Detecting misconfigurations in access-control systems

Sintesi

Candidato: Fabiana Marchetti
Relatore: Roberto Di Pietro, PhD
Correlatore: Alessandro Colantonio, PhD

Anno Accademico 2011/2012
Ottobre 2012

AMS classification: 68T10, 05C90, 05C50, 68N19
Key words: Access Control, Anomaly Detection, Misconfiguration, Binary Matrix
Introduction

In information system security, *access control* represents the process of mediating requests to data and services maintained by a system, and determining whether the requests should be granted or denied \[13\]. The access control system has the responsibility to ensure that only users with legitimate credentials are granted permissions to access requested resources.

Significant research has focused on providing formal representation of access control models. Among all proposed models *Role-Based Access Control* (RBAC) \[11\] is certainly the most adopted by organizations. The RBAC model introduces a framework for using *roles*; a role identifies the set of permissions required to perform particular system functions. Rather than assigning permissions directly to users, one introduces a set of roles and defines two kind of relations: role-to-permission relations (that assign permissions to roles) and user-to-role relations (that assign users to roles). Roles for various job functions are created first, then users are assigned to appropriate roles, based on their competencies and qualifications. Through those assignments each user acquires the permissions to perform his duties.

In an ideal RBAC environment, all users in the organization should have well-defined and precise roles, and resource access management should be done only via roles. In real world, very often the provisioning of resource access is still user-based instead of being purely role-based, and sometimes administrators and resource owners manually provide or revoke access on an as-needed and sometimes ad-hoc basis. This typically leads to incorrect access permissions, also referred to as *access control misconfigurations*. Misconfigurations usually fall into one of the following two categories:

- *missing values*: misconfigurations that arise when a user could legitimately have access to an object but administrators have not granted it yet;

- *outliers*: misconfigurations that arise when a user who should not have access to a certain resource according to organizational policy, does indeed have access.

The first class of misconfigurations is often the result of employees joining the
organization or joining a new project, and it represents both a business risk and security risk. Instead, the second class of incorrect access permissions is often the result of users changing job functions in an organization and represent a considerable security risk.

Access control misconfigurations are a special kind of the so called anomalies, in an access control context. Broadly speaking, anomalies are defined as patterns in the data that do not conform to a well defined notion of normal behaviour [1]. Anomalies usually translate to significant and often critical actionable information in a wide variety of application domains. Focusing on access control, the identification of misconfigurations allows administrators to fix them before they interfere with the use of the system. Access-control policy generally exhibits patterns across users and resources they access, owing to the use of groups and roles in the creation of policy. These patterns are evidenced in the accesses that are allowed in the system and, as Bauer et al. [6] demonstrated, they can be leveraged to predict misconfigurations and to infer what administrators should be consulted to resolve them.

Several methods can be found in the current literature to identify and fix misconfigurations. Kim et al. [7] proposed an imputation method (LL-Simpute) based on the least squares formulation to estimate missing values. Their work is focused on gene expression data; each target gene that has missing values is represented as a linear combination of similar genes, and rather than using all available genes in the data, it selects only $k$ similar genes based on a similarity measure that can be the L2-norm or the Pearson correlation coefficients.

Binary matrix decomposition and rank reduction are well-known machine learning techniques that have been also used in access control to identify misconfigurations [15, 2]. In particular, Molloy et al. [9] described how to apply those techniques in a RBAC system: given $n$ users and $m$ permissions, they model the access control system with a matrix $M \in \{0, 1\}^{n \times m}$. Such a matrix $M$ can be decomposed into two matrices $A \in \mathbb{R}^{n \times k}$ and $B \in \mathbb{R}^{m \times k}$ such that $M \approx AB^T$, that is the distance between $M$ and $AB^T$ is minimized. The rank of $AB^T$ will equal $k$, where $k \leq \min(m, n)$. In particular, if $k < \min(n, m)$, then the matrix $AB^T$ has a rank lower than $M$. Intuitively, this means that $AB^T$ has found information in $M$ that are structured and can be explained.
by some more succinct model. Hence, the differences between $M$ and $AB^T$ identifies assignments in $M$ that may correspond to noise. There are several factorization models that can be used to decompose a binary matrix. Among them, those that produce results more suitable to be interpreted in the context of RBAC are the singular value decomposition (SVD), the non-negative matrix factorization (NNMF) and the bayesian non-negative matrix factorization (BNMF).

Apart from least-squares-based and matrix-decomposition-based methods, there is another class of anomaly detection algorithms that is applicable to a general context. In particular, distance-based anomaly detection algorithms have emerged as a viable and scalable alternative to detect anomalies. A distance-based anomaly in a dataset is a data object with a given percentage of the objects in the dataset having a distance greater than $d_{\text{min}}$ away from it. The definition of distance can be extended by considering the distance of a point from its $k$-nearest neighbour \cite{12,10}, or as the sum of distances from its $k$-nearest neighbours \cite{3}. However, in the current literature there is no application of such a class of algorithms to identify misconfigurations in access control systems.

The aim of this thesis is thus to apply well-known distance-based anomaly detection algorithms to access control. The problem is that most of distance-based algorithms consider the data to be analysed as points in $n$-dimensional space, where $n$ is the number of features used to describe each data. In particular, they use a different distance depending on the nature of these attributes. Instead, a natural representation for user-permission assignments is a binary matrix, where rows and columns correspond to users and permissions, respectively, and each cell is “on” when a certain user has a certain permission granted \cite{5}. Each cell of this matrix should be considered as a single data: in fact, relationships between both rows and columns have particular meaning in access control, and by considering rows or columns only would lead to information loss. For instance, if one handled each row or each column as a single data point, then a single data point marked as outlier would mean that such user or permission should not exist at all. This does not make sense. What we want to do is to find which individual user should not have access to which single resource, namely which single cell should be
labelled as outlier. The same can be said for missing values.

A further contribution of this thesis is represented by the introduction of a novel distance on binary matrices that is particularly meaningful in an access control context. As a matter of fact, in order to apply existing distance-based algorithms to binary matrices, first we need to define which properties two cells of the matrix should have in order to be considered close in the access control framework. However, there is no classic distance defined between cells of binary matrices that also has meaning in the access control context. Indeed, since each cell represents a particular permission granted to a user, we would assign a short distance to cells that represent user-permission pairs that can be managed through the same role. Based on the meaning of an access control matrix, we then defined a measure for the closeness of matrix cells.

Finally, experimental results on extended synthetic and real datasets confirmed the viability of our algorithms; the quality of these results has been analyzed and displayed through the use of ROC graphs.

**Mapping binary matrices to unipartite graphs**

In this thesis we propose a novel distance, based on a known mapping between graph theory and the access control problem, that allows us to apply well-known methods to binary matrices. In fact the binary matrix, that represents user-permission assignments, can be expressed through the use of a bipartite graph $B = \langle \text{USERS} \cup \text{PERMS}, \text{UP} \rangle$, where two vertices $u \in \text{USERS}$ and $p \in \text{PERMS}$ are connected by an edge if the user $u$ is granted permission $p$, namely $\langle u, p \rangle \in \text{UP}$. Every biclique identifies a role, and the vertices of the biclique identify the users and the permissions assigned to this role. Starting from this bipartite graph, it is possible to construct an indirect unipartite graph $G$, where the edges of $B$ become the vertices of $G$ and two vertices in $G$ are connected by an edge if and only if the endpoints of the corresponding edges of $B$ induce a biclique in $B$ [4]. Hence, the undirected unipartite graph $G$ induced from $\text{UP}$ can be defined as:

$$G = \langle \text{UP}, \{\langle \omega_1, \omega_2 \rangle \in \text{UP} \times \text{UP} \mid \omega_1, \omega_2 \text{ induce a biclique in } B \} \rangle.$$
It can be proved that any clique partition of $G$ corresponds to a biclique cover of $B$. In other words, each biclique in the bipartite graph $B$ corresponds to a clique in the unipartite graph $G$, and vice-versa. Since the role concept in access control corresponds to the biclique concept in bipartite graphs, any partition of $G$ can be also viewed as a possible solution for the access control problem represented by the sets $\text{USERS, UA, and PA}$.

**A new distance for binary matrices**

In order to obtain a new distance for the binary matrix which is also consistent with the framework of access control, we first define a distance on the unipartite graph $G$. Then starting from the unipartite graph, we can study how the proposed distance is translated in terms of binary matrix, hence defining a new distance directly on binary matrices.

On the indirect unipartite graph $G$ we use the classical distance defined between vertices in a graph:

**Definition 1 (Distance on the unipartite graph).** Taken two vertices $\omega_1 = \langle u_1, p_1 \rangle$, $\omega_2 = \langle u_2, p_2 \rangle$ of the unipartite graph $G$, the distance between $\omega_1$ and $\omega_2$ is defined as the length of the shortest path between them.

If there is an edge between two vertices, then we can consider these vertices close according to the access control context: in fact the edges of the unipartite graph are constructed from the bicliques present in the bipartite graph, where every biclique identifies a role. Hence, the two vertices are related since they belong to the same role.

Now, we first consider the vertices at distance one on the unipartite graph and we examine which cells of the binary matrix correspond to them. Since we will show that the property that link such cells is the belonging to the maximal pseudo-bicluster generated by a same cell of the binary matrix, first we introduce the following definitions:

**Definition 2 (Pseudo-Bicluster).** Given a matrix $M$, a pseudo-bicluster $B$ is a pair $\langle R, C \rangle : R \subseteq [n], C \subseteq [m]$ that has at least one row and one column filled by 1’s, formally:

$$\exists i \in R, \exists j \in C, \forall l \in R, \forall k \in C : m_{ik} = 1, m_{lj} = 1.$$
**Definition 3 (Maximal Pseudo-Bicluster).** Let $B = (R, C)$ be a pseudo-bicluster in the matrix $M$. It is also a maximal pseudo-bicluster if:

$$\nexists \text{ a pseudo-bicluster } B' = (R', C') : R \times C \subset R' \times C'.$$

A maximal pseudo-bicluster is a pseudo-bicluster such that its rows and columns filled by 1’s cannot be “expanded” by adding columns and rows.

**Definition 4 (Maximal Pseudo-Bicluster Generator).** An element $m_{ij} \in M$ is referred to as a generator of the maximal pseudo-bicluster $B = (R, C)$ if $R = \{l \in [n] \mid m_{ij} = 1\}$ and $C = \{k \in [m] \mid m_{ik} = 1\}$. If $m_{ij} \in M$ is a generator of $B$, we will also say that $B$ is the maximal pseudo-bicluster generated by $m_{ij}$, and we indicate it as $B_{m_{ij}}$.

The results obtained by analysing the cells of the binary matrix that correspond to the vertices at distance one on the unipartite graph, can be summarized in the following lemmas. Note that hereafter, unless otherwise stated, when we talk about a cell of the binary matrix we always mean a cell set to one.

**Lemma 5.** Let $m_{u_1p_1}$ and $m_{u_2p_2}$ two cells of the binary matrix that represent the access control system and $\omega_1 = (u_1, p_1)$, $\omega_2 = (u_2, p_2)$ the corresponding vertices in the unipartite graph. Let $d$ the distance defined on the graph. Then $d(\omega_1, \omega_2) = 1 \iff m_{u_1p_1} \in \text{maximal pseudo-bicluster generated by } m_{u_2p_2} \iff m_{u_2p_2} \in \text{maximal pseudo-bicluster generated by } m_{u_1p_1}$.

Hence, to maintain the correspondence between the binary matrix and the unipartite graph we want the new distance $D$ satisfies the following property:

**Property 6.** Let $m_{u_1p_1}$ and $m_{u_2p_2}$ two cells of the binary matrix and $D$ the distance defined on the binary matrix. Then $D(m_{u_1p_1}, m_{u_2p_2}) = 1 \iff m_{u_1p_1} \in \text{maximal pseudo-bicluster generated by } m_{u_2p_2} \iff m_{u_2p_2} \in \text{maximal pseudo-bicluster generated by } m_{u_1p_1}$.

In particular, a distance that satisfies this property then it satisfies the hypotheses of the following lemma:
Lemma 7. Let $m_{u_1p_1}$ a cell of the binary matrix and $D$ a distance such that $D(m_{u_1p_1}, m_{u_2p_2}) = 1 \iff m_{u_1p_1} \in B_{m_{u_2p_2}} \iff m_{u_2p_2} \in B_{m_{u_1p_1}}$. Then the cells that have distance equal to 1 from $m_{u_1p_1}$ are all and only the cells $m_{u_ip_i}$ such that $m_{u_ip_i} \in$ maximal pseudo-bicluster generated by $m_{u_1p_1}$, formally:

$$\{m_{u_ip_i} \in M \mid D(m_{u_1p_1}, m_{u_ip_i}) = 1\} = \{m_{u_ip_i} \in M \mid m_{u_ip_i} \in B_{m_{u_1p_1}}\}.$$

Hence, we can easily move from the binary matrix to the unipartite graph, constructed as described above, and vice versa simply using the maximal pseudo-biclusters generated by each cell set to 1 of the matrix. Using this correspondence we can generalize the results obtained to the case of vertices at distance $d$, and so we can define the distance between cells of the binary matrix that has meaning in the access control context.

We first introduce some definitions that will help us to express this new distance.

Definition 8 (Path on the binary matrix). Let $m_1, m_2, \ldots, m_n$ cells of the binary matrix. A path $p_{x,y}$ from the cell $x$ to the cell $y$ is an ordered set of cells $p_{x,y} = \{m_1, m_2, \ldots, m_n\}$ such that $x = m_1$, $y = m_n$ and $m_i \in B_{m_{i-1}}$ for all $i = 2, 3, \ldots, n$, that is each cell of $p_{x,y}$ belong to the maximal pseudo-bicluster generated by the previous cell in the sequence.

We say that two cells are unreachable if there is not a path between them and we only consider paths with no repeated cells.

Definition 9 (Length of a path). Let $p_{x,y} = \{m_1, m_2, \ldots, m_n\}$ a path from $x$ to $y$. The length of $p_{x,y}$ is defined as $l_{p_{x,y}} = |p_{x,y}| - 1$, where $|\cdot|$ is the cardinality of the set.

After having provided Definition 8 and Definition 9 we can finally define the new distance as follows.

Definition 10 (Distance on the binary matrix). Let $x$ and $y$ two cells of the binary matrix and let $P$ the set of all the possible path from $x$ to $y$. The distance between $x$ and $y$ is defined as the shortest path from $x$ to $y$, that is

$$D(x, y) = \min_{p_{x,y} \in P} l_{p_{x,y}} = |\overline{p}_{x,y}| - 1$$

where $\overline{p}_{x,y}$ denote the shortest path from $x$ to $y$. 7
The correctness of Definition 10 is proved with the following lemma.

**Lemma 11.** The distance $D$ in Definition 10 is well defined.

**Applications to nearest neighbour algorithms**

We can now explain how to replace the euclidean distance in $\mathbb{R}^n$ used by most of distance-based methods with the new distance proposed in this thesis. In this way we are able to apply these methods on binary matrices. In particular, we focus on the nearest neighbour based anomaly detection techniques which use as anomaly score the distance of each data instance from its $k^{th}$ nearest neighbour in a given data set.

The main difference in applying distance based algorithms on binary matrices is that the distance between a point $x$ and a point $y$ is a value $\alpha \in \mathbb{N}$ instead of a value $\beta \in \mathbb{R}$, $\beta \geq 0$. Hence, we can think to the neighbourhood of a point $x$ as a series of tiers or circumferences centred in $x$ and having radius in $\mathbb{N}$; on the circumference of radius $r = 1$ there are all the points $p_i$ such that $p_i \in B_x$. On the circumference of radius $r = 2$ there are the points $p'_i \in B_{p_i}, \forall p_i$ such that $p_i \in B_x$; that is on the second circumference there are the neighbours of the neighbours, all the points at distance 1 from all the points that belong to the pseudo-bicluster generated by $x$. On the circumference of radius $r = 3$ there are the neighbours of the neighbours of the neighbours of $x$ and so on.

Hence, to use nearest neighbour based techniques we have to find for each data instance its $k$ nearest neighbours and its distance from the farthest, the $k^{th}$. Using our novel approach this task can be easily accomplished repeating the following proceeding for each cell $m$ of the matrix that is set to 1.

In the following procedure we will denote with $B_\mu$ the maximal pseudo-bicluster generated by cell $\mu$, $\tilde{B}_\mu = \{m_{ij} \in B_\mu \mid m_{ij} = 1\}$. Moreover, we denote $\tilde{C}_k = \bigcup_\mu \tilde{B}_\mu$ and $C_k = \tilde{C}_k \cup \tilde{C}_{k-1} \cup \cdots \cup \tilde{C}_1$. If a cell $\mu \in C_l$ and also $\mu \in C_k$, with $l < k$, then we consider this cell only in the previous step, thus $\mu \in C_l$. Hence, we do not consider the same cell twice.

**Step 0** We calculate the maximal pseudo-bicluster generated by the cell $m$, thus $B_m$. Let $\tilde{B}_m = \{m_{ij} \in B_m \mid m_{ij} = 1\} = \tilde{C}_1$ and let $C_1 = \tilde{C}_1$. 

8
Step 1 If $|C_1| \geq k \implies D(m, k^{th} \text{point}) = 1$ and we can randomly choose the $k$ nearest neighbour of $m$ in $\tilde{C}_1$.
Otherwise $\implies$ Step 2.

Step 2 We calculate $B_\alpha, \forall \alpha \in \tilde{C}_1$. We denote with $\tilde{C}_2 = \cup_\alpha \tilde{B}_\alpha$ and $C_2 = \tilde{C}_2 \cup \tilde{C}_1$.
If $|C_2| \geq k \implies D(m, k^{th} \text{point}) = 2$ and we take all the points in $C_1$ and we randomly choose the remaining $k - |C_1|$ nearest neighbour of $m$ in $\tilde{C}_2$.
Otherwise $\implies$ Step 3.

Step 3 We calculate $B_\beta, \forall \beta \in \tilde{C}_2$. We denote with $\tilde{C}_3 = \cup_\beta \tilde{B}_\beta$ and $C_3 = \tilde{C}_3 \cup C_2$.
If $|C_3| \geq k \implies D(m, k^{th} \text{point}) = 3$ and we take all the points in $C_2$ and we randomly choose the remaining $k - |C_2|$ nearest neighbour of $m$ in $\tilde{C}_3$.
Otherwise $\implies$ Step 4.

Step j We calculate $B_\xi, \forall \xi \in \tilde{C}_{j-1}$. We denote with $\tilde{C}_j = \cup_\xi \tilde{B}_\xi$ and $C_j = \tilde{C}_j \cup C_{j-1}$.
If $|C_j| \geq k \implies D(m, k^{th} \text{point}) = j$ and we take all the points in $C_{j-1}$ and we randomly choose the remaining $k - |C_{j-1}|$ nearest neighbour of $m$ in $\tilde{C}_j$.
Otherwise $\implies$ Step $j + 1$.

Applications to two well-known algorithms

Now, we summarize two existing algorithms that are based on the concept of the nearest neighbour analysis, that is LOF (Local Outlier Factor)\cite{14} and ODIN (Outlier Detection using In-degree Number)\cite{8}. We show how our approach can be applied to these two particular cases and we present two new algorithms inspired by these two methods that use the new distance introduced to detect anomalies and such that they can work with binary matrices. Both these algorithms have been developed to deal with only one kind of misconfiguration, the outliers.
LOF (*Local Outlier Factor*) Breunig at al.\[14\] propose LOF, a density-based outlier detection method for finding outliers in a multi-dimensional dataset. Usually density-based techniques perform poorly if the dataset has regions of varying densities; instead this approach is able to handle this issue by taking into consideration, for any object, the object’s relative density as compared to those of its nearest neighbours. Breunig at al.\[14\] assign a factor called *Local Outlier Factor* to each object in the dataset; this quantity indicates the object’s degree of outlierness. This outlier factor is local in the sense that only a restricted neighbourhood of each object is taken into account.

The LOF of an object is based on the single parameter of $k$, which is the number of nearest neighbours used in defining the local neighbourhood of the object. To calculate this factor, the authors first define the notion of *k-distance* of an object $p$, of *k-distance neighbourhood* of $p$ and the reachability distance of an object $p$ with respect to an object $o$.

**Definition 12 (k-distance of an object $p$).** For any positive integer $k$, the *k-distance* of object $p$, denoted as $k$–*distance*($p$), is defined as the distance $d(p,o)$ between $p$ and its $k^{th}$ nearest neighbour $o \in D$.

**Definition 13 (k-distance neighborhood of an object $p$).** Given the *k-distance* of $p$, the *k-distance neighbourhood* of $p$, denoted as $N_k(p)$, contains every object whose distance from $p$ is not greater than the *k-distance*, i.e.

$$N_{k-distance}(p)(p) = \{q \in D \setminus \{p\} \mid d(p,q) \leq k-distance(p)\}.$$ 

These objects $q$ are called the *k*-nearest neighbours of $p$.

**Definition 14 (reachability distance of an object $p$ w.r.t object $o$).** Let $k$ be a natural number. The *reachability distance* of object $p$ with respect to object $o$ is defined as $reach – dist_k(p,o) = max\{k-distance(o),d(p,o)\}$

To detect density-based outliers, however, it is necessary to compare the densities of different sets of objects. Therefore Breunig at al. use the parameter $k$ that specify the minimum number of objects considered and they use the values $reach – dist_k(p,o)$, for $o \in N_k(p)$, as a measure of the volume to determine the density in the neighbourhood of an object $p$. 

10
Definition 15 (local reachability density of an object $p$). The local reachability density of $p$ is defined as

$$lrd_k(p) = \frac{1}{\left(\sum_{o \in N_k(p)} \text{reach} - \text{dist}_k(p,o)\right)}$$

Definition 16 ((local) outlier factor of an object $p$). The (local) outlier factor of $p$ is defined as

$$LOF_k(p) = \frac{\sum_{o \in N_k(p)} lrd_k(o)}{|N_k(p)|}$$

For a normal instance lying in a dense region, its local density will be similar to that of its neighbours, while for an anomalous instance, its local density will be lower than that of its nearest neighbours. Hence the anomalous instance will get a higher LOF score.

We revisit the definitions proposed by Breunig at al.\cite{14} in such a way that is possible to use them on a binary matrices with the new distance defined in this thesis. We obtain the following definitions:

Definition 17 (neighbour of a cell). Let $m_{ij}$ a cell of a binary matrix $M$. We say that another cell $m_{st} \in M$ is a neighbour of $m_{ij}$ if and only if exists a path from $m_{ij}$ to $m_{st}$.

Definition 18 (k-distance of a cell). For any positive integer $k$, the $k$-distance of a cell $m_{ij} \in M$, denoted as $k - \text{distance}(m_{ij})$, is defined as the distance $D(m_{ij}, m_{st}) = \min_{p_{m_{ij}, m_{st}} \in P_{m_{ij}, m_{st}}} l_{p_{m_{ij}, m_{st}}}$ between $m_{ij}$ and its $k$th nearest neighbour $m_{st} \in M$.

Definition 19 (k-distance neighbourhood of a cell). Given the $k$-distance of $m_{ij}$, the $k$-distance neighbourhood of $m_{ij}$, denoted as $N_k(m_{ij})$, contains every cell whose distance from $m_{ij}$ is not greater than the $k$-distance. These objects $m_{st}$ are called the $k$-nearest neighbours of $m_{ij}$.

Replacing these definitions in Definition 14-16 we have:

Definition 20 ((local) outlier factor of a cell $m_{ij}$). The (local) outlier factor of $m_{ij}$ is defined as

$$LOF_k(m_{ij}) = \frac{\sum_{o \in N_k(m_{ij})} lrd_k(m_{st})}{|N_k(m_{ij})|}$$
The algorithm that we obtain is the *LofModified* algorithm. It has two control parameters: \( k \), the number of neighbours that we want to find for each cell \( m_{ij} \), and a threshold \( t \), which represents the minimum LOF value that each cell must have. To save the LOF value of all the cells, we use a matrix that we denote with \( \text{result} \); each cell \( r_{ij} \in \text{result} \) contains the LOF value of the cell \( m_{ij} \) of the binary matrix. To produce this kind of matrix we can use the `computeLOFModified` method. The idea is that for every cell \( m_{ij} \) of the binary matrix we first find its \( k \)-nearest neighbours using the new distance defined in this thesis; then we calculate its LOF value with Definition 20 and we put this value in the corresponding cell \( r_{ij} \). After having constructed the \( \text{result} \) matrix in this way, we check if the value contained in each cell \( r_{ij} \) is \( t \) or more; in this case the corresponding cell \( m_{ij} \) is marked as an outlier, otherwise the cell is marked as an inlier.

**ODIN (Outlier Detection using In-degree Number)** Hautamaki et al.\cite{Hautamaki} propose a simpler version of LOF in which a quantity called ODIN is calculated for each data instance.

Their method is a density-based outlier detection approach that use the \( k \)-nearest neighbour (KNN) graphs. Hautamaki et al. consider the KNN graph as a weighted directed graph in which every vertex represents a single vector, and each edge corresponds to pointers to neighbour vectors. Every vertex has exactly \( k \) edges to the \( k \) nearest vectors according to a given distance function. The weight of the edge \( e_{i,j} \) is defined as the distance between vectors \( e_i \) and \( e_j \).

Their definition of outlier is based on the *indegree number* of each vertex in the KNN graph, that is a vertex is considered as an outlier if it appear in less than \( T \) neighbourhoods. Formally:

**Definition 21 (outlier).** Given a KNN graph \( G \) for dataset \( D \), an outlier is a vertex whose indegree number is less than equal to \( T \).

The algorithm proposed by Hautamaki et al. takes as input two parameters: the number of the outgoing edges \( k \) and the indegree threshold \( T \). The algorithm can be summarized as follows: taken as input a dataset \( D \), the KNN graph is first created. Then, for each vertex \( i \) is checked its indegree
number: if it is \( T \) or less the vector is marked as an outlier, otherwise the vector is marked as an inlier.

To show how to apply this algorithm to binary matrices, we first introduce some definitions to help explain the new approach.

**Definition 22 (neighbour of a cell).** Let \( m_{ij} \) a cell of a binary matrix \( M \). We say that a cell \( m_{st} \in M \) is a *neighbour* of \( m_{ij} \) if and only if exists a path from \( m_{ij} \) to \( m_{st} \).

**Definition 23 (k-neighbour of a cell).** A cell \( m_{st} \in M \) is a *k-neighbour* of \( m_{ij} \) if \( m_{st} \) is one of the \( k \)-nearest neighbours of \( m_{ij} \).

The algorithm that we obtain is the *OdinModified* algorithm. It has the same two control parameters: \( k \), the number of neighbours that we want to find for each cell \( m_{ij} \), and a threshold \( t \), which represents the minimum number of times that each cell must be a \( k \)-neighbour of another cell. Instead of using a KNN graph, we use a matrix, that we denote with *result*, in which each cell \( r_{ij} \) contains the number of times that the cell \( m_{ij} \) of the binary matrix is a \( k \)-neighbour. To produce this kind of matrix we can use the *computeOdinModified* method. The idea is that for every cell \( m_{ij} \) of the binary matrix we find its \( k \)-nearest neighbours using the new distance proposed in this thesis; for each neighbour \( m_{st} \) that we find, we increase of one \( r_{st} \). After having constructed the *result* matrix in this way, we check if the value contained in each cell \( r_{ij} \) is \( t \) or less; in this case the corresponding cell \( m_{ij} \) is marked as an outlier, otherwise the cell is marked as an inlier.

**A novel approach to impute missing values and outliers in binary matrices**

We present a new algorithm referred to as ABBA (*An Adaptive Bicluster-Based Approach*) that is able to impute both missing values and outliers in binary matrices. Compared to the other methods presented, this algorithm has only one parameter to be set, and the misconfigurations found need not to be recomputed from scratch as the parameter changes.
The analysis behind this novel approach is mainly based upon two considerations. First, the more a maximal pseudo-bicluster \( B \) is close to being a maximal bicluster (namely, it is “almost” filled by 1’s), the more the pattern represented by \( B \) correctly describes the available data. The second consideration is that the more a maximal pseudo-bicluster \( B \) has a large area (that is, a large number of involved cells), the more it should be considered a significant pattern, and subsequently it should have a higher relevance. Both considerations represent a possible way to select good “candidate” patterns to be used to infer anomalies. The following index captures both the above described points:

**Definition 24 (Relevance of a Maximal Pseudo-Bicluster).** Let \( \text{MPBS} \) be the set of all maximal pseudo-bicluster existing in \( M \), and \( B \in \text{MPBS} \). The relevance of \( B \) is defined as:

\[
\varrho(B) = \left| \{ m_{ij} \in M \mid m_{ij} \text{ is a generator of } B \} \right| = \sum_{m_{ij} \in B} [m_{ij} \text{ generates } B],
\]

where the operator \([\cdot]\) maps a boolean condition \( b \) to \( \{0, 1\} \), namely:

\[
[b] = \begin{cases} 
1 & b = \text{true} \\
0 & \text{otherwise}
\end{cases}
\]

We introduce another relevance index for each element \( m_{ij} \in M \) that is based on the relevance of maximal pseudo-biclusters. By using this index we are able to evaluate misconfigurations \( m_{ij} \) by leveraging all the patterns represented by the maximal pseudo-bicluster which \( m_{ij} \) belongs to. In this way, each misconfiguration is imputed considering all and only those patterns that could involve it.

**Definition 25 (Relevance of a Cell).** Given an element \( m_{ij} \in M \), and let \( \text{MPBS} \) be the set of all existing maximal pseudo-biclusters within the data, the relevance of \( m_{ij} \) is the sum of the indices \( \varrho(B) \) of all the maximal pseudo-biclusters \( B \) which \( m_{ij} \) belongs to, normalized by \( |M| \). Formally:

\[
\sigma(m_{ij}) = \sum_{B \in \text{MPBS}: m_{ij} \in B} \varrho(B) = \sum_{m_{lk} \in M: m_{lk} = 1} [m_{ij} \in B(m_{lk})].
\]
Hence, ABBA leverages the pseudo-bicluster concept in order to assign a relevance to the cells of a binary matrix. Then the algorithm uses this relevance to identify both missing values and outliers; high relevances for cells containing 0’s are typical of missing values, low relevances for cells containing 1’s are typical of outliers.

The idea behind the update operation can be explained as follows. ABBA uses all the maximal pseudo-biclusters generated by each cell \( m \in M \) to impute misconfigurations and the maximal pseudo-biclusters are generated starting from all the cells filled by 1. We only have two possible modifications of the original binary matrix, that is when some 0’s is switched to 1’s, and/or some 1’s is switched to 0’s. The result of these two modifications are presented in the following theorems, respectively.

**Theorem 26.** By switching the value of a cell \( m_{st} \in M \) from 0 to 1, the relevance of a cell \( m_{ij} \in M \) will increase by the quantity

\[
[m_{ij}^{(1)} \in B(m_{st}^{(1)})] + [j = t] \left( \sum_{m_{sk}^{(0)} \in m_{s}^{*} \setminus \{m_{st}\}, m_{sk}^{(0)} = 1} [m_{ik}^{(0)} = 1] \right) + [i = s] \left( \sum_{m_{lt}^{(0)} \in m_{t}^{*} \setminus \{m_{st}\}, m_{lt}^{(0)} = 1} [m_{lj}^{(0)} = 1] \right)
\]

where the exponents (0) and (1) indicate respectively the configuration of the matrix before and after the switch.

**Theorem 27.** By switching the value of a cell \( m_{st} \in M \) from 1 to 0, the relevance of a cell \( m_{ij} \in M \) will decrease by the quantity

\[
[m_{ij}^{(0)} \in B(m_{st}^{(0)})] + [j = t] \left( \sum_{m_{sk}^{(0)} \in m_{s}^{*} \setminus \{m_{st}\}, m_{sk}^{(0)} = 1} [m_{ik}^{(0)} = 1] \right) + [i = s] \left( \sum_{m_{lt}^{(0)} \in m_{t}^{*} \setminus \{m_{st}\}, m_{lt}^{(0)} = 1} [m_{lj}^{(0)} = 1] \right)
\]
Evaluation

Finally, we compare the viability of the new algorithms proposed in this thesis with four different methods based on rank reduced matrix factorization and on least squares formulation. For the completeness of the results, the comparison is made both in supervised mode, by generating several artificial datasets, and in unsupervised mode, that is on real datasets. The quality of the results is also displayed through the use of ROC graphs.

Experimental results confirmed the viability of our algorithms. Comparing ourselves with the LLSimpute algorithm we have seen that the best approach to deal with user-permission matrices is to consider each cell as a single data, that is the approach used by each of our proposed algorithms. The comparison against the matrix decomposition and rank reduction methods has emphasized the superiority of ABBA in the missing values detection. The same comparison for the outliers detection has shown that, despite the other methods achieve better performances, our algorithms always obtain good results. The more the dataset considered reproduces a realistic RBAC configuration, the more the quality of results improve. This demonstrates the validity of our algorithms for the outliers detection in a real context of access control. Finally, tests on real datasets have pointed out the flexibility of our algorithms compared to the other methods; for instance, ABBA, Lof-Modified and OdinModified allow us to identify the exact number of desired outliers in any datasets.
References


