sure that the user reached KL in all zones to force the (x, y, z) -collusion resistance algorithm to run BFS $(deg(parcelid) + 1)$ times, although it's a rare situation in the cadastral application. BFS has $O(b^z)$ runtime complexity (where b is the branching factor, or the average number of neighbors per parcel which is 4.55 for Maupiti), therefore you would expect the algorithm to have a $O(deg(parcelid) + 1) \times (b^z) \approx O(b^{z+1})$ runtime complexity. Indeed, this is what Figure 8.7a shows us. This complexity would be exponential if z was variable, however, z is not: it should be fixed for the complete database; therefore the complexity of (x, y, z) -collusion resistance is polynomial with degree $z + 1$. On the other hand, Figure 8.7b shows a linear growth of the algorithm with respect to the number of users in the database (i.e. once z is fixed; we only show 2 cases for ease of visualization).



(a) Execution Time of (x, y, z) -collusion resistance for Parcels with Neighbors Ranging from 2 to 13 and z Ranging from 2 to 7



(b) Execution Time of (x, y, z) -collusion resistance as a Function of the Number of Users in the DB where the Number of Neighbors $N = 2$ and $z = 4$ and 5

Figure 8.7: Performance Figures for (x, y, z) -collusion resistance

8.2.2 Comparing Zones and Dominant Zones

In this section we will show how the usage of “*dominant zones*” is advantageous. We will compare our implementations of (x, y, z) -collusion resistance for zones and dominant zones (both based on Algorithm 2) and a second implementation of dominant zones where we calculate dominant zones, only, on-the-fly. Displayed lines of Algorithm 5 show the necessary modification to compute dominant zones on-the-fly: they replace line 6 of Algorithm 2. First of all we change the name of `requestedZones` to `potentialZones`: this is the list of all zones (line 6). Now we calculate the zone with maximum cardinality before selecting dominant zones (line 7): G indicates the application of an aggregate function and the lefthand subscript indicates a SQL *Group By* (i.e. *COUNT* and *MAX* are aggregate functions, and $_{PID}G_{COUNT(PIDN)}$ indicates that the results of the aggregate are grouped by *PID*).

Algorithm 5: Modifications for Algorithm 2 to Compute Dominant Zones on-the-fly

```

6 PotentialZones ← (πPIDNσPID=ParcelID ZONE) ∪ {ParcelID}
7 DominantSize ← GMAX(COUNT(PIDN)) (PIDGCOUNT(PIDN) (PotentialZones ⋈
8                                     Neighbor))
9 RequestedZones ← πPIDσCOUNT(PIDN)=DominantSize (PIDNGCOUNT(PIDN)
10 (PotentialZones ⋈ Neighbor))

```

In fact, Figure 8.8 compares the execution time of (x, y, z) -collusion resistance for zones and dominant zones. Dominant zones were implemented according to Algorithms 2 and 5. In the first, we create a *DOMINANT-ZONE* relation to store dominant zones and in the second we compute them on-the-fly (OTF). Both Figure 8.8a shows the execution time of the three different implementations for $y = 3$ and for different values of z . Figure 8.8b does the same thing but for $y = 4$. These results are valid for other values of y , but we chose to reduce it to two examples for clarity.

Figures 8.8a and 8.8b are the results of the following experiment: we chose 300 random parcels from the database of Maupiti. For every algorithm, for different values of y and z (3 to 4, and 2 to 6, respectively), we clear all stored history of the database. Afterwards we create 100 users, make them access selected parcels in the same order, and we calculate the average time of this traversal.

Before we compare zones to dominant zones, we need to compare both implementations of the latter. As it is clear in Figure 8.8b, adding the storage dominant zones in the database yields better performance (around 50% performance gain) for the QBA enforcement algorithm.

Let us now compare the zone and dominant zone implementations. The first impression is puzzling: on one hand QBA enforcement performs better with zones, especially for low values of z , and on the other hand performance under zones accelerate exponentially, and is inferior to the one under dominant zones for higher values of z . These figures are puzzling especially that both Algorithms 2 and 5 use Breadth-First search on the NEIGHBOR relation. The explanations for both results are the following:

1. *Dominant Zone DB* performs a BFS and filters the result to include dominant zones of ZONE only, while *Zone* does no filtering at all, which explains the performance hit on low values of z .
2. The number of “active zones” returned by BFS in *Dominant Zone DB* is far lower than that of *Zone* (as a result of Point 1). This will affect the time designated to update ZONE - USER and PARCEL - USER. Figure 8.9 shows both average and minimum number of parcels returned by BFS for both zones and dominant zones. As it is clear in this figure, the average and maximum number of parcels that could be returned for dominant zones is far lower and does not experience dramatic jumps like the case for zone.

Which means that we gain on performance on the expense of storage (more storage is needed when compared to zones) and formalism (tolerating redundancy

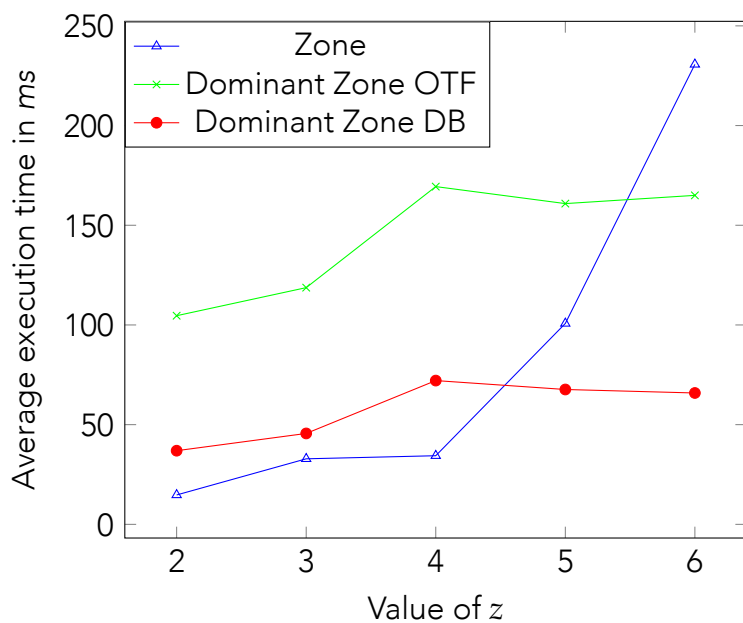
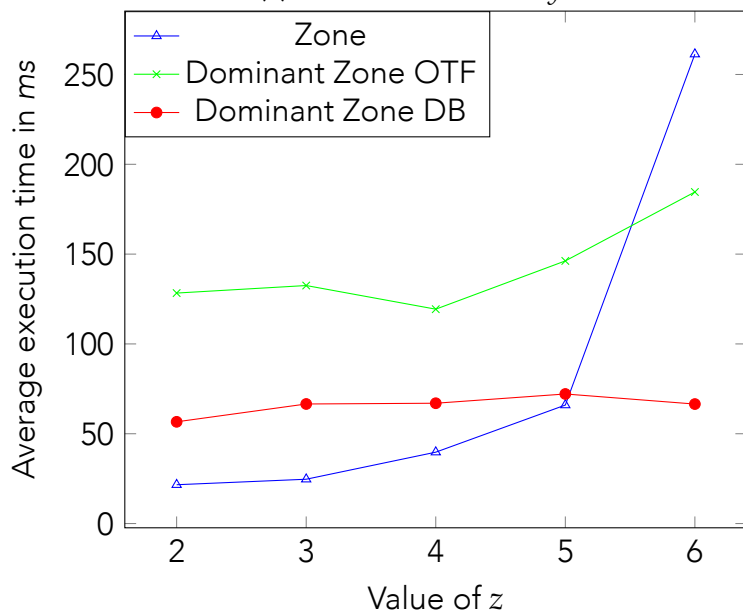
(a) Execution Time for $y = 3$ (b) Execution Time for $y = 4$

Figure 8.8: Execution Time of (x, y, z) -collusion resistance for 2 Values of y (3 and 4 for Figures 8.8a and 8.8b respectively). OTF and DB Designate Algorithms 5 and 2 Respectively, where "Dominant Zones" are Computed *On-The-Fly* and Retrieved from the *Data Base*, Respectively

when compared to OTF). This gain in performance is not exclusive to the QBA enforcement algorithm. We also gain performance on update operations: every time a parcel is updated (merge/split), a BFS should be applied per user per “zone” or “dominant zone.”

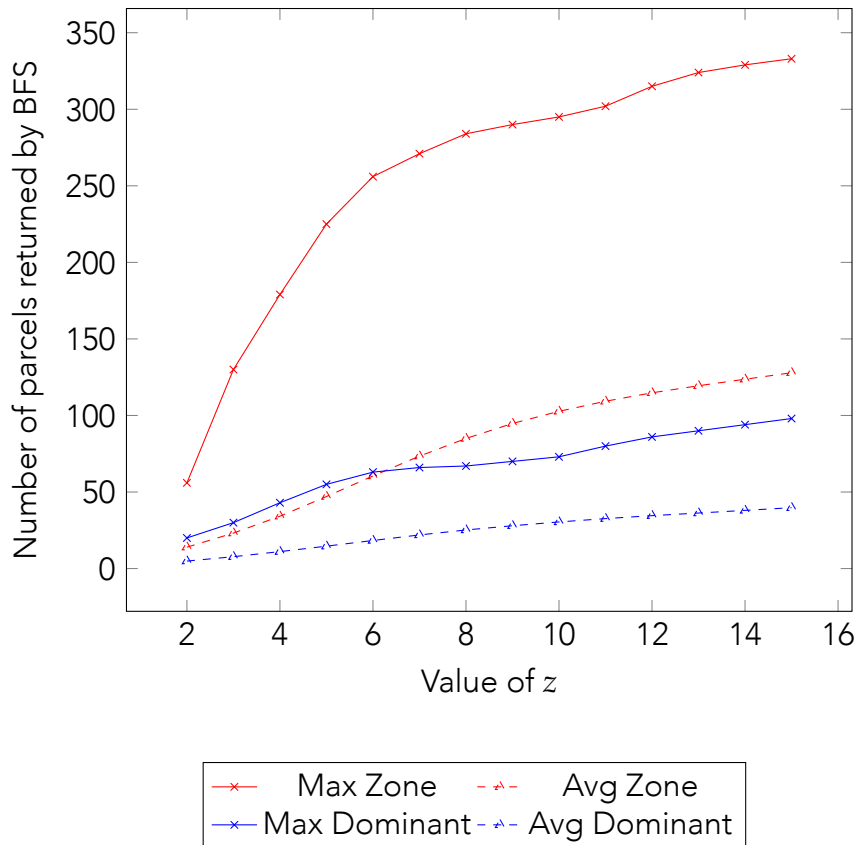


Figure 8.9: Comparison of the Number of Parcels, Resulting from Breadth-First Traversal, as a Function of z

Therefore dominant zones are beneficial and superior to zones, especially for higher values of z. And this is apparent in Figure 8.9, which shows essentially that the average performance of Breadth-First traversal using zones is lower than the worst-case performance using dominant zones.

9

Conclusion

In this thesis, we presented two distinct yet similar QBA problems. The goal was to publish—on the Internet—the cadastral database of French Polynesia. We explained the legislative point of view on the subject. Since cadastral data contain personal information, the law imposes some restrictions on its online publication. These restrictions are expressed in Pr_1 and Pr_2 which are two QBA problems. We also reviewed the state of the art on inference, aggregation and QBA problems where we paid special attention to the work of Motro, Marks and Jajodia [MMJ94; MMJ96]. In their work, they consider that the user can execute “*arbitrary queries*” while we consider that users can select one tuple at a time. This makes their model not inappropriate to the cadastral application.

Afterwards we presented our model: how to enforce and implement Pr_1 and Pr_2 . We introduced different concepts like zones, dominant zones and z -regions. We needed a dynamic definition of a region that captures people’s perception of a region. The zone of a given parcel is the parcel itself and all its neighbors. Every parcel belongs to its proper zone and the zone of every neighboring parcel. A dominant zone of a parcel is the zone containing the parcel with the highest number of parcels in it. A z -region is a generalization of a zone that goes beyond 1st degree neighbors; it reaches the z^{th} neighbor.

We also treated collusions: when multiple users collaborate to circumvent any prohibition. As a matter of fact we presented two collusion resistance schemes. x -collusion resistance is used to prevent x users from colluding on the same dominant zone. With (x, y, z) -collusion resistance, a user is considered as a potential colluder as soon as he exceed the allowed threshold of y collusions in a given z -region, after which the x -collusion resistance scheme is enforced on that z -region.

We showed how to implement the model in relational databases which is the most suitable solution to integrate QBA control with the actual products of the computer science service of French Polynesia—that maintains the cadastral database.

We tackled additional aspects related to QBA control: mutations, which are updates in the cadaster, and how to properly handle them to allow equal access rights to all users; how to periodically, and gradually, reset access to users; how to anticipate inference channels that could arise from QBA enforcement itself due to users' background knowledge and/or a denial of access; and finally, authentication and the need for “*strong*” authentication to avoid Sybil attacks.

The discussion on QBA control in cadastral databases was general, and could be applied to any cadastral database. We dedicated a chapter to talk about specific aspects of the application of QBA control to the cadaster of French Polynesia, namely the current physical process of “*cadastral excerpt requests*” and how the real-estate service intends to keep as much as possible of the workflow when developing the online application. In that context, we showed that a basic authentication scheme (e.g. based on IP addresses) is sufficient for the real-estate service. We also showed that Pr_1 can be implemented with (x, y, z) -collusion resistance and Pr_2 can be implemented with x -collusion resistance only because of the way ownership information is stored in this database.

Additionally, we showed that a successful publication of the cadastral data requires serious fine-tuning by the security administrator: x , y , and z should

be carefully chosen until s/he gets what s/he evaluates as the best compromise between data availability and its confidentiality on one hand, and Computational resources and traffic on the other.

Finally, we presented our prototype that is currently accessible online. We detailed every component and even showed the graph generation process using R-trees. Then, we showed in our benchmarks how the use of dominant zones for QBA control, instead of zones only, provides higher availability of data while keeping their confidentiality. We also showed performance benchmarks for the developed algorithms. We showed that our algorithms' performance is acceptable and can be used in a production environment.

It is worth mentioning that the model we presented is not restricted to cadastral databases. One can imagine using this model in any application where k out of N disclosure control for cases where "*phonebook entries*" are shared among multiple "*phonebooks*" (like in Pr_1), or where a meaningful relationship between "*phonebook entries*" exists (like in Pr_2).

For example, the Tahitian pearl is French Polynesia's largest export, and pearl farms can be found at different marine sites on different islands. An application designated for pearl farmers and traders where they can discover already used farming sites could be very beneficial: it could be used to see who is the owner of a farm, his contact information, etc. to start a commercial partnership with her/him. QBA control can be used to limit the abuse of such an application; for instance, preventing a user from seeing all farming sites on an island, or preventing her/him from acquiring the knowledge of all farming sites of the same farmer.

Currently, we are in the beginning stages of the implementation of a production-ready web application that will be at the disposal of the public. We are negotiating, in partnership with the real-estate service of French Polynesia and computer science department, technical and logistic details with a third-party—a renowned company in GIS development.

9.1 Future Work

The first thing that should be thoroughly investigated is the model itself: *how to better track colluding users? How to improve the ability to label users as colluders and non-colluders? Is it possible to further minimize the number of “active zones” (like the shift we did from zones only to dominant zones), thus achieving better performance and possibly more control over data availability?*

Due to the scarcity of work targeting QBA control in general, research is much required. The work presented here targets QBA control for cadastral databases. Further investigation should be conducted to try to find a unifying framework for our work and that of Motro, Marks and Jajodia [MMJ94; MMJ96] that targets general QBA control. *Could these works be unified under a single framework? Is QBA control in the cadastral database a special case of (what we dubbed) general QBA, or is it of a different type?*

Another interesting topic is graph generation itself. Currently, we use an “intuitive” method to generate the neighborhood graph of a set of parcel using a spatial index, namely an R-tree. A study can be devised comparing our method and other possible ones in terms of processing time, memory consumption and precision. Indeed, precision should be taken into account when considering parcels that are separated by roads, rivers, etc. which require a threshold τ greater than 0. Our current method may fall short on some special cases (with “unusual” shapes and/or configurations of parcels), therefore the need for further research.

And last but not least: the implementation. We used the relational model because it is more convenient for the computer-science service. However future work should include studies comparing different implementation strategies: *should the graph be stored entirely in the database? Given the fact that some user queries are distant (e.g. a user can query a parcel in the center of the city, while another one queries a parcel far away in the country-side), could QBA control be parallelized?*

A

Example of a Cadastral Excerpt



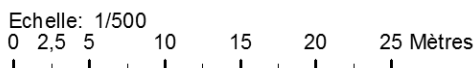
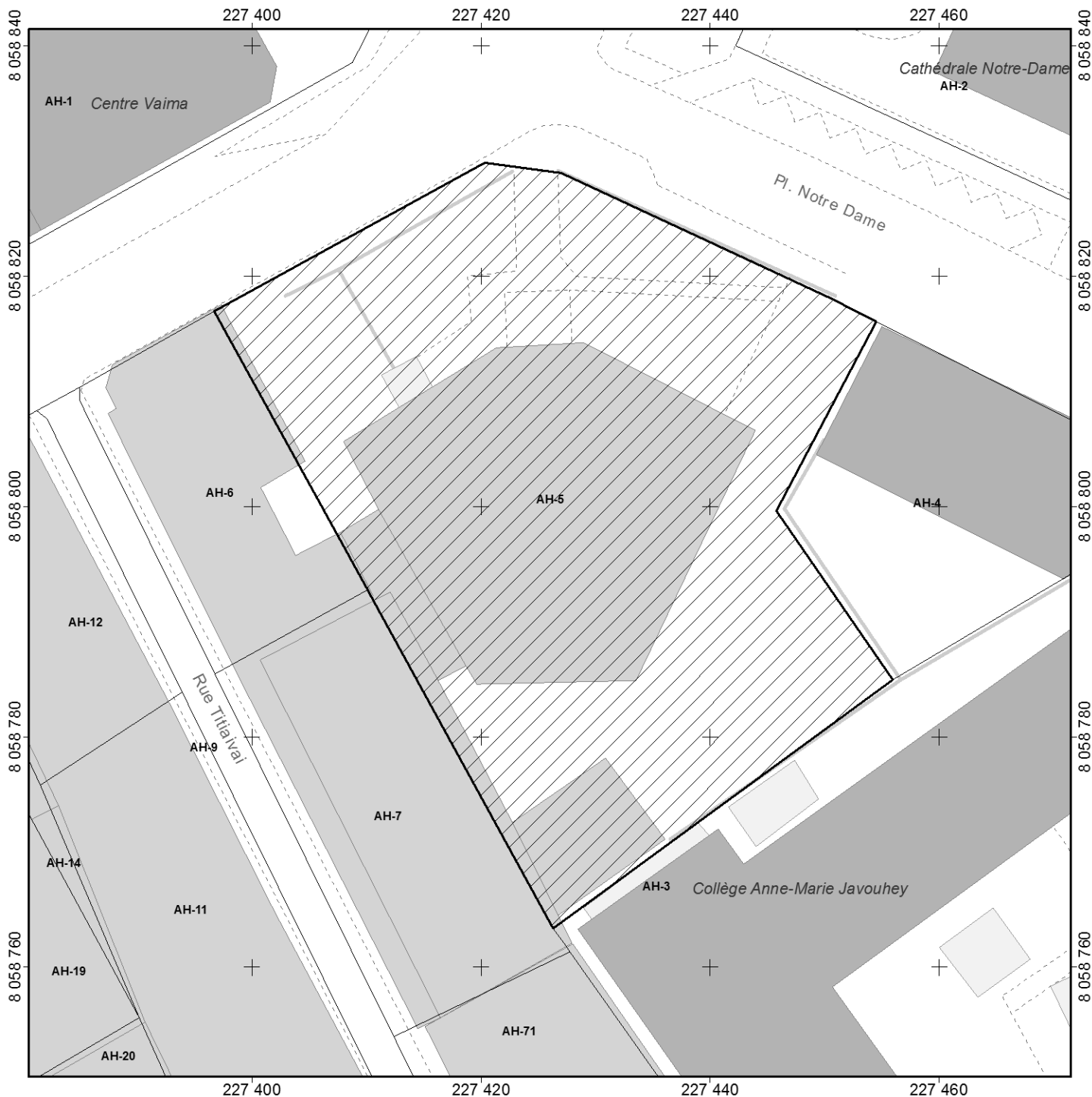
Extrait de plan Cadastral

L'extrait de plan cadastral ne constitue pas un titre de propriété

Ile : Tahiti
 Commune : PAPEETE
 Com. Associée :

Direction des Affaires Foncières
 Division du cadastre

Parcelle	Surface (m ²)	Terre	Propriétaire
AH-5	2281	TITIAIVAI PARCELLE	BANQUE DE POLYNESIE



Pour le Ministre et par délégation
 le directeur des affaires foncières
 Cadastre au 20/11/2014

B

The Model of Motro, Marks and Jajodia Applied to the Cadastral Database

In order to apply Motro, Marks and Jajodia's model [MMJ94; MMJ96] we start with the following hypotheses:

1. Database:
 - (a) The cadaster is analogous to the phonebook.
 - (b) Static
2. We will only consider Pr_1 . Thus, every dominant zone forms a sensitive aggregate.
3. A user cannot submit "*arbitrary*" queries. Selections are only allowed on primary keys.

Table B.1: Tuples of the CADASTER Relation corresponding to Figures 3.1 and 3.2

PARCELID	OWNER	Geometry	DOMINANT
1	Joe	Polygon	3
2	Joe	Polygon	3
3	Lucy	Polygon	3
4	Elissa	Polygon	5
5	Elissa	Polygon	3
5	Lucy	Polygon	3
6	Elissa	Polygon	5
6	Elissa	Polygon	7
7	Elissa	Polygon	3
8	Joe	Polygon	3

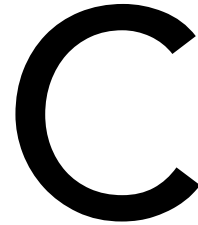
Notice that we assumed the exact same hypothesis as the authors except for the allowed querying behavior which is a fundamental requirement of the cadastral application. We will represent the cadastral database with a single relation: CADASTER (PARCELID, OWNER, GEOMETRY, DOMINANT) as shown in Table B.1. The authors recommended this representation: «*Databases consisting of several relations may be treated view the Universal Relation formalism*».

It follows from this hypothesis that a sensitive aggregate S_i representing a parcel p_i has the following pattern $(*, *, *, ID_{dom_{p_i}})$, where $N_i = |dom_{p_i}|$. Every user query Q requesting the owner of a parcel p_i has the following pattern $(ID_{p_i}, *, *, *)$. Therefore, every query Q will cover all sensitive aggregates. However, we know beforehand that reading the owner of a parcel p_i only affects dominant zones containing p_i . This fact opens the door for an optimization: the input of Algorithm 1 should be Q and the set of sensitive aggregates are the dominant zones which p_i belongs to. A modification of Algorithm 1 is depicted in Algorithm 6.

Notice that Algorithm 6 does not use the overlapping test and restrictions (lines 4 and 5, respectively, of Algorithm 1 on page 29), which raises the question on the usability of the proposed model.

Algorithm 6: Modified QBA Control Enforcement Algorithm in the Model of Motro, Marks and Jajodia for the Cadastral Application

```
Input: A user query  $Q$  and a set of sensitive aggregates  $S$   
Output: Tuples from the database or the empty set  $\emptyset$   
1 res = Materialize  $Q$   
2 for all  $S_i \in S$  do  
3    $M_i = 0$   
4   if  $\alpha \in P_i$  then  
5     continue  
6   if  $D_i + 1 > k_i$  then  
7     return  $\emptyset$   
8 end  
9 for  $i = 1, 2 \dots N$   
10  if  $\alpha \in P_i$  then  
11    continue  
12     $D_i = D_i + 1$   
13     $P_i = P_i \vee \alpha$   
14 end  
15 return res
```

Enforcement Scripts in PL/SQL

Please note that

1. all algorithms presented here are for learning purposes only. Actual implementations for production environments may (and will) differ.
2. some keywords are reserved in Postgres, so we had to change them a little bit (e.g. OWNER becomes OWNr)

Listing C.1: Enforcement Algorithm in PL/SQL

```
1 CREATE OR REPLACE FUNCTION xyzcr_(parcelid integer,  
2                               userid integer,  
3                               y integer,  
4                               z integer)  
5 RETURNS SETOF ownr AS  
6 $BODY$  
7  
8 DECLARE  
9   maxed          integer; -- Used to check if disclosed > kh  
10                  --                               or ydisclosed > y  
11   expection     text;    -- Stores exception message  
12   dominantZones CURSOR (clef integer) FOR      -- Dominant Zones  
13     SELECT pidn FROM zones WHERE pid = clef; -- cursor  
14  
15   potential     CURSOR(ppp integer, uuu integer) FOR  
16     SELECT DISTINCT p.pid  
17     FROM   parcel p  
18     JOIN   zone_user z ON p.pid = z.pid
```

```

19     WHERE p.pid IN (SELECT z.pidn
20                     FROM   zones z
21                     WHERE  z.pid IN (SELECT n.pidn
22                                     FROM   neighbor n
23                                     WHERE  n.pid = parcelid
24                                     UNION  SELECT parcelid))
25     AND   z.disclosed = p.kl
26     AND   z.uid = uuu;
27
28     -- A cursor for breadth-first traversal
29     bfs curs CURSOR(p integer, z integer) FOR
30         SELECT pid FROM bfs_(p, z);
31
32 BEGIN
33
34 IF (SELECT count(pid)           -- See if the user has already
35     FROM   parcel_user pu       -- accessed the requested parcel
36     WHERE  pu.uid = userid
37     AND    pu.pid = parcelid
38     ) > 0
39 THEN
40     RETURN QUERY                -- If he did, return results
41     SELECT o.oid, o.nam         -- immediatly
42     FROM   ownr o
43     JOIN   ownr_parcel op ON o.oid = op.oid
44     WHERE  op.pid = parcelid;
45 ELSE                             -- If he didn't
46     SELECT count(p.pid) INTO maxed
47     FROM   parcel p
48     JOIN   zone_user z ON p.pid = z.pid -- See the number
49     AND    z.uid = userid              -- of dominant zones
50     AND    p.pid IN (SELECT z.pidn     -- where he reached
51                     FROM   zones z
52                     WHERE  z.pid IN (SELECT n.pidn
53                                     FROM   neighbor n
54                                     WHERE  n.pid = parcelid
55                                     UNION  SELECT parcelid))
56     AND    z.disclosed >= p.kh;
57
58 IF maxed > 0 THEN                -- If he reached KH in 1 or more
59     RETURN;                       -- dominant zones, then return nothing
60 END IF;
61
62 IF(SELECT count(p.pid)           -- If the number of
63     FROM   parcel p              -- dominant zones
64     JOIN   zone_user z ON p.pid = z.pid -- where DISCLOSED
65     WHERE  p.pid IN (SELECT z.pidn     -- might get bigger
66                     FROM   zones z
67                     -- than KL is > 0

```

```

67         WHERE z.pid IN (SELECT n.pidn
68                        FROM   neighbor n
69                        WHERE  n.pid = parcelid
70                        UNION  SELECT parcelid))
71     AND   z.disclosed = p.kl
72     AND   z.uid = userid) > 0      -- Then we should check
73 THEN                                     -- if they pass the limit
74     SELECT count(zu.pid) INTO maxed -- y
75     FROM   zone_user zu, ydiskcalc_(parcelid, userid, z) xx
76     WHERE  zu.uid = userid
77     AND    zu.pid = xx.pid
78     AND    zu.ydisclosed + xx.ydisclosed > y;
79
80     IF maxed > 0      -- If the number of dominant zones
81     THEN             -- that go beyond y is positive
82         exception = ''; -- return an exception listing all
83         FOR maxed IN SELECT zu.pid      -- dominant zones
84                        FROM   zone_user zu, -- causing it
85                        ydiskcalc_(parcelid, userid, z) xx
86                        WHERE  zu.uid = userid
87                        AND    zu.pid = xx.pid
88                        AND    zu.ydisclosed + xx.ydisclosed > y
89     LOOP
90         exception = exception || ' ' || maxed;
91     END LOOP;
92     exception = 'Reached y ' || exception;
93     RAISE EXCEPTION USING MESSAGE =
94         'Reached y: ' || exception;
95     END IF;
96 END IF;
97
98 --Update code for disclosed
99 INSERT INTO parcel_user VALUES(parcelid, userid);
100 FOR x in dominantZones (parcelid) LOOP
101     IF (SELECT count(*)
102        FROM   zone_user
103        WHERE  pid = x.pidn
104        AND    uid = userid) > 0
105     THEN
106         UPDATE zone_user
107         SET     disclosed = disclosed + 1
108         WHERE  pid = x.pidn
109         AND    uid = userid;
110     ELSE
111         INSERT INTO zone_user VALUES(x.pidn,userid,1,0);
112     END IF;
113 END LOOP;
114

```



```
115  --update code for ydisclosed
116  FOR x IN potential (parcelid, userid) LOOP
117      FOR y IN bfscurs (x.pid, z) LOOP
118          BEGIN
119              UPDATE zone_user
120              SET      ydisclosed = ydisclosed + 1
121              WHERE   uid = userid
122              AND     pid = y.pid;
123          BEGIN
124              INSERT INTO zone_user
125              VALUES (y.pid, userid, 0, 1);
126              EXCEPTION WHEN unique_violation THEN
127              UPDATE zone_user
128              SET      ydisclosed = ydisclosed + 1
129              WHERE   uid = userid
130              AND     parcelid = y.pid;
131          END;
132      END;
133  END LOOP;
134 END LOOP;
135
136 RETURN QUERY          -- return ownership information
137 SELECT o.oid, o.nam
138 FROM   ownr o
139 JOIN   ownr_parcel op on o.oid = op.oid
140 WHERE  op.pid = parcelid;
141 END IF;
142 END;
143
144 $BODY$
145 LANGUAGE plpgsql VOLATILE;
```

Listing C.2: Building Breadth-First Traversal SQL Request

```

1 CREATE OR REPLACE FUNCTION ydiskcalc_(IN parcelid integer,
2                                     IN userid integer,
3                                     IN z integer)
4 RETURNS TABLE(pid integer, ydisclosed bigint) AS
5 $BODY$
6
7 DECLARE
8   ret      text;      -- The resulting query
9   potential CURSOR(ppp integer, uuu integer) FOR
10  SELECT p.pid          -- A cursor for the set
11  FROM   parcel p      -- of dominant zones
12  JOIN   zone_user z ON p.pid = z.pid -- belonging to the parcel
13  WHERE  p.pid IN (SELECT z.pidn     -- and its direct
14                FROM   zones z     -- neighbors
15                WHERE  z.pid IN (SELECT n.pidn
16                              FROM   neighbor n
17                              WHERE  n.pid = ppp
18                              UNION SELECT ppp))
19        AND z.disclosed = p.kl
20        AND z.uid = uuu;
21   pot      integer;
22   append   boolean;
23 BEGIN
24   ret = 'SELECT pid, count(pid) as ydisclosed FROM (';
25   append = false;
26   FOR pot in potential(parcelid, userid) LOOP
27     IF append THEN
28       ret = ret || ' UNION ALL ';
29     ELSE
30       append = true;
31     END IF;
32     ret = ret || 'SELECT pid FROM bfs_(' || pot || ',' || z || ')';
33   END LOOP;
34   ret = ret || ') as sub GROUP BY pid';
35   -- RAISE NOTICE '%',ret; -- If you want to visualize
36   -- the resulting query
37   RETURN QUERY EXECUTE ret;
38 END;
39
40 $BODY$
41 LANGUAGE plpgsql STABLE;

```

Listing C.3: Breadth-First Traversal Implementation in SQL (1)

```

1 CREATE OR REPLACE FUNCTION bfs_(IN parcelid integer,
2                               IN z integer)
3 RETURNS TABLE(pid integer, k1 integer, kh integer) AS
4 $BODY$
5
6 BEGIN
7
8 RETURN QUERY
9 SELECT distinct b.*
10 FROM   bfs(parcelid, z) d, zones z
11 WHERE  b.pid = z.pidn;
12 END;
13
14 $BODY$
15 LANGUAGE plpgsql VOLATILE

```

Listing C.4: Breadth-First Traversal Implementation in SQL (2)

```

1 CREATE OR REPLACE FUNCTION bfs(IN parcelid integer,
2                               IN z integer)
3 RETURNS TABLE(pid integer, k1 integer, kh integer) AS
4 $BODY$
5
6 BEGIN
7
8 RETURN QUERY
9 WITH RECURSIVE search_graph(root, depth) AS (
10     SELECT parcelid, 0
11     UNION
12     SELECT nxt.pidn, sg.depth + 1
13     FROM neighbor nxt, search_graph sg
14     WHERE sg.root = nxt.PID
15     AND sg.depth <= z
16 )
17 SELECT DISTINCT s.root, p.k1, p.kh
18 FROM search_graph s
19 JOIN parcel p on p.pid = s.root;
20 END;
21
22 $BODY$
23 LANGUAGE plpgsql STABLE

```

Bibliography

- [09] *Les guides de la CNIL. Les collectivités locales*. 2009. url: http://www.cnil.fr/fileadmin/documents/Guides_pratiques/CNIL_Guide_CollLocales.pdf (see p. 15).
- [13] *Fiscalité locale et cadastre*. accessed June 2013. url: <http://www.cada.fr/fiscalite-locale-et-cadastre,6090.html> (see p. 15).
- [98] *ESRI Shapefile Technical Description. An ESRI White Paper*. July 1998. url: <http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf> (see p. 81).
- [AC09] Bechara Al Bouna and Richard Chbeir. “Detecting Inference Channels in Private Multimedia Data via Social Networks”. In: *Data and Applications Security XXIII, 23rd Annual IFIP WG 11.3 Working Conference, Montreal, Canada, July 12-15, 2009. Proceedings*. 2009, pp. 208–224. doi: [10.1007/978-3-642-03007-9_14](https://doi.org/10.1007/978-3-642-03007-9_14) (see p. 20).
- [AGC13] Firas Al Khalil, Alban Gabillon, and Patrick Capolsini. “Collusion Resistant Inference Control for Cadastral Databases”. In: *Foundations and Practice of Security - 6th International Symposium, FPS 2013, La Rochelle, France, October 21-22, 2013, Revised Selected Papers*. 2013, pp. 189–208. doi: [10.1007/978-3-319-05302-8_12](https://doi.org/10.1007/978-3-319-05302-8_12) (see pp. 6, 97).

- [AGC14a] Firas Al Khalil, Alban Gabillon, and Patrick Capolsini. “Implementing Quantity Based Aggregation Control for Cadastral Databases”. In: *2014 IEEE World Congress on Services, Anchorage, AK, USA, June 27 - July 2, 2014*. 2014, pp. 137–144. doi: [10.1109/SERVICES.2014.33](https://doi.org/10.1109/SERVICES.2014.33) (see pp. 8, 97).
- [AGC14b] Firas Al Khalil, Alban Gabillon, and Patrick Capolsini. “Quantity Based Aggregation Control for Cadastral Databases”. In: *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Privacy in Geographic Information Collection and Analysis, GeoPrivacy '14, Dallas/Fort Worth, Texas, USA, November 4-7, 2014*. 2014, 7:1–7:8. doi: [10.1145/2675682.2676394](https://doi.org/10.1145/2675682.2676394) (see pp. 9, 98).
- [AY08] Charu Aggarwal and Philip Yu, eds. *Privacy-Preserving Data Mining*. Vol. 34. Advances in Database Systems. Springer Science+Business Media, LLC, 2008. doi: [10.1007/978-0-387-70992-5](https://doi.org/10.1007/978-0-387-70992-5) (see p. 21).
- [BB04] Joachim Biskup and Piero A Bonatti. “Controlled Query Evaluation for Known Policies by Combining Lying and Refusal”. In: *Ann. Math. Artif. Intell.* 40.1-2 (2004), pp. 37–62. doi: [10.1023/A:1026106029043](https://doi.org/10.1023/A:1026106029043) (see p. 19).
- [Bez+10] Michele Bezzi et al. “Protecting privacy of sensitive value distributions in data release”. In: *Security and Trust Management - 6th International Workshop, STM 2010, Athens, Greece, September 23-24, 2010, Revised Selected Papers*. Springer, 2010, pp. 255–270 (see p. 25).
- [Bez+12] Michele Bezzi et al. “Modeling and preventing inferences from sensitive value distributions in data release”. In: *Journal of Computer Security* 20.4 (2012), pp. 393–436 (see p. 25).

- [BFJ00] Alexander Brodsky, Csilla Farkas, and Sushil Jajodia. “Secure Databases: Constraints, Inference Channels, and Monitoring Disclosures”. In: *IEEE Trans. on Knowl. and Data Eng.* 12.6 (Nov. 2000), pp. 900–919. doi: [10.1109/69.895801](https://doi.org/10.1109/69.895801) (see p. 17).
- [BN89] David FC Brewer and Michael J Nash. “The Chinese Wall Security Policy”. In: *Proceedings of the 1989 IEEE Symposium on Security and Privacy, Oakland, California, USA, May 1-3, 1989*. 1989, pp. 206–214. doi: [10.1109/SECPRI.1989.36295](https://doi.org/10.1109/SECPRI.1989.36295) (see pp. 13, 23, 34).
- [Bou+94] Nora Boulahia-Cuppens et al. “Decomposition of Multilevel Objects in an Object-Oriented Database”. In: *Computer Security - ESORICS 94, Third European Symposium on Research in Computer Security, Brighton, UK, November 7-9, 1994, Proceedings*. 1994, pp. 375–402. doi: [10.1007/3-540-58618-0_75](https://doi.org/10.1007/3-540-58618-0_75) (see p. 17).
- [BT11] Joachim Biskup and Cornelia Tadros. “Inference-Proof View Update Transactions with Minimal Refusals”. In: *Data Privacy Management and Autonomous Spontaneous Security - 6th International Workshop, DPM 2011, and 4th International Workshop, SETOP 2011, Leuven, Belgium, September 15-16, 2011, Revised Selected Papers*. 2011, pp. 104–121. doi: [10.1007/978-3-642-28879-1_8](https://doi.org/10.1007/978-3-642-28879-1_8) (see p. 19).
- [Cal12] Douglas R Caldwell. *Unlocking the mysteries of the bounding box*. ALA Map and Geography Round Table. 2012. url: <http://hdl.handle.net/1969.1/129183> (see p. 86).
- [CC08] Yu Chen and Wesley Chu. “Protection of Database Security Via Collaborative Inference Detection”. In: *Intelligence and Security Informatics, Techniques and Applications*. Springer, 2008, pp. 275–303 (see pp. 17, 34).

- [CG01] Frédéric Cuppens and Alban Gabillon. “Cover story management”. In: *Data & Knowledge Engineering* 37.2 (2001), pp. 177–201. doi: [10.1016/S0169-023X\(01\)00006-4](https://doi.org/10.1016/S0169-023X(01)00006-4) (see p. 17).
- [CG98] Frédéric Cuppens and Alban Gabillon. “Rules for Designing Multi-level Object-Oriented Databases”. In: *Computer Security - ESORICS 98, 5th European Symposium on Research in Computer Security, Louvain-la-Neuve, Belgium, September 16-18, 1998, Proceedings*. 1998, pp. 159–174. doi: [10.1007/BFb0055862](https://doi.org/10.1007/BFb0055862) (see p. 17).
- [CG99] Frédéric Cuppens and Alban Gabillon. “Logical foundations of multilevel databases”. In: *Data & Knowledge Engineering* 29.3 (1999), pp. 259–291. doi: [10.1016/S0169-023X\(98\)00044-5](https://doi.org/10.1016/S0169-023X(98)00044-5) (see pp. 17, 67).
- [Com10] Wikimedia Commons. *Simple example of an R-tree for 2D rectangles*. 2010. url: <http://commons.wikimedia.org/wiki/File:R-tree.svg> (see p. 87).
- [CT13] Chris Clifton and Tamir Tassa. “On Syntactic Anonymity and Differential Privacy”. In: *Trans. Data Privacy* 6.2 (Aug. 2013), pp. 161–183 (see p. 21).
- [Cup91] Frédéric Cuppens. “A Modal Logic Framework to Solve Aggregation Problems”. In: *Database Security, V: Status and Prospects, Results of the IFIP WG 11.3 Workshop on Database Security, Shepherdstown, West Virginia, USA, 4-7 November, 1991*. 1991, pp. 315–332 (see p. 25).
- [CV13] C. Conejo and A. Velasco. *Cadastral Web Services in Spain*. Accessed June 2013. url: http://www.ec-gis.org/Workshops/13ec-gis/presentations/4_sdi_implementation_I_Conejo_1.pdf (see p. 14).

- [CW05] X. Chen and Ruizhong Wei. “A Dynamic Method for Handling the Inference Problem in Multilevel Secure Databases”. In: *International Symposium on Information Technology: Coding and Computing (ITCC 2005), Volume 1, 4-6 April 2005, Las Vegas, Nevada, USA*. 2005, pp. 751–756. doi: [10.1109/ITCC.2005.7](https://doi.org/10.1109/ITCC.2005.7) (see pp. 18, 62).
- [Den+88] Dorothy E Denning et al. “The SeaView security model”. In: *Proceedings of the 1988 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 18-21, 1988*. IEEE. 1988, pp. 218–233. doi: [10.1109/32.55088](https://doi.org/10.1109/32.55088) (see p. 67).
- [DH06] Jochen Dinger and Hannes Hartenstein. “Defending the Sybil Attack in P2P Networks: Taxonomy, Challenges, and a Proposal for Self-Registration”. In: *Proceedings of the First International Conference on Availability, Reliability and Security. ARES '06*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 756–763. doi: [10.1109/ARES.2006.45](https://doi.org/10.1109/ARES.2006.45) (see p. 69).
- [DH96] Harry S. Delugach and Thomas H. Hinke. “Wizard: A database inference analysis and detection system”. In: *IEEE Transactions on Knowledge and Data Engineering* 8.1 (1996), pp. 56–66. doi: [10.1109/69.485629](https://doi.org/10.1109/69.485629) (see p. 17).
- [DJ01] Jörg Desel and Gabriel Juhás. ““What Is a Petri Net?” Informal Answers for the Informed Reader”. In: *Unifying Petri Nets*. Vol. 2128. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2001, pp. 1–25. doi: [10.1007/3-540-45541-8_1](https://doi.org/10.1007/3-540-45541-8_1) (see p. 20).
- [Don99] Judith S Donath. “Identity and deception in the virtual community”. In: *Communities in cyberspace 1996* (1999), pp. 29–59 (see p. 69).
- [Dou02] John R. Douceur. “The sybil attack”. In: *Peer-to-Peer Systems, First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8,*

- 2002, *Revised Papers*. 2002, pp. 251–260. doi: [10.1007/3-540-45748-8_24](https://doi.org/10.1007/3-540-45748-8_24) (see pp. 58, 69).
- [Dwo06] Cynthia Dwork. “Differential privacy”. In: *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*. 2006, pp. 1–12. doi: [10.1007/11787006_1](https://doi.org/10.1007/11787006_1) (see p. 21).
- [Dwo08] Cynthia Dwork. “Differential Privacy: A Survey of Results”. In: *Theory and Applications of Models of Computation, 5th International Conference, TAMC 2008, Xi’an, China, April 25-29, 2008. Proceedings*. 2008, pp. 1–19. doi: [10.1007/978-3-540-79228-4_1](https://doi.org/10.1007/978-3-540-79228-4_1) (see p. 21).
- [Dwo11] Cynthia Dwork. “A Firm Foundation for Private Data Analysis”. In: *Commun. ACM* 54.1 (Jan. 2011), pp. 86–95. doi: [10.1145/1866739.1866758](https://doi.org/10.1145/1866739.1866758) (see p. 21).
- [FJ02] Csilla Farkas and Sushil Jajodia. “The inference problem: a survey”. In: *ACM SIGKDD Explorations Newsletter* 4.2 (2002), pp. 6–11. doi: [10.1145/772862.772864](https://doi.org/10.1145/772862.772864) (see p. 17).
- [Fol91] Simon N Foley. “A taxonomy for information flow policies and models”. In: *IEEE Symposium on Security and Privacy*. 1991, pp. 98–109. doi: [10.1109/RISP.1991.130778](https://doi.org/10.1109/RISP.1991.130778) (see p. 25).
- [Fol92] Simon N Foley. “Aggregation and separation as noninterference properties”. In: *Journal of Computer Security* 1.2 (1992), pp. 159–188. doi: [10.3233/JCS-1992-1203](https://doi.org/10.3233/JCS-1992-1203) (see p. 25).
- [Fri10] Keith B Frikken. “Algorithms and Theory of Computation Handbook”. In: ed. by Mikhail J. Atallah and Marina Blanton. Chapman & Hall/CRC, 2010. Chap. Secure Multiparty Computation, 14:1–14:16 (see p. 21).

- [FTE01] Csilla Farkas, Tyrone S Toland, and Caroline M Eastman. “The Inference Problem and Updates in Relational Databases”. In: *Database and Application Security XV, IFIP TC11/WG11.3 Fifteenth Annual Working Conference on Database and Application Security, July 15-18, 2001, Niagara on the Lake, Ontario, Canada*. 2001, pp. 181–194 (see p. 17).
- [Fun+10] Benjamin Fung et al. “Privacy-preserving Data Publishing: A Survey of Recent Developments”. In: *ACM Comput. Surv.* 42.4 (June 2010), 14:1–14:53. doi: [10.1145/1749603.1749605](https://doi.org/10.1145/1749603.1749605) (see p. 21).
- [Goo13] Dan Goodin. “Anatomy of a hack: How crackers ransack passwords like “qeadzwcwrsfxv1331””. In: *Arstechnica* (May 27, 2013). url: <http://arstechnica.com/security/2013/05/how-crackers-make-minced-meat-out-of-your-passwords/> (see p. 68).
- [Gut84] Antonin Guttman. “R-trees: A Dynamic Index Structure for Spatial Searching”. In: *SIGMOD’84, Proceedings of Annual Meeting, Boston, Massachusetts, June 18-21, 1984*. 1984, pp. 47–57. doi: [10.1145/602259.602266](https://doi.org/10.1145/602259.602266) (see p. 86).
- [Hin88] Thomas H. Hinke. “Inference aggregation detection in database management systems”. In: *Proceedings of the 1988 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 18-21, 1988*. 1988, pp. 96–106. doi: [10.1109/SECPRI.1988.8101](https://doi.org/10.1109/SECPRI.1988.8101) (see pp. 5, 22, 34).
- [JM95] Sushil Jajodia and Catherine Meadows. “Inference problems in multi-level secure database management systems”. In: *Information Security: An integrated collection of essays 1* (1995), pp. 570–584 (see pp. 17, 23).
- [Kam50] Erich Kamke. *Theory of Sets*. Courier Dover Publications, 1950, p. 6 (see p. 5).

- [KVK11] Vasilios Katos, Dimitris Vrakas, and Panagiotis Katsaros. “A Framework for Access Control with Inference Constraints”. In: *Proceedings of the 35th Annual IEEE International Computer Software and Applications Conference, COMPSAC 2011, Munich, Germany, 18-22 July 2011*. 2011, pp. 289–297. doi: [10.1109/COMPSAC.2011.45](https://doi.org/10.1109/COMPSAC.2011.45) (see p. 18).
- [Lin03] Tsau Young Lin. “Chinese Wall Security Policy Models: Information Flows and Conflicting Trojan Horses”. In: (2003), pp. 275–297 (see p. 24).
- [LLV07] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. “t-closeness: Privacy beyond k-anonymity and l-diversity”. In: 2007, pp. 106–115. doi: [10.1109/ICDE.2007.367856](https://doi.org/10.1109/ICDE.2007.367856) (see p. 20).
- [Loc11] Jan-Hendrik Lochner. “An Effective and Efficient Inference Control System for Relational Database Queries”. PhD thesis. Technischen Universität Dortmund. an der Fakultät für Informatik, 2011 (see p. 19).
- [LSM06] Brian Neil Levine, Clay Shields, and N. Boris Margolin. “A survey of solutions to the sybil attack”. In: *University of Massachusetts Amherst, Amherst, MA* (2006) (see p. 69).
- [Lun89] T.F. Lunt. “Aggregation and inference: Facts and fallacies”. In: *Proceedings of the 1989 IEEE Symposium on Security and Privacy, Oakland, California, USA, May 1-3, 1989*. 1989, pp. 102–109. doi: [10.1109/SECPRI.1989.36284](https://doi.org/10.1109/SECPRI.1989.36284) (see pp. 5, 23, 34).
- [LX14] Xiaowei Li and Yuan Xue. “A Survey on Server-side Approaches to Securing Web Applications”. In: *ACM Comput. Surv.* 46.4 (Mar. 2014), 54:1–54:29. doi: [10.1145/2541315](https://doi.org/10.1145/2541315) (see p. 80).

- [LXL13] Jinfei Liu, Li Xiong, and Jun Luo. “Semantic Security: Privacy Definitions Revisited”. In: *Trans. Data Privacy* 6.3 (Dec. 2013), pp. 185–198 (see p. 21).
- [Mac+07] Ashwin Machanavajjhala et al. “l-diversity: Privacy beyond k-anonymity”. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1.1 (2007), p. 3. doi: [10.1145/1217299.1217302](https://doi.org/10.1145/1217299.1217302) (see p. 20).
- [Mat03] Kanta Matsuura. “Digital Security Tokens and Their Derivatives”. In: *Netnomics* 5.2 (Nov. 2003), pp. 161–179. doi: [10.1023/A:1026052902146](https://doi.org/10.1023/A:1026052902146) (see p. 68).
- [Mea90] Catherine Meadows. “Extending the Brewer-Nash model to a multi-level context”. In: *Proceedings of the 1990 IEEE Symposium on Security and Privacy, Oakland, California, USA, May 7-9, 1990*. 1990, pp. 95–103. doi: [10.1109/RISP.1990.63842](https://doi.org/10.1109/RISP.1990.63842) (see pp. 24, 25, 34).
- [MMJ94] Amihai Motro, Donald G Marks, and Sushil Jajodia. “Aggregation in relational databases: Controlled disclosure of sensitive information”. In: *Computer Security - ESORICS 94, Third European Symposium on Research in Computer Security, Brighton, UK, November 7-9, 1994, Proceedings*. 1994, pp. 431–445. doi: [10.1007/3-540-58618-0_77](https://doi.org/10.1007/3-540-58618-0_77) (see pp. 5, 13, 26, 115, 118, 121).
- [MMJ96] Donald G Marks, Amihai Motro, and Sushil Jajodia. “Enhancing the controlled disclosure of sensitive information”. In: *Computer Security - ESORICS 96, 4th European Symposium on Research in Computer Security, Rome, Italy, September 25-27, 1996, Proceedings*. 1996, pp. 290–303. doi: [10.1007/3-540-61770-1_42](https://doi.org/10.1007/3-540-61770-1_42) (see pp. 5, 29, 115, 118, 121).
- [MS04] Gerome Miklau and Dan Suciu. “A Formal Analysis of Information Disclosure in Data Exchange”. In: *Proceedings of the ACM SIGMOD*

- International Conference on Management of Data, Paris, France, June 13-18, 2004*. New York, NY, USA: ACM, 2004, pp. 575–586. doi: [10.1145/1007568.1007633](https://doi.org/10.1145/1007568.1007633) (see p. 18).
- [MT07] Frank McSherry and Kunal Talwar. “Mechanism Design via Differential Privacy”. In: *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*. 2007, pp. 94–103. doi: [10.1109/FOCS.2007.41](https://doi.org/10.1109/FOCS.2007.41) (see p. 21).
- [OG03] Lawrence O’Gorman. “Comparing passwords, tokens, and biometrics for user authentication”. In: *Proceedings of the IEEE* 91.12 (Dec. 2003), pp. 2021–2040. doi: [10.1109/JPROC.2003.819611](https://doi.org/10.1109/JPROC.2003.819611) (see p. 68).
- [PT97] Dimitris Papadias and Yannis Theodoridis. “Spatial relations, minimum bounding rectangles, and spatial data structures”. In: *International Journal of Geographical Information Science* 11.2 (1997), pp. 111–138. doi: [10.1080/136588197242428](https://doi.org/10.1080/136588197242428) (see p. 86).
- [Rob82] Dorothy Elizabeth Robling Denning. *Cryptography and Data Security*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1982 (see p. 21).
- [San98] Ravi S Sandhu. “Role-Based Access Control”. In: *Advances in Computers* 46 (1998), pp. 237–286. doi: [10.1016/S0065-2458\(08\)60206-5](https://doi.org/10.1016/S0065-2458(08)60206-5) (see p. 22).
- [Sch05] Bruce Schneier. “Two-Factor Authentication: Too Little, Too Late”. In: *Communications of the ACM* 48.4 (2005), p. 136. doi: [10.1145/1053291.1053327](https://doi.org/10.1145/1053291.1053327) (see p. 69).
- [Sch08] Bruce Schneier. “Passwords Are Not Broken, but How We Choose them Sure Is”. In: *The Guardian* (Nov. 13, 2008). url: <http://www.>

- theguardian.com/technology/2008/nov/13/internet-passwords (see p. 68).
- [SGZ07] Jessica Staddon, Philippe Golle, and Bryce Zimny. “Web-Based Inference Detection”. In: *Proceedings of the 16th USENIX Security Symposium, Boston, MA, USA, August 6-10, 2007*. 2007 (see pp. 20, 34).
- [Sha49] Claude E Shannon. “Communication Theory of Secrecy Systems*”. In: *Bell system technical journal* 28.4 (1949), pp. 656–715. doi: [10.1002/j.1538-7305.1949.tb00928.x](https://doi.org/10.1002/j.1538-7305.1949.tb00928.x) (see p. 18).
- [SJ92] Ravi S Sandhu and Sushil Jajodia. “Polyinstantiation for cover stories”. In: *Computer Security ESORICS 92*. Springer, 1992, pp. 307–328. doi: [10.1007/BFb0013905](https://doi.org/10.1007/BFb0013905) (see pp. 34, 66).
- [SS94] Ravi S Sandhu and Pierangela Samarati. “Access control: principle and practice”. In: *Communications Magazine, IEEE* 32.9 (Sept. 1994), pp. 40–48. doi: [10.1109/35.312842](https://doi.org/10.1109/35.312842) (see p. 22).
- [Sta03] Jessica Staddon. “Dynamic inference control”. In: *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery, DMKD 2003, San Diego, California, USA, June 13, 2003*. 2003, pp. 94–100. doi: [10.1145/882082.882103](https://doi.org/10.1145/882082.882103) (see pp. 13, 18).
- [SVH12] Usama Salama, Vijay Varadharajan, and Michael Hitchens. “Metadata Based Forensic Analysis of Digital Information on the Web”. In: *Annual Symposium on Information Assurance & Secure Knowledge Management, June 5-6, 2012, Albany, NY, USA*. 2012, pp. 9–15 (see p. 19).
- [Swe00] Latanya Sweeney. “Simple demographics often identify people uniquely”. In: *Health (San Francisco)* 671 (2000), pp. 1–34 (see p. 20).

- [Swe02] Latanya Sweeney. “k-anonymity: A model for protecting privacy”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), pp. 557–570. doi: [10.1142/S0218488502001648](https://doi.org/10.1142/S0218488502001648) (see p. 20).
- [TFE05] Tyrone S Toland, Csilla Farkas, and Caroline M Eastman. “Dynamic Disclosure Monitor (D^2Mon): An Improved Query Processing Solution”. In: *Secure Data Management, Second VLDB Workshop, SDM 2005, Trondheim, Norway, September 2-3, 2005, Proceedings*. 2005, pp. 124–142. doi: [10.1007/11552338_9](https://doi.org/10.1007/11552338_9) (see p. 17).
- [TFE10] Tyrone S Toland, Csilla Farkas, and Caroline M Eastman. “The inference problem: Maintaining maximal availability in the presence of database updates”. In: *Computers & Security* 29.1 (2010), pp. 88–103. doi: [10.1016/j.cose.2009.07.004](https://doi.org/10.1016/j.cose.2009.07.004) (see p. 17).
- [van05] Henk C. A. van Tilborg. *Encyclopedia of Cryptography and Security*. Springer, 2005. doi: [10.1007/0-387-23483-7](https://doi.org/10.1007/0-387-23483-7) (see p. 22).
- [Var90] Vijay Varadharajan. “Petri net based modelling of information flow security requirements”. In: *Computer Security Foundations Workshop III, 1990. Proceedings*. June 1990, pp. 51–61. doi: [10.1109/CSFW.1990.128185](https://doi.org/10.1109/CSFW.1990.128185) (see p. 20).
- [Way+05] James Wayman et al. “An Introduction to Biometric Authentication Systems”. In: *Biometric Systems*. Springer London, 2005, pp. 1–20. doi: [10.1007/1-84628-064-8_1](https://doi.org/10.1007/1-84628-064-8_1) (see p. 68).
- [WB10] Philip Woodall and Pearl Brereton. “A Systematic Literature Review of Inference Strategies”. In: *Int. J. Inf. Comput. Secur.* 4.2 (Aug. 2010), pp. 99–117. doi: [10.1504/IJICS.2010.034813](https://doi.org/10.1504/IJICS.2010.034813) (see p. 17).
- [WTV01] Vlad Ingar Wietrzyk, Makoto Takizawa, and Vijay Varadharajan. “A Strategy for MLS Workflow”. In: *Information Security and Privacy, 6th*

- Australasian Conference, ACISP 2001, Sydney, Australia, July 11-13, 2001, Proceedings*. 2001, pp. 159–175. doi: [10.1007/3-540-47719-5_14](https://doi.org/10.1007/3-540-47719-5_14) (see p. 20).
- [XT06] Xiaokui Xiao and Yufei Tao. “Anatomy: Simple and Effective Privacy Preservation”. In: *Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006*. 2006, pp. 139–150 (see p. 20).
- [YL04] Xiaochun Yang and Chen Li. “Secure XML publishing without information leakage in the presence of data inference”. In: *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, Toronto, Canada, August 31 - September 3 2004*. 2004, pp. 96–107 (see p. 20).
- [YL98] Raymond W. Yip and E. N. Levitt. “Data level inference detection in database systems”. In: *Proceedings of the 11th IEEE Computer Security Foundations Workshop, Rockport, Massachusetts, USA, June 9-11, 1998*. 1998, pp. 179–189. doi: [10.1109/CSFW.1998.683168](https://doi.org/10.1109/CSFW.1998.683168) (see pp. 17, 34).
- [YTM07a] Ding Yixiang, Peng Tao, and Jiang Minghua. “Secure Multiple XML Documents Publishing without Information Leakage”. In: *Convergence Information Technology, International Conference on* (2007), pp. 2114–2119. doi: [10.1109/ICCIT.2007.345](https://doi.org/10.1109/ICCIT.2007.345) (see p. 20).
- [YTM07b] Ding Yixiang, Peng Tao, and Jiang Minghua. “Secure multiple xml documents publishing without information leakage”. In: *International Conference on Convergence Information Technology*. 2007, pp. 2114–2119. doi: [10.1109/ICCIT.2007.135](https://doi.org/10.1109/ICCIT.2007.135) (see p. 34).
- [Yu11] Haifeng Yu. “Sybil Defenses via Social Networks: A Tutorial and Survey”. In: *SIGACT News* 42.3 (Oct. 2011), pp. 80–101. doi: [10.1145/2034575.2034593](https://doi.org/10.1145/2034575.2034593) (see p. 69).

