Informed Privilege-Complexity Trade-Offs in RBAC Configuration

Jon Currey
HashiCorp Inc.
jc@hashicorp.com

Robbie McKinstry
HashiCorp Inc.
robbie@hashicorp.com

Armon Dadgar
HashiCorp Inc.
armon@hashicorp.com

Mark Gritter
HashiCorp Inc.
mgritter@hashicorp.com

ABSTRACT

Role-Based Access Control (RBAC) has the potential both to simplify administration and improve an organization's security. But for non-trivial configurations, there is a conflict between defining fine-grained roles which adhere to the principle of least privilege, and coarse-grained roles which simplify administration by reducing configuration complexity.

In this paper we propose OnPar, a multi-objective role mining approach which introduces *minimization of unnecessary privilege* as a role mining objective, along with an associated *unnecessary privilege metric*. These allow an RBAC configuration's level of adherence to the principle of least privilege to be reasoned about and traded off against other objectives, including minimization of configuration complexity.

A key feature of our approach is the elimination of user tuning of global optimization weights. We show experimentally that this tuning typically leads to the evaluation of sub-optimal candidates, while still missing many optimal candidates. To avoid these issues we leverage *Pareto optimality* and introduce *multi-stage Pareto filtering* and the *hypervolume indicator* to role mining. Their use allows OnPar to efficiently select a small set of candidates for evaluation by the administrator, which are equal best and representative of the full range of trade-offs that were found.

Our experimental results demonstrate the effectiveness of this approach across a wide range of input configurations.

CCS CONCEPTS

• **Security and privacy** → **Information accountability and usage control**; **Access control**; • **Information systems** → **Data mining**.

KEYWORDS

RBAC, role engineering, role mining, Pareto optimality, multi-objective optimization, clustering

ACM Reference Format:

Jon Currey, Robbie McKinstry, Armon Dadgar, and Mark Gritter. 2020. Informed Privilege-Complexity Trade-Offs in RBAC Configuration. In *Proceedings of the 25th ACM Symposium on Access Control Models and Technologies (SACMAT '20)*, June 10–12, 2020, Barcelona, Spain. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3381991.3395597>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

SACMAT '20, June 10–12, 2020, Barcelona, Spain

© 2020 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-7568-9/20/06.

<https://doi.org/10.1145/3381991.3395597>

Figure 1: OnPar allows administrators to explore trade-offs between reduction in unnecessary privilege and configuration complexity, via small sets of candidates that are equal best and representative of the full range of trade-offs that were found.

1 INTRODUCTION

Role-Based Access Control (RBAC) [10] [35] has the potential both to simplify administration and improve an organization's security. Assigning privilege to users via roles, instead of directly, aligns with the *principle of least privilege* [36], which requires that they be given no more privilege than is necessary to perform their duties [9]. It also offers a way to reduce configuration complexity, since there are typically far fewer roles than users [42].

These benefits are not guaranteed, and RBAC configuration administration is a critical task. A configuration must be designed well initially and then actively maintained to avoid over-permissioned users and redundant or overly-permissive roles [3]. Role mining tools [22] assist the administration process by examining the current system state and generating candidate roles or full RBAC configurations for consideration by an administrator. A role mining tool must be guided by some objective, and many different objectives have been identified. *Minimization of configuration complexity* is by far the most popular objective, and for good reason: A less complex configuration is easier to approve, change, and audit, and can reduce administration overhead and error [22] [47] [4].

However, a conflict exists between reduction of RBAC configuration complexity and privilege. In practice, simple configurations tend to have "coarse-grained" roles encompassing a wide swath of business functions. But roles with broad scope will be less precise,

authorizing users for more business functions than they need, leading to a violation of least privilege. Conversely, fine-grained role assignment enables closer adherence to least privilege, but results in a more complex configuration.

Administrators must generate roles which reasonably trade off overall configuration complexity and least privilege, negotiating these conflicting objectives while respecting organizational constraints and ensuring continuity of necessary privilege.

Usage information is uniquely placed to act as an indicator of necessary privilege [3], and some role mining approaches have incorporated it [13] [33] [49]. But in each case, the usage information is employed to improve performance of other objectives. This leads us to define a new role mining objective that directly supports reasoning about least privilege, and to explore its use in a multi-objective role mining approach, traded against minimization of configuration complexity.

However, in the process of building an end-to-end solution for use by administrators, we encountered issues around user-tuning of the weighting coefficient that is typically used to control the balance of consideration given to two objectives. Solving these challenges leads us to propose a new overall approach to multi-objective role mining.

1.1 Contributions

In this paper we propose OnPar, a new role mining approach that makes the following contributions:

We introduce *minimization of unnecessary privilege* as a new formal objective for role mining, along with an associated *unnecessary privilege metric*. Together they allow an RBAC configuration's level of adherence to the principle of least privilege to be measured and reasoned about directly for the first time.

We apply *Pareto filtering* in order to eliminate the user-tuning of a weighting coefficient in a way that avoids presenting additional, sub-optimal candidates to the administrator.

We introduce *multi-stage Pareto filtering* and the *hyper-volume indicator* to role mining, to sample the Pareto optimal candidates in a representative manner. This allows the number of choices offered for evaluation to be reduced significantly, with negligible reduction in the ability to convey the range of trade-off alternatives available.

We use our implementation of OnPar to evaluate this approach. Our experimental results show that it is effective across a broad range of system configurations.

The rest of the paper is structured as follows. Section 2 explains the motivation for the approach taken. Section 3 discusses related work. Section 4 presents the OnPar design. Section 5 details our empirical evaluation of the approach. In Section 6, conclusions are drawn, and limitations and future work are discussed.

2 MOTIVATION

We built OnPar to support our customers, who are users of HashiCorp Vault [16], an open-source based [15] commercial access control solution. In this section we discuss the requirements that led to the new role mining techniques that we propose, most of which have broad applicability.

2.1 Privilege Minimization As An Explicit Goal

Businesses are increasingly moving from traditional physical plus perimeter security to *identity-based security*, where users must re-authenticate periodically to maintain use of the privileges associated with their account. This model is essential for public cloud deployment, where physical networks and servers are out of the control of tenants. It is also becoming best practice for private datacenters and on-premises IT [45], due to increasingly sophisticated hacking and ransomware, punitive fines for disclosure of customers' personal information, and workforce mobility.

In this setting, the principle of least privilege becomes a business imperative, since every privilege associated with an account is a potential attack vector if the account is compromised or its owner becomes a bad actor. All else being equal, security administrators have the objective of eliminating unnecessary privilege.

Defining formal objectives is a well established technique in role mining [1] [14], and in Section 3.1 we review the wide variety of objectives that have been explored. In addition, RBAC's ability to assist in enforcing the principle of least privilege was one of the factors that drove its widespread adoption and standardization [9]. It was therefore surprising to us when we could not find a previously defined role mining objective that directly addresses this principle.

Usage information is uniquely placed to act as an indicator of necessary privilege [3], and can be mined from activity logs, which are increasingly retained for audit as well as intrusion detection and remediation. Several approaches have extended role mining to incorporate usage information. However, as we discuss in Section 3.3, none of them defines an objective related to the principle of least privilege.

Given this absence, and the centrality of the principle of least privilege to modern security practices, we introduce *minimization of unnecessary privilege* as a formal role-mining objective, along with an associated *unnecessary privilege metric*.

2.2 Business Continuity Over Semantic Roles

Just as agile software development has replaced long-cycle development methodologies, DevOps [20] [50] has broken down the traditional wall between developers and system administrators. To achieve a high "velocity" of software improvement, developers are increasingly empowered to deploy code updates on a daily or even minute to minute basis. A large ecosystem of tools has arisen to support this [19], with some offering the ability to roll back an update if problems are encountered that cannot be addressed by more updates.

In this paradigm, application components such as services are treated as security principals, and *non-human users* may greatly outnumber human users in an RBAC configuration, even if separate identities are assigned at the level of service types rather than individual deployed instances. The privileges a given service needs may change rapidly from version to version, as it is iteratively refactored to support new business and technical requirements, in some cases by decomposition into multiple, simpler microservices.

The ability to roll back updates can make it difficult to be sure when a particular version of a service has gone off-line for the last time. Administrators must weigh the abstract potential benefit of

revoking an unused privilege against certain disruption to the business if they are too hasty. *Policy revocation tools* have emerged which reflect the practice of erring on the side of over-permissioning. These observe the use of privileges over time and make suggestions to remove unused ones [24][23][38]. In one case it is recommended that a privilege be unused for 30 to 90 days before being removed [34] [28].

This approach is seemingly at odds with the development of role mining research, which has moved from *exact optimization*, to exploring *probabilistic* approaches [12][33][46][11]. In this *inexact* view, *under-assignment* - removal of an existing privilege, even if recently used - is permitted if it supports migrating the whole user base to a more semantically meaningful (or just more enduring) set of roles. Whereas, because our user base embraces the high-velocity approach, we must make avoidance of under-assignment the default behavior of OnPar.

However, given that audit log samples are incomplete, these inexact and probabilistic techniques could be useful, so long as they can be configured to focus on adding additional privilege, not present in the samples. A mechanism for integrating such techniques into OnPar is presented in Section 4.2.1. More generally, our formula for the *unnecessary privilege* metric is simplistic, and we anticipate alternative measures and extensions being proposed, that in the best case lead to a “WSC of unnecessary privilege”.

2.3 Multi-Criteria Decision Making

While minimization of unnecessary privilege is a first-class objective for our users, minimization of configuration complexity is still very important. A less complex configuration is easier to approve, change, and audit, and can reduce administration overhead and error [22] [47] [4].

The two objectives of minimizing unnecessary privilege and minimizing configuration complexity conflict with one another. In all but trivial cases, it is impossible to find a single solution that fully optimizes a configuration for them both simultaneously, and a trade-off must be accepted. Fortunately multi-objective role-mining is an established technique, with multiple works adopting the *linear combination* approach, by which a *weighting coefficient* is used to balance the influence of the two objectives in the selection of a candidate.

However, in building an end-to-end solution for use by administrators we encountered an issue with the linear combination approach: presenting the decision maker this single candidate that is optimal for one weighting coefficient value may result in a better candidate being missed. This occurs because the search space exploration for any particular weight value is not exhaustive. Searching with a different weight can often produce a candidate that is *strictly better* - i.e. simultaneously better at reducing unnecessary privilege and configuration complexity. In fact, as we show empirically in Section 5.5, no one weighting coefficient value performs best across a wide range of inputs, and generating candidates using a range of values will generally improve candidate quality.

But even if strictly worse candidates are eliminated, multiple candidates may remain. Picking one of these candidates on behalf of the decision maker could lead to organizational preferences being disregarded. This issue with the linear combination approach to role

mining was recently identified by Crampton *et al.* [5], who address it by introducing the alternative *nonlinear* multi-objective approach to role mining. By this approach, the equally good candidates are identified as *Pareto optimal* (forming a *Pareto set* or *Pareto front*), and must be chosen between based on organizational preference not captured by the formal objectives.

We build upon this connection to the wider literature of multi-objective optimization by introducing *multi-stage Pareto filtering* [27] and the *hypervolume indicator* [2]. Together these allow the set of Pareto optimal candidates to be sampled in a way that reduces the number of candidates the decision maker must reason about to a humans-scale number, while still representing the full range of trade-off choices that were found.

3 RELATED WORK

We focus on work directly related to the current approach. For an authoritative survey of role mining, see Mitra *et al.* [30].

3.1 Role Mining Objectives

Vaidya *et al.* [1] and Guo [14] introduce the concept of objectives to role mining problem and formalized it: Each role mining solution must have one or more objective, each of which has an *objective function*. The role mining literature has considered a wide variety of objectives:

Minimization of configuration complexity has received the greatest attention [42][40][8][31][32][39][17]. This is logical as RBAC aims to rationalize and regularize permission assignment that can easily occur with prior access control models [9]. *Weighted Structural Complexity* (WSC) [31][32] has proven to be flexible and widely applicable compound complexity metric. OnPar uses WSC, which we discuss in Section 4.3.

Minimization of perturbation, meaning the amount of difference between the current and proposed RBAC configuration, has received sustained attention [1] [39] [49]. It can be used to slow the rate of configuration change, or to support the intuition that the current configuration holds some amount of institutional knowledge. We discuss future support for perturbation in OnPar in Section 6.

Reduction in administrative cost [22] and **minimization of administrative burden** [47] [4] allow business considerations to be represented directly within the role mining activity.

Mining semantically meaningful roles, defined loosely as ones that correspond to business roles or organizational structure, has been pursued since the introduction of role mining [26] [33] [46] with a variety of metrics being offered. This objective can be used to preserve organizational semantics when that is desirable. However, it is less relevant in our business environment, where the focus is on application permissions, which typically do not have a clear mapping to organizational structure.

The current work introduces the objective of *minimization of unnecessary privilege* to role mining, along with the associated *unnecessary privilege* metric.

3.2 Multi-Objective Role Mining

Vaidya *et al.* [1] and Guo [14] also introduce multi-objective role mining. To consider multiple objectives, they combine individual

objective functions into a *combination function*. Linear combination via a *weighting coefficient* is proposed on the grounds that it is an easy way to control the relative contribution of the different objectives. Subsequent multi-objective role mining works have primarily followed this *linear combination approach* [12][39][49], differing only in terminology. For consistency, we use the terms introduced by [1] and [14] throughout this paper.

The evaluation methodology used in this line of work is the same in each case: The weighting coefficient value is swept between 0.0 and the highest useful value (typically 1.0) and a relationship between the weighting coefficient and the two individual objective functions is confirmed. It is also confirmed that the extreme weighting coefficient values have the expected effect of choosing a candidate with properties similar to those chosen by single-objective optimization.

Two of the evaluations [12] [49] consider the relationship between the pair of individual objective values as the weighting coefficient value is varied, by plotting one objective on the x-axis and the other on the y-axis of the same chart. Examining these charts (Figure 5 of [12] and Figure 4 of [49]), it can be seen that some weight values produce a solution that is strictly better than one of its neighbors. That is: a solution that is either better with respect to both objectives, or better with respect to one objective and as good with respect to the other objective. However, the same charts also show significant ranges of weighting coefficient value where neither of a pair of neighboring weights is strictly better than the other. [12] identifies that this demonstrates *Pareto optimality*, and offers that *in a concrete application*, the choice between the different trade-offs that these Pareto-optimal solutions represent would need to be based on other conditions, besides the two objective values.

Despite these characteristics, none of the works taking the linear combination approaches offers guidance in this area. It is left as an exercise for the user to decide when and how the weighting coefficient should be varied, and no method is proposed to avoid the presentation of strictly worse solutions.

OnPar follows the linear combination approach of Vaidya et al. [1] and Guo [14] for the purpose of candidate generation. But to address the above issues surrounding user configuration of the weighting coefficient, OnPar automates the varying of the weight value.

Crampton et al. [5] introduce *nonlinear multi-objective optimization* [29] and hence *Pareto optimality* to role mining. By this approach, the scores for the individual objectives are not combined in order to reduce them to a single score. Their motivation is the observation that when two objectives conflict such that they cannot be fully optimized simultaneously, presenting the *Pareto front* of alternative solutions to the decision maker allows them to choose the most suitable alternative for that particular situation.

While OnPar uses the linear combination approach to generate candidates, it follows [5] in evaluating the candidates in terms of their individual objective scores. After candidate generation, any weighted score associated with a candidate during generation is disregarded, and the individual objective scores are considered directly both for filtering the set of candidates to be presented to the decision maker, and in the presentation for decision making.

Building on the connection to the wider literature of Multi-Criteria Decision Making (MCDM) established by [5], OnPar introduces the use of *multi-stage Pareto filtering* [27], in order to allow the number of candidates presented to the decision maker to be kept within the limits of human cognitive load. However, OnPar differs from [27] in using the *hypervolume indicator* [2] as the filtering metric, since this metric does not make assumptions about the distribution of solutions along the Pareto front. The reason for this approach is discussed in Section 4.7.

3.3 Usage Based Role Mining

Molloy, Park and Chari [33] introduce usage based role mining. They define a binary matrix that indicates whether a given user has used a given permission. We adopt this matrix in OnPar. However, we refer to it as *USE* (for *Use Matrix*), since their name (*GUPA*, for *Generative User Permission Assignment*) is specific to their setting.

Molloy et al. also define a metric (λ -DISTANCE) which includes a term for over-assignment of permissions. That term by itself is isomorphic with the *unnecessary privilege* metric, but λ -DISTANCE also has a term for under-assignment, which is unweighted and hence cannot be eliminated. Therefore λ -DISTANCE cannot be configured to be equivalent to the *unnecessary privilege* metric.

Zhang et al. [49] also consider privilege use information. To reason about usage information, they introduce the *User-Permission Invocation Matrix*, which is analogous to the *GUPA/USE* matrix, but records the count of how many times a given user has used a given permission. They define a homogeneity of permission use rate metric, to express the intuition that good roles have similar rates of use for all their assigned users. They do not give special treatment to unused permissions, and they do not track the amount of privilege assigned or seek to optimize it.

Lu et al. [25] introduce the *Usage Role Mining Problem* (Usage RMP). But *usage* refers to a set of user-role assignments. Similarly, John et al. [18] introduce a *role-usage cardinality constraint*, which relates to the number of roles that one user may be assigned. Neither work considers the usage of assigned permissions.

The present work introduces *minimization of unnecessary privilege* as a role mining objective, which is the first objective designed to allow the principle of least privilege to be considered explicitly in multi-objective optimization. It also introduces the associated *minimization of unnecessary metric*, which is the first metric that compares usage information to assigned privilege (so called *entitlement information*) for the purpose of quantifying unnecessary privilege.

4 DESIGN

4.1 Overall Approach

OnPar is designed to support administrators in making trade-off decisions between the two conflicting objectives of minimization of unnecessary privilege and minimization of configuration complexity. To achieve this, it adopts the Generate First-Choose Later approach [6]. As a result, its three stage pipeline, illustrated in Figure 2, starts with a Candidate Generator step which is only concerned with finding a set of candidates that has good coverage of the space of feasible solutions. The Pareto-Optimal Candidate Filter takes this potentially large *candidate set* removes candidates that are

Figure 2: OnPar System Architecture. Candidates are generated and then filtered - once to remove sub-optimal alternatives and again to choose a small representative subset for consideration by the administrator.

sub-optimal, since these should never be shown for evaluation. The Distinct Candidate Filter takes that Pareto-optimal set of candidates and reduces it to a human-scale number of candidates. The output of the Distinct Candidate Filter is presented to the administrator for evaluation.

The stages of this design are loosely coupled and the candidate set passed to the Pareto-Optimal Candidate Filter could come from the generator we propose or a different generator entirely, including one from an existing multi-objective role mining solution.

Each candidate is scored against the two independent objectives at the time it is generated. The scores are used conduct the final evaluation. They are also used by all three stages of the pipeline. For this reason, we first present the metrics and then the details of each pipeline stage in order of execution.

4.2 Unnecessary Privilege Metric

Conceptually, the *unnecessary privilege metric* (UP) for a RBAC configuration is the count of unused user-permission assignments that it authorizes, which is equivalent to the Hamming distance between the assigned permissions (UPA) and used permissions (USE).

To define it formally, we adopt the NIST RBAC model [35]. Given a set of *users* U , a set of *permissions* P , and a set of *roles* R :

- The relation $UA \subseteq U \times R$ describes the assignment of users to roles.
- The relation $PA \subseteq R \times P$ describes the assignment of permissions to roles.
- The relation $UPA \subseteq U \times P$ describes the assignment of permissions to users.

We follow the standard role mining practice [40] of representing these relations as *binary matrices*. Figure 3 shows a user-permission assignment matrix (UPA), where each row corresponds to a user, each column to a permission, and the elements are binary values indicating whether a given user is assigned a given permission (1) or not (0). For visual clarity we represent 0 values by a blank cell.

To represent usage information we adopt the additional binary matrix introduced by [33] but as discussed in Section 3.3 we name it USE due to its general applicability. Like UPA , USE has a row per user and a column per permission, but its element values indicating whether a given permission has been used by a given user.

With the UPA and USE matrices defined, UP can be calculated by counting the number of non-zero elements in the UPA matrix where the corresponding element in the USE matrix is zero:

$$UP(\gamma) = \sum_{i=1}^{|U|} \sum_{j=1}^{|P|} [UPA_{i,j} = 1 \wedge USE_{i,j} = 0]$$

An important property of this metric is that a privilege authorized for the same user by multiple roles with overlapping permissions is only counted once. From the perspective of the principle of least privilege, having the same permission assigned to the same user via multiple roles does not increase the risk associated with that assignment: An attacker can exploit the privilege no more or less due to the duplicate assignment.

4.2.1 Privilege Assignment Overrides. OnPar allows overrides to be applied between populating the USE matrix from available activity log samples and calculation of UP . This mechanism can be used in the event that the log sample is known to be insufficient or inaccurate. It is also used in the normal course of operation, for example to produce candidates that remove a privilege present in the logs but known to no longer be required, or to produce candidates that include a privilege that is about to be granted.

In the current implementation, overrides may be defined by simple predicate rules on user, role and permission - for example to treat all privileges associated with a particular user and role as unused. These overrides are applied in an automated manner.

However, the mechanism is agnostic to the origin of the overrides, and more advanced techniques could be applied as well or instead. In particular, learning generative models from logs [49] and *missing value* tolerant variants of matrix factorization such as Multiple Factor Optimization (MFO) [44] have both been shown to be effective at handling incomplete activity records, and their results could be applied as overrides.

4.3 Configuration Complexity Metric

OnPar uses Weighted Structural Complexity (WSC) [31][32] as the configuration complexity metric. Given an RBAC configuration γ , WSC is defined as:

$$wsc(\gamma, W) = W_r|R| + W_u|URA| + W_p|RPA| + W_d|DUPA| + W_t|t_reduce(RH)|$$

where R is the set of roles, URA is set of user-role assignments, RPA is the set of role-permission assignments, $DUPA$ is the set of direct user-permission assignments, and $t_reduce(RH)$ is the transitive reduction of the role hierarchy. Since Vault [16], the system OnPar performs role mining for, does not support $DUPA$ or role hierarchy, we set coefficients W_t , W_d to zero. This is equivalent to eliminating the $DUPA$ and role-hierarchy terms.

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9
U_1	1	1	1						
U_2	1	1	1	1					
U_3				1	1	1			
U_4				1	1	1			
U_5			1	1			1	1	1
U_6			1	1	1	1	1	1	1

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9
U_1	1	1							
U_2		1	1						
U_3				1		1			
U_4				1					
U_5			1					1	
U_6				1		1	1	1	1

Figure 3: The UPA binary matrix (left) describes permission assignment for an RBAC configuration. The USE binary matrix (right) describes permission usage, which can be used to optimize the configuration.

The considerations for selecting the weight values are the same as with prior use of WSC. As with previous approaches, the WSC weights cannot be varied within one execution of OnPar, since the comparison of candidates involves comparison of their individual complexity scores.

4.4 Privilege-Complexity Charts

To understand the operation of OnPar and evaluate its performance, it is useful to visualize a candidate set as a scatter plot where the points correspond to candidates offering different trade-offs between unused privilege and complexity. By convention we plot configuration complexity on the x-axis and unused privilege on the y-axis. We refer to these plots as *privilege-complexity charts*.

OnPar calculates UP for each RBAC configuration considered during candidate generation (where a *candidate* is a possible replacement RBAC configuration OnPar suggests as an alternative to the input configuration), and the value drives the selection of candidates at each stage of the pipeline. For candidate generation, UP is used as component of the *candidate score*, which guides the exploration of the space of possible candidates. The UP value for each generated candidate is considered directly by the Pareto and Distinct filters. UP is also calculated for the input RBAC configuration, as a baseline.

4.5 Candidate Generator

The Candidate Generator consumes the input RBAC configuration, plus a sample of the activity logs, which is presented to it in the form of a USE matrix, as defined in Section 4.2. Its goal is to generate a large candidate set, with good coverage of the space of possible trade-offs between unnecessary privilege and configuration complexity.

Cluster-Based Candidate Generation Algorithm We follow ORCA [37] in using *Hierarchical Agglomerative Clustering* (HAC) to iteratively construct alternative candidate configurations. However, we extend the approach in several ways:

Multiple initial cluster sets. ORCA can only produce candidates where each permission appears in one role. This is a consequence of seeding the HAC algorithm with clusters which correspond to individual permissions. We address this by running HAC independently on multiple, alternative *initial cluster sets*. We keep ORCA's *permission-based initial cluster set*, but supplement it with a *user-based initial cluster set*, where each cluster corresponds to a group of users. The union of the candidates found via the two initial cluster sets is taken as the output of the Candidate Generator.

Usage based initial cluster sets. ORCA bases its initial clusters on permission assignment information. OnPar instead bases the initial clusters on patterns of permission usage present in the USE matrix. The initial clusters in each set form an exact cover of the privileges that are marked as used, with the results that the candidate corresponding to each initial cluster set has zero unnecessary privilege (UP), and a relatively high configuration complexity (WSC). As clusters are iteratively merged by HAC, a succession of candidates is generated, each having lower complexity than the last (since the number of roles is decreasing), but having either the same or higher unnecessary privilege (since the initial roles covered only the necessary privilege).¹

Weighted Candidate Score Based Clustering. At each iteration of HAC, OnPar chooses which pair of clusters to merge based on the UP and WSC scores of candidates. As with other multi-objective role mining algorithms, a combination function is applied to produce a weighted linear combination of the individual scores:

$$S(\gamma) = \alpha UP(\gamma) + (1 - \alpha) WSC(\gamma)$$

where α is a weighting coefficient.

Weighting Coefficient Sweep The Candidate Generator runs HAC multiple times for different weighting coefficient values. By default, and for the experiments used for the evaluation, the value of α is swept from 0.0 to 1.0 in steps of 0.1.

This multi-weight approach allows candidates to be discovered that may offer novel trade-offs between the different candidate metrics, that would not be found via a single-weight search. In Section 5.5 we empirically evaluate the effectiveness of this approach.

4.6 Pareto Optimal Filter

The Candidate Generator produces a large candidate set, without considering whether some of them are redundant. Some candidates are worse than others by all goodness measures. These can safely be eliminated, as no administrator would choose one of them given an understanding of their relationship to other candidates.

The Pareto Optimal Filter achieves this by reducing the candidate set to only contain the subset that are Pareto-optimal. A candidate C_1 is Pareto-optimal if there does not exist another candidate C_2

¹Conceptually, HAC is finding a succession of set-covers of the necessary privileges, with the roles as the covering sets. The candidate corresponding to an initial cluster set is an exact cover, and the ones that follow are inexact, with an increasing number of unnecessary privileges also covered.

Figure 4: Privilege-Complexity Charts show the candidates emitted versus filtered out by OnPar at each stage. *Input* is the input RBAC configuration, which acts as a baseline.

such that:

$$\begin{aligned}
 & (UP(C_2) < UP(C_1) \wedge WSC(C_2) \leq WSC(C_1)) \\
 & \quad \vee \\
 & (WSC(C_2) < WSC(C_1) \wedge UP(C_2) \leq UP(C_1))
 \end{aligned}$$

If the above condition does not hold, then C_1 is said to be dominated by C_2 , since C_2 is better than (*i.e.* dominates) C_1 in one dimension while being no worse in any other dimension. C_2 can also be referred to as *strictly better than* C_1 .

In the current implementation, the Pareto Optimal filter starts with an empty candidate set and the candidates produced by the generator are considered one by one to be added to this set. If the next candidate is dominated by any of the current set members, it is not added to the new set. If it is not dominated by any of them, it is added to the set. In the latter case, it may dominate some of the current set members, in which case they are removed from the set. The running time of this streaming approach is evaluated in Section 5.6, and found to be negligible compared to that of candidate generation.

4.7 Distinct Filter

The Distinct Filter is applied to the candidate set output by the Pareto Optimal Filter, in order to reduce the candidate count to human-scale. In this context, the literature refers to a Pareto set as a *Pareto front*. The goal of this phase is to select a small number of candidates (μ) from the full Pareto front such that they still represent the variety of trade-off possibilities presented by the full front.

Many performance assessment metrics for Pareto fronts have been proposed. However a large number of them depend on the assumption that a Pareto front with uniformly distributed points is achievable [52], whereas Pareto fronts produced by OnPar typically have non-uniform distribution, particularly at the extremes. Inspection shows that this is due to the way unnecessary privilege accumulates as roles are repeatedly merged, which is itself a consequence of the discrete nature of roles and user-role assignments.

Uniform distribution measures have mostly been proposed in domains where solution parameters are continuous variables, to which they are better suited.

The *hypervolume indicator* [51] is agnostic to the distribution of points, and considered to be one of the most robust Pareto front measures [52][2], and we therefore adopt it. For $\mathbb{R}^2$, it is defined as the area above the Pareto front with respect to a bounding box formed by the origin and a reference point. It is a scalar indicator of the optimality of a Pareto front which treats each dimension with equal weight.

A key requirement for using the hypervolume indicator is that the reference point must be dominated by every point in the input set [2]. To ensure this, OnPar takes the maximum values for WSC and UP that are assigned to candidates in the set passed to the Distinct Candidate Filter, adds 5% padding to each value and sets these as the coordinates of the reference point.

Figure 5 gives an example of the hypervolume indicator for a sample Pareto front. Each point P_i has a *hypervolume contribution* HC_i , which represents the area added to the hypervolume indicator by including this point in the set. The hypervolume contribution of P_i in a given Pareto set is defined as the area which would be removed from the bounding box were this point removed from the rest of the Pareto set. Thus, the hypervolume contribution is useful for measuring the loss generated by removing a point from the set. In Figure 5, the hypervolume contribution of P_3 is identified.

Having selected a reference point, OnPar calculates the hypervolume indicator h_0 of the points on the Pareto front. h_0 serves as a baseline for measuring loss as candidates are removed from the front. The μ points which maximize the hypervolume indicator are then identified. For small Pareto fronts, exhaustive search is used. However, as the number of points on the front increases, exhaustive examination of all combinations of μ points is too slow for practical use. Therefore, simulated annealing[21] is used to search for the optimal candidates in the general case. The effectiveness and running time of this approach are evaluated.

Figure 5: The hypervolume indicator. Each P_i is a point on the Pareto front. Shading indicates the hypervolume defined by points P_0, P_2, P_3, P_5, P_6 and P_8 , while the shaded area labeled HC_3 is the hypervolume contribution of the point P_3 to that hypervolume. The hypervolume is defined relative to a Reference Point. Padding (of 5%) is used to ensure it is dominated by all points on the front.

5 EVALUATION

We evaluate OnPar in terms of the quality of its output and its running time. A wide range of inputs are considered, and the properties of the input that affect the various aspects of performance are identified. The three stages of OnPar’s pipeline are evaluated separately, since they each contribute to overall performance.

5.1 Input Data Set Generation

To benchmark against a diverse range of inputs, we generate a large number of synthetic RBAC states. We follow the randomized approach established by FastMiner [42][1][41][43]: Each RBAC state has a specified number of users (U), permissions (P) and roles (R). Each role has between 1 and *maximum permissions per role* (MPR), and each user has between 1 and *maximum roles per user* (MRU). (We deviate from FastMiner (and hence follow Zhang *et al.* [48]) by not allowing users with zero assigned roles. Such users would be ignored, and their inclusion would only obscure the effect of U and MRU on performance.)

Since an RBAC state only describes permission assignment, we extend the above approach to model different patterns of usage. We follow Zhang *et al.* [49], in referring to the roles in the RBAC state as *actual roles* and supplement them with *ideal roles*, each of which represents a different pattern of permission usage. They are ideal in the sense that they would have no unnecessary privilege if they replaced the actual roles.

We take a “top down” approach and derive a set of ideal roles for each actual role, whereas [49] creates one set of ideal roles and combines random subsets to form actual roles, in “bottom up” fashion. Our approach preserves the diversity of *permission overlap* (same permission assigned to multiple roles) that the above

RBAC state generator produces, which allows its effect on OnPar’s performance to be investigated.

We define an *Ideal Role Ratio* (IRR), which determines how many ideal roles each actual role is decomposed into, and run experiments with 2,3,4 and 5 ideal roles per actual role. The ideal roles are used to generate synthetic event logs, which OnPar loads to create the *UPIM* for each experiment. Since the number of times each privilege is used is not considered, it is sufficient to log one event for each user-permission pair associated with each ideal role.

Table 1: Range of RBAC state generation parameters used

Parameter	Min Value	Max Value
Users (U)	10	410
Permissions (P)	10	1010
Roles (R)	5	410
Max. Roles per User (MRU)	3	25
Max. Perms. per Role (MPR)	1	155

5.2 Evaluation Methodology

Table 1 shows the range of values explored for RBAC state generator’s parameters. A total of 336 parameter combinations are considered. Each parameter combination is used for four experiments, each with a different value of IRR in the range 2 to 5.

The randomized RBAC state generation approach means that multiple runs with the same generator parameters produce RBAC states with very different structural properties such as the actual number and overlap of user-roles assignments and number and overlap of role-permissions assignments. As the experimental results show, these structural properties have a more direct influence on OnPar’s performance than the generator parameters. We therefore run each experiment once, and focus on ensuring that the parameter combinations tested offer good coverage of the space of realistic parameter values.

Experiments are performed on AWS m5a.4xlarge class VMs, which offer 16 vCPUs and 64GB of RAM. Ubuntu Linux 18.04LTS is used as the operating system. Our implementation of OnPar is written in Go, and is currently single-threaded. The Go runtime makes periodic use of 1 or 2 additional CPU cores for activities such as garbage collection, but we are able to run 4 experiments concurrently on a single instance with the *loadavg* of the Linux scheduler remaining below 16 at all times, which indicates that the experiments do not interfere with one another.

5.3 Candidate Set Quality Metrics

We evaluate candidate set quality via the metrics described below. Where possible we consider each metric both for the *full set of candidates*, and for just the candidates that are *strictly better* than the input configuration, as defined in Section 4.6.

The output of the Distinct Filter stage represents OnPar’s final output, and these are the primary concern. However, where possible metrics are reported for the two earlier stages of the pipeline, to understand their effect on final performance.

Number of candidates. We expect the target number ($\mu = 5$) of candidates to be output by the Distinct filter unless there are

Figure 6: Factors (x-axis of charts) with biggest effect on key aspects of OnPar's performance (y-axis of charts).

Figure 7: Percentage of globally optimal candidates found by individual weights.

fewer than that number of Pareto-optimal candidates to choose from.

Reduction in unnecessary privilege and configuration complexity. We define metrics RUP_c and $RWSC_c$ for the reductions offered by an individual candidate and μRUP and $\mu RWSC$ for the average reduction offered by a set of candidates. RUP_c , the reduction in unused privilege for an individual candidate p is:

$$RUP_c = 1.0 - (UP_c / UP_I)$$

where UP_c is the value of the unused privilege metric UP for that candidate and UP_I is the value for the same metric for the input RBAC state. μRUP , the mean reduction in unused privilege for a candidate set CS is then:

$$\mu RUP = \frac{1}{|CS|} \sum_{c \in CS} RUP_c$$

Similarly, $RWSC_c$, the reduction in complexity for an individual candidate p is:

$$RWSC_c = 1.0 - (WSC_c / WSC_I)$$

where WSC_c is the value of the complexity metric WSC for that candidate and WSC_I is the value for the same metric for the input RBAC state. The mean reduction in complexity $\mu RWSC$ for a candidate set CS is then:

$$\mu RWSC = \frac{1}{|CS|} \sum_{c \in CS} RWSC_c$$

Representative sample. The candidates chosen by the Distinct Candidate Filter should be drawn from the Pareto-optimal set in a way that reflects the range of trade-off choices offered by the full set. We measure this via the percentage reduction in hypervolume between the output of the Pareto Optimal and Distinct Candidate Filter. A hypervolume-based metric is used, rather than a metric based on distribution across the range of metric values, for the reasons discussed in Section 4.7.

5.4 Candidate Set Quality

Table 2 summarizes the number of candidates per candidate set across all experiments. OnPar is able to offer the target number of 5

distinct candidates in all cases where that many Pareto-optimal candidates are found. For the strictly better results, we run the Distinct filter on just the strictly better candidates, on the assumption that the administrator would want to choose from among this subset. In both the full and strictly better case, the Pareto Optimal filter significantly reduces the number of candidates that the Distinct filter must consider.

Table 3 characterizes the average reduction in unnecessary privilege and complexity offered by strictly better candidates. On average, the distinct candidates selected by OnPar offer a 16% reduction in configuration complexity and a 63% reduction in unnecessary privilege.

As expected, the remaining (not strictly better) candidates show even greater average reductions in one of the metrics, at the cost of actually increasing the other metric above its value in the input configuration. We do not characterize them further as in most cases administrators are likely to only be interested in strictly better candidates, which are typically plentiful.

The number of Pareto-optimal candidates found is most strongly correlated with the input unnecessary privilege and complexity (UP_I and WSC_I). The correlations are discernible in the left-hand and center charts in Figure 6.

Table 4 shows the performance of the Distinct filter, in terms of the percentage of the hypervolume of the full Pareto set that is lost by the hypervolume of the five points that are selected (Actual Loss). The median loss is less than 6%. This delta rises with the size of the full set (Set Size) but remains below 12% in the worst case encountered. However, much of this loss is unavoidable. Inspection shows that the larger Pareto sets are not necessarily convex, so the presence of more points to choose from does not necessarily mean a 5-point approximation can be found that preserves as much of the hypervolume as is the case with smaller Pareto sets. To confirm this, we found the optimal 5-point hypervolume by brute force for as many of the configurations as possible, and calculated its loss (Ideal Loss). We were able to compute it for sets of up to 49 candidates, which corresponds to the 68th percentile of the full set sizes seen across all experiments. The results his shows that almost all of the loss is unavoidable, and the quality of the Distinct Filter's candidates is near optimal, at least to the 68th percentile.

5.5 Effectiveness of Multiple Weights

We evaluate the effectiveness of OnPar's multi-weight approach by comparing the full set of Pareto-optimal candidates it finds for each experiment with the set found by each individual weight. Figure 7 shows the percentage of the full set found by each weight, across all experiments. (Tukey whiskers are shown - at 1.5x Inter-Quartile Range (IQR).)

Considering the full set of Pareto-optimal candidates found by OnPar, in some cases a single weight can find around 90% of the full set, but the median percentage found varies by weight between 5% and 22%. Considering only strictly better candidates, the maximum percentage found by one weight drops to 80% and the median ranges from 0% to 25%.

This shows clearly that using a single weight, or even a few neighboring weights will lead to a significant proportion of the Pareto-optimal candidates being missed.

5.6 Running Time

Table 5 shows the variation in running time of the three stages of OnPar's pipeline, across all experiments. It is dominated by the running time of the candidate generator, which is expected given the complexity of the hierarchical clustering algorithm compared to the Pareto Optimal filter and annealing-based μ -approximation algorithms. The Pareto Optimal filter takes only 4ms to run in the median case and 314ms in the worst case encountered in the experiments. The Distinct filter is also efficient, taking 19ms in the median case and 127ms in the worst case encountered in the experiments.

The running time for a given input is strongly correlated with the number of initial clusters used for the Candidate Generator stage of OnPar. This can be seen clearly in the right-hand chart of Figure 6. This relationship is expected, given that the number of levels in a HAC hierarchy is equal to it's number of input clusters.

6 CONCLUSION

In this paper we proposed OnPar, a new role mining approach that introduces *minimization of unnecessary privilege* as a new formal objective for role mining, along with an associated *unnecessary privilege metric*. We have also introduced the use of Pareto optimality and Pareto filtering to role mining. These allow the elimination of user tuning of global objective weights while guaranteeing that sub-optimal candidates are not offered for evaluation. Lastly, we have applied the hypervolume indicator, to sample the Pareto optimal candidates in a representative manner. This allows the number of candidates offered to the user to be significantly reduced, with negligible reduction in ability to convey the range of trade-off alternatives available.

Our evaluation, using a wide range of synthetic input data sets, demonstrates that these techniques are robust and efficient, across a wide range of input configuration scale and structural properties. Crucially, the results confirm that the OnPar approach can find significantly more high-quality candidates than a single weight.

Opportunities for future work include:

Extensions to the unnecessary privilege metric. Activity log samples are imperfect records of necessary privilege. OnPar's privilege assignment override mechanism, described in Section 4.2, allows variation from the log's view of necessary privilege to be introduced in an automated manner. At present we use predicate-based overrides, but the override mechanism is generic which allows for more advanced techniques to be investigated.

Learning generative models from logs [49] and *missing value* tolerant variants of matrix factorization such as Multiple Factor Optimization (MFO) [44] have both been shown to be effective at handling incomplete activity records, and both approaches should be investigated. More generally, formalizing the contributions of different sources of overrides could lead to the development of a more nuanced and configurable unnecessary privilege metric, analogous to WSC.

Elimination of the weighting coefficient from the candidate generation stage. This is a logical next step, given that the Pareto-Optimal and Distinct Filters are inspired by weight-free search methods from the literature of Multi-Objective Optimization.

Table 2: Number of candidates per candidate set

Percentile	Total Number of Candidates			Strictly Better Candidates		
	Generated	Pareto	Distinct	Generated	Pareto	Distinct
25th	231	17	5	25	5	5
50th	430	27	5	66	14	5
75th	767	138	5	223	63	5

Table 3: Simultaneous reduction in unnecessary privilege and complexity offered by *strictly better* candidates

Percentile	Average Reduction Complexity			Average Reduction Unnecessary Privilege		
	Generated	Pareto	Distinct	Generated	Pareto	Distinct
25th	6%	6%	7%	23%	27%	37%
50th	10%	13%	16%	44%	50%	63%
75th	17%	19%	29%	56%	61%	95%

Table 4: Distinct Filter quality

Percentile	Pareto Set Size	Actual Loss	Ideal Loss
Min	5	0.00%	0%
25th	18	2.68%	2.67%
50th	28	5.43%	4.99%
75th	143	7.18%	n/a
Max	739	11.84%	n/a

Table 5: Running times of OnPar's three stages

Percentile	Generator	Pareto Filter	Distinct Filter
Min	0.02s	0ms	0ms
25th	14s	1ms	17ms
50th	148s	4ms	19ms
75th	760s	30ms	32ms
Max	11655s	314ms	127ms

Pareto-based Genetic Algorithms [6] and in particular NSGA-II [7] are worthy of consideration.

Extension to three or more objectives. Considering more than two objectives would allow other goodness measures to be considered simultaneously. Minimization of perturbation has shown good results [1][39][49] in the bi-objective setting. However we encounter customers with current RBAC configurations known to be undesirable. For these cases, attribute-based approaches, that consider available metadata about users or permissions might be a useful alternate [33][46]. Performance with a third objective should

also be compared to the performance that can be achieved by expressing the additional objective as a constraint on proposal generation or a penalty term in the case where weight-based search is still employed.

REFERENCES

- [1] Vijayalakshmi Atluri, Jaideep Vaidya, and Qi Guo. 2008. Migrating to Optimal RBAC with Minimal Perturbation. (2008).
- [2] Anne Auger, Johannes Bader, Dimo Brockhoff, and Eckart Zitzler. 2009. Theory of the Hypervolume Indicator: Optimal μ -Distributions and the Choice of the Reference Point. In *Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA '09)*. 87–102.
- [3] Elisa Bertino, Seraphin Calo, Hong Chen, Ninghui Li, Tiancheng Li, Jorge Lobo, Ian Molloy, and Qihua Wang. 2008. Some usability considerations in access control systems. In *Symposium on Usable Privacy and Security (SOUPS) 2008*.
- [4] Alessandro Colantonio, Roberto Di Pietro, and Alberto Ocello. 2008. A Cost-Driven Approach to Role Engineering. In *Proceedings of the 2008 ACM Symposium on Applied Computing (SAC '08)*. 2129–2136.
- [5] Jason Crampton, Gregory Gutin, Daniel Karapetyan, and Rémi Watrigant. 2017. The bi-objective workflow satisfiability problem and workflow resiliency 1. *Journal of Computer Security* 25, 1 (2017).
- [6] Kalyanmoy Deb. 2001. *Multi-objective optimization using evolutionary algorithms*. Vol. 16. John Wiley & Sons.
- [7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002).
- [8] Alina Ene, William Horne, Nikola Milosavljevic, Prasad Rao, Robert Schreiber, and Robert E. Tarjan. 2008. Fast exact and heuristic methods for role minimization problems. (2008), 1–10.
- [9] David Ferraiolo and Richard Kuhn. 1992. Role-Based Access Control. In *15th NIST-NCSC National Computer Security Conference*. 554–563.
- [10] David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. Richard Kuhn, and Ramaswamy Chandramouli. 2001. Proposed NIST Standard for Role-based Access Control. *ACM Trans. Inf. Syst. Secur.* 4, 3 (2001), 224–274.
- [11] Mario Frank, Joachim M. Buhman, and David Basin. 2013. Role Mining with Probabilistic Models. *ACM Trans. Inf. Syst. Secur.* 15, 4, Article Article 15 (April 2013).
- [12] Mario Frank, Andreas P. Streich, David Basin, and Joachim M. Buhmann. 2009. A probabilistic approach to hybrid role mining. In *Proceedings of the 16th ACM conference on Computer and communications security*. 101–111.
- [13] Nurit Gal-Oz, Yaron Gonen, Ran Yahalom, Ehud Gudes, Boris Rozenberg, and Erez Shmueli. 2011. Mining Roles from Web Application Usage Patterns. In *Trust, Privacy and Security in Digital Business*, Steven Furnell, Costas Lambrinoudakis, and Günther Pernul (Eds.). 125–137.
- [14] Qi Guo. 2010. *A formal approach to the role mining problem*. Ph.D. Dissertation. Rutgers University-Graduate School-Newark.

- [15] HashiCorp. 2015. hashicorp/vault: A tool for secrets management, encryption as a service, and privileged access management. <https://github.com/hashicorp/vault>. [Online; accessed 17-Apr-2020].
- [16] HashiCorp. 2015. Vault by HashiCorp. <https://www.vaultproject.io>. [Online; accessed 17-Apr-2020].
- [17] Jafar Haadi Jafarian, Hassan Takabi, Hakim Touati, Ehsan Hesamifard, and Mohamed Shehab. 2015. Towards a General Framework for Optimal Role Mining: A Constraint Satisfaction Approach. (2015), 211–220.
- [18] John C. John, Shamik Sural, Vijayalakshmi Atluri, and Jaideep S. Vaidya. 2012. Role Mining under Role-Usage Cardinality Constraint. In *Information Security and Privacy Research*. Springer Berlin Heidelberg, 150–161.
- [19] Mik Kersten. 2018. A cambrian explosion of DevOps tools. *IEEE Software* 35, 2 (2018), 14–17.
- [20] Gene Kim, Jez Humble, Patrick Debois, and John Willis. 2016. *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations*. IT Revolution.
- [21] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. 1983. Optimization by Simulated Annealing. *Science* 220, 4598 (1983).
- [22] Martin Kuhlmann, Dalia Shohat, and Gerhard Schimpf. 2003. Role mining - revealing business roles for security administration using data mining technology (SACMAT '03). ACM, 179–186.
- [23] Duo Labs. 2018. Cloudtracker GitHub Repository. <https://github.com/duo-labs/cloudtracker>. [Online; accessed 10-Feb-2020].
- [24] Duo Labs. 2018. Introducing Cloudtracker. <https://duo.com/blog/introducing-cloudtracker-an-aws-cloudtrail-log-analyzer>. [Online; accessed 10-Feb-2020].
- [25] Haibing Lu, Jaideep Vaidya, and Vijayalakshmi Atluri. 2014. An Optimization Framework for Role Mining. *J. Comput. Secur.* 22, 1 (Jan. 2014), 1–31.
- [26] Xiaopu Ma, Ruixuan Li, and Zhengding Lu. 2010. Role Mining Based on Weights (SACMAT '10). 65–74.
- [27] Christopher A. Mattson, Anoop A. Mullur, and Achille Messac. 2004. Smart Pareto filter: obtaining a minimal representation of multiobjective design space. *Engineering Optimization* 36, 6 (2004).
- [28] Travis McPeak. 2018. Least Privilege: Security Gain without Developer Pain. In *Enigma 2018 (Enigma 2018)*.
- [29] Kaisa Miettinen. 2012. *Nonlinear multiobjective optimization*. Vol. 12. Springer Science & Business Media.
- [30] Barsha Mitra, Shamik Sural, Jaideep Vaidya, and Vijayalakshmi Atluri. 2016. A Survey of Role Mining. *ACM Computing Surveys (CSUR)* 48 (2016), 50.
- [31] Ian Molloy, Hong Chen, Tiancheng Li, Qihua Wang, Ninghui Li, Elisa Bertino, Seraphin Calo, and Jorge Lobo. 2008. Mining roles with semantic meanings. (2008), 21–30.
- [32] Ian Molloy, Hong Chen, Tiancheng Li, Qihua Wang, Ninghui Li, Elisa Bertino, Seraphin Calo, and Jorge Lobo. 2010. Mining Roles with Multiple Objectives. *ACM Trans. Inf. Syst. Secur.* 13, 4 (2010).
- [33] Ian Molloy, Youngja Park, and Suresh Chari. 2012. Generative Models for Access Control Policies: Applications to Role Mining over Logs with Attribution. In *Proceedings of the 17th ACM Symposium on Access Control Models and Technologies (SACMAT '12)*. ACM, 45–56.
- [34] Netflix. 2017. Introducing Aardvark and Repokid. <https://netflixtechblog.com/introducing-aardvark-and-repokid-53b081bf3a7e>. [Online; accessed 5-Feb-2020].
- [35] U.S.Department of Commerce/National Institute of Standards and Technology. 2004. ANSI/INCITS 359-2004, Role Based Access Control.
- [36] Jerome H Saltzer and Michael D Schroeder. 1975. The protection of information in computer systems. *Proc. IEEE* 63, 9 (1975), 1278–1308.
- [37] Jürgen Schlegelmilch and Ulrike Steffens. 2005. Role Mining with ORCA. (2005), 168–176.
- [38] Florian Sellmayr. 2017. Trailscraper GitHub Repository. <https://github.com/florell/trailscraper>. [Online; accessed 10-Feb-2020].
- [39] Hassan Takabi and James B.D. Joshi. 2010. StateMiner: An Efficient Similarity-Based Approach for Optimal Mining of Role Hierarchy (SACMAT '10). 55–64.
- [40] Jaideep Vaidya, Vijayalakshmi Atluri, and Qi Guo. 2007. The Role Mining Problem: Finding a Minimal Descriptive Set of Roles (SACMAT '07). 175–184.
- [41] Jaideep Vaidya, Vijayalakshmi Atluri, and Qi Guo. 2010. The role mining problem: A formal perspective. *ACM Transactions on Information and System Security (TISSEC)* 13 (2010), 27.
- [42] Jaideep Vaidya, Vijayalakshmi Atluri, and Janice Warner. 2006. RoleMiner: mining roles using subset enumeration. (2006), 144–153.
- [43] J. Vaidya, V. Atluri, J. Warner, and Q. Guo. 2010. Role Engineering via Prioritized Subset Enumeration. *IEEE Transactions on Dependable and Secure Computing* 7, 3 (2010), 300–314.
- [44] Sokratis Vavilis, Alexandru Ionut Egner, Milan Petković, and Nicola Zannone. 2016. Role mining with missing values. In *2016 11th International Conference on Availability, Reliability and Security (ARES)*. IEEE.
- [45] Rory Ward and Betsy Beyer. 2014. BeyondCorp: A new approach to enterprise security. (2014).
- [46] Zhongyuan Xu and Scott D Stoller. 2012. Algorithms for mining meaningful roles. In *Proceedings of the 17th ACM symposium on Access Control Models and Technologies*. ACM, 57–66.
- [47] Dana Zhang, Kotagiri Ramamohanarao, and Tim Ebringer. 2007. Role engineering using graph optimisation. (2007), 139–144.
- [48] Dana Zhang, Kotagiri Ramamohanarao, Tim Ebringer, and Trevor Yann. 2008. Permission Set Mining: Discovering Practical and Useful Roles. *2008 Annual Computer Security Applications Conference (ACSAC)* (2008), 247–256.
- [49] Wen Zhang, You Chen, Carl Gunter, David Liebovitz, and Bradley Malin. 2013. Evolving Role Definitions Through Permission Invocation Patterns (SACMAT '13). ACM, 37–48.
- [50] Liming Zhu, Len Bass, and George Champlin-Scharff. 2016. DevOps and its practices. *IEEE Software* 33, 3 (2016), 32–34.
- [51] Eckart Zitzler and Lothar Thiele. 1998. Multiobjective optimization using evolutionary algorithms—a comparative case study. In *International conference on parallel problem solving from nature*. Springer.
- [52] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca. 2003. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *Trans. Evol. Comp* 7, 2 (2003), 117–132.