A Survey on Access Control Deployment

Vivy Suhendra

Institute for Infocomm Research, A*STAR, Singapore
vsuhendra@i2r.a-star.edu.sg

Abstract. Access control is a security aspect whose requirements evolve with technology advances and, at the same time, contemporary social contexts. Multitudes of access control models grow out of their respective application domains such as healthcare and collaborative enterprises; and even then, further administering means, human factor considerations, and infringement management are required to effectively deploy the model in the particular usage environment. This paper presents a survey of access control mechanisms along with their deployment issues and solutions available today. We aim to give a comprehensive big picture as well as pragmatic deployment details to guide in understanding, setting up and enforcing access control in its real world application.

Keywords: access control, deployment, socio-technical system

1 Introduction

Access control is indispensable in organizations whose operation requires sharing of digital resources with various degrees of sensitivity. Innovations in business models such as cloud computing, matrix-structuring and inter-enterprise collaborations further necessitate sophisticated access management to enforce customized security policies beyond conventional office boundaries.

An effective access control system should fulfill the security requirements of *confidentiality* (no unauthorized disclosure of resources), *integrity* (no improper modifications of resources), and *availability* (ensuring accessibility of resources to legitimate users) [27]. A complete access control infrastructure covers the following three functions:

1. **Authentication**: identifying a legitimate user. The proof of identity can be what the user *knows* (password or PIN), what the user *has* (smart card), what the user *is* (biometrics), or a combination of the above (multi-factor authentication). For each of these identification methods, there exist choices of authentication schemes and protocols. A comprehensive survey of authentication technologies is available from IETF [24]; we leave this function outside the scope of this paper.
2. **Authorization**: granting or denying permission to an authenticated user to perform certain operations on a resource based on security policies. Authorization is the core of access control where most of the complexity lies, and is

the focus of this article. The process involves defining *access control policies* as rules to regulate access, choosing an *access control model* to encapsulate the policies, and implementing *access control mechanisms* to administer the model and enforce the defined controls.

3. **Accountability**: tracing or logging of actions performed by a user within the system for later auditing. This function mainly involves technical solutions (log management and security, limited automation of auditing process) and corporate management (manual inspection, crisis response), and shall not be discussed in depth here.

Technologies for these functions can be supplied by separate vendors. Existing enterprise resource management systems typically provide an authorization framework coupled with logging features, with interfaces to popular choices of authentication mechanisms (e.g., Kerberos) assumed to be already in place.

There have been comprehensive conceptual treatments of access control [27, 29, 26, 11] as well as ongoing research on the various security aspects. Despite this, effective access control remains a challenge in real world application, as it heavily involves unpredictable human factor and response to social environments that cannot be thoroughly accounted for with one-stop solutions [30]. This article therefore examines access control as a *socio-technical* system from the perspective of deployment, focusing on pragmatic issues in setting up and enforcing access control policies with flexibility to suit the social application contexts.

2 Access Control Policies and Models

Regulation of access is expressed as *policies*, which are high-level rules defined for the particular organization or project. A common policy, for example, is "separation of duties", which prohibits granting a single person access to multiple resources that together hold high damage potential when abused. Laying down the policies requires taking into account the work nature, objectives, criticality of resources handled, and so on. The policies are also dynamic as they adapt to the change in these influencing factors.

Diverse as they are, access control policies can be categorized into two based on the underlying objective. *Discretionary* access control essentially leaves access permissions to the discretion of the resource owner. These policies can be highly flexible, but weak in security. They are commonly implemented using direct, explicit identity-based mechanisms such as Access Control List (ACL), which maps individual users to individual resource permissions. On the other hand, *non-discretionary* access control regulates access through administrative action (rule-based). Examples are Mandatory Access Control (MAC) where regulations are imposed by a central authority, and policies that impose constraints on the nature of access (time, history, user roles, etc.) [11].

Such access rules are configured into control mechanisms using a *policy language*. A prevailing standard for this purpose is XACML (eXtensible Access Control Markup Language) [21], an XML-based language that supports fine-

grained access control. Proposals for new access control languages to handle particular needs continue to emerge as well [19].

Configuring access control policies is a non-trivial and highly critical process, and it should be subjected to periodic review and verification to ensure that security policies are correctly expressed and implemented. Proposed verification methods include formally testable policy specification [2], detection of anomalies or conflicting rules via segmentation technique [10], and analysis tools that enable policy administrators to evaluate policy interpretations [13].

Bridging the policies and the actual mechanisms to enforce them are *access control models*. Each model has emerged with specific concepts catering to the different needs of the different fields, but as they evolve to more extensive usage, their application domain boundaries have also blurred. The remainder of this section examines major access control models that are representative of the concepts in their category: role-based, attribute-based, and risk-based.

2.1 Role-Based Access Control (RBAC)

The RBAC [28, 7] model fits static organizational hierarchy where members have defined roles or tasks (e.g. HR Manager, Network Admin), and the roles determine the resources they need to access (e.g. payroll database, server configurations). RBAC essentially maps users to roles and roles to permissions, as many-to-many relationships (Figure 1(a)). It can be considered a higher-level form of Access Control List (ACL), which is built into all modern operating systems, and thus can be implemented on top of ACL without much difficulty. Enterprise management systems that cater to general industries typically employ some form of customized Role-Based access control, often in conjunction with their proprietary Information Rights Management technology; as seen in Microsoft SharePoint and Oracle PeopleSoft.

Aside from the basic (*core*) RBAC model, there are three extended RBAC models [11]: *hierarchical RBAC*, supporting role hierarchy and rights inheritance; *statically constrained RBAC*, supporting static constraints (e.g., on role assignment); and *dynamic constrained RBAC*, supporting time-dependent constraints (e.g., activation of roles). Many variants further refine these models for specific requirements [6, 22]. There have also been efforts to adapt RBAC for distributed environments, where multiple policy decision points need to reconcile dynamic changes in job functions as well as diverse sets of users who may not be known throughout the system [1, 31].

Administration Roles are identified and assigned permissions through the *role engineering* process, either via the *top-down* approach which takes a job function and associates needed permissions to it, or the *bottom-up* approach which takes existing user permissions and aggregates them into roles [32]. The bottom-up approach has been more popular because much of the process can be automated, giving rise to research efforts in *role mining* [8, 17].

Limitation The notion of roles, while intuitive to administer, may limit the granularity of control over resources, as users of the same role inevitably share the

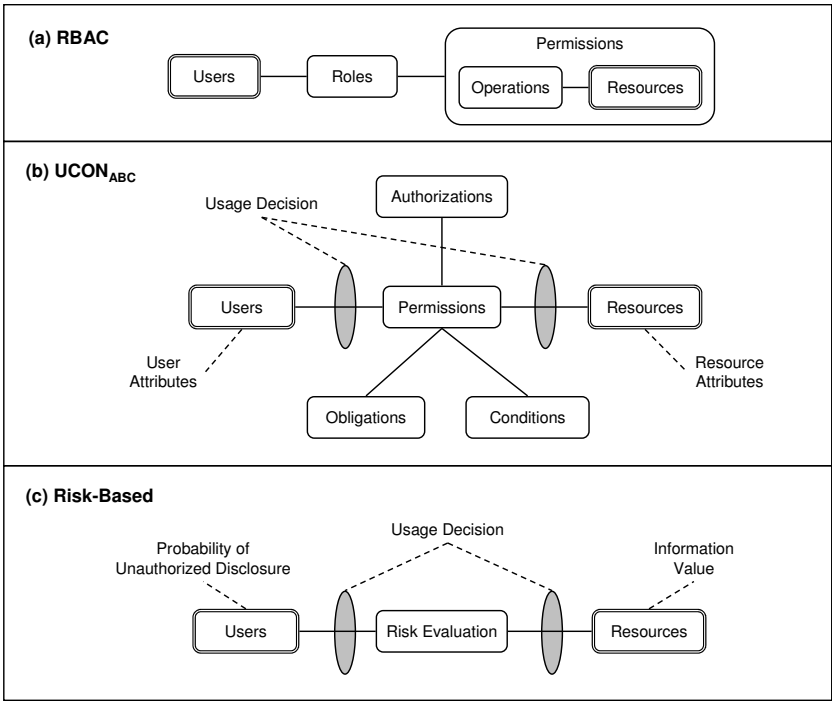

Fig. 1. Illustrated principles of access control models: (a) RBAC; (b) UCON$_{ABC}$; (c) Risk-Based.

same permissions assigned to that role. This may be mitigated via advanced role administration, e.g., by making use of inheritance in a role hierarchy, or refining static roles into workflow-centric tasks [9]. However, the principle remains that permissions are tied to roles, which have to be defined beforehand.

2.2 Usage Control (UCON$_{ABC}$)

The UCON$_{ABC}$ model [23] provides a means for fine-grained control over access permissions. It is so named because it is described in terms of ***A****uthorizations*, *o****B****ligations*, and ***C****onditions*. Fundamental to UCON$_{ABC}$ is the concept of *attributes* attached to both users and resources (Figure 1(b)). Attributes can be any information deemed relevant for granting access, such as the user's location or how many times the resource has been accessed. Permissions for a particular resource are specified in terms of conditions on attribute values: users with attribute values that meet the conditions are allowed access. Thus access rights to a resource can be assigned without needing to predict the full set of potential users. Further, attributes are mutable—they can be updated after an access (e.g., access count), and users can perform actions to fulfill the obligations necessary to access the resource (e.g., agreeing to terms and conditions). As such,

authorization may take place before or during access (on-going). An example of on-going authorization is a system where users can stay logged in by periodically clicking on an advertisement banner.

Administration Delegation of rights in UCON$_{ABC}$ is largely concerned with the question of which *authority* can set and verify attribute values that will grant access to a user. In open environments, a distributed authority is usually preferred over a central authority, where several users can assert attributes of other users and resources, with an administrator or the resource owner acting as authority root in case of conflicting assertions [25].

Limitation The expressive power of UCON$_{ABC}$ comes at the cost of complexity. Unlike for RBAC, operating system support is not readily available, thus UCON is often implemented at the application layer. It may also need database support if attributes are complex or tied to personal information. This complexity may lead to error-prone deployment in heterogeneous environments.

2.3 Risk-Based Access Control

The Risk-Based Access Control [5, 20] is motivated by highly dynamic environments where it is often difficult to predict beforehand what resources a user will need to access. In such environments, rigid access control that prevents users from accessing information in a timely manner may result in loss of profit or bad crisis response. The Risk-Based model makes real-time decisions to grant or deny a user access to the requested resource, by weighing the risk of granting the access against the perceived benefit (Figure 1(c)). In the Quantified Risk–Adaptive Access Control (QRAAC) variant [5], this risk is computed as $risk = V \times P$, where V is the information value, reflecting the sensitivity level of the resource, and P is the probability of unauthorized disclosure, reflecting the trustworthiness of the user. The security policy is then specified in terms of risk tolerance levels, which will determine the permissions given at the points of decision.

Administration The *risk assessment* process measures information value of resources, estimates probability of abuse, and sets risk tolerance levels. It is largely dependent on the organization's security objectives. Information value may be measured as costs from loss of availability (if gained access turns into a Denial of Service attack), loss of confidentiality (in case of unauthorized disclosure after access), and loss of integrity (if the resource is modified to a worse state) [18]. Probability of access abuse can be estimated by considering various scenarios enabled by existing policies [14].

Limitation Risk assessment is a subjective process that requires expertise and careful analysis. This makes Risk-Based model difficult to deploy.

3 Enforcing Access Control with Flexibility

Once the access control model and policies are set up, the underlying access control mechanisms will ensure that they are enforced in normal operation. However,

this is often not enough to guarantee the desired level of security, simply because it is hardly possible to anticipate all usage scenarios when laying down the policies. Even in models that enable access decisions to be made real-time (e.g., risk-based models), there is a lack of ability to distinguish between malicious break-in and well-intentioned infringements, such as those necessary in emergencies (e.g., a nurse taking charge of a patient while the doctor is unavailable) or those done in the best interest of the organization (e.g., an IT support staff tracing content of email attachments to resolve a crash). Access control deployment that apply over-restrictions in favor of strong security can be prone to circumvention attempts by its users wanting only "to get the job done" [30] and ironically exposed to higher risk. This problem is recognized to arise from the fact that access control is a socio-technical system for which current technical solutions are yet to satisfactorily anticipate conflicting social contexts in which they will be applied [16]. To mitigate the problem, additional mechanisms can be applied on top of the underlying access control models, to allow fine-tuned enforcement and achieve flexibility without sacrificing security.

3.1 Overriding Permissions

If it is commonly observed that truly legitimate users need to circumvent access control via offline means in order to access the resource, it may be worthwhile to set up standalone, overriding permissions that apply to these users. This may be done by tweaking the access control mechanism or adding special policies, for example, in the spirit of discretionary access control where the resource owner explicitly specifies who are allowed access [12].

This approach can achieve flexibility at relatively low risk, assuming that the overriding permissions are set by an authority with full rights over the resource (e.g., the owner). The effect is localized to information owned by the user who exercises the override option. In principle, it introduces no additional risk that is not already there (due to circumvention attempts) while providing a way for the access control exception to be properly captured in audits.

3.2 Break-Glass Mechanism

Studies of access control in the real world have shown that there often arise emergency situations that require violation of policies so that an ordinary user can gain access to critically needed resource and solve the crisis [30]. While Risk-Based Access Control model enables ad-hoc upgrade of privileges, the deployment as-is does not guarantee proper crisis response; that is, risk assessment may fail to override the decision to deny access, or the incident may not be recognized as an emergency that requires special handling. In organizations with static access control such as RBAC, the existing solution is to employ a *break-glass* strategy that will override access control decisions. The term is derived from the simple but insecure way to achieve this, which is to create a special temporary account with the highest privileges, stored in a place that a user can

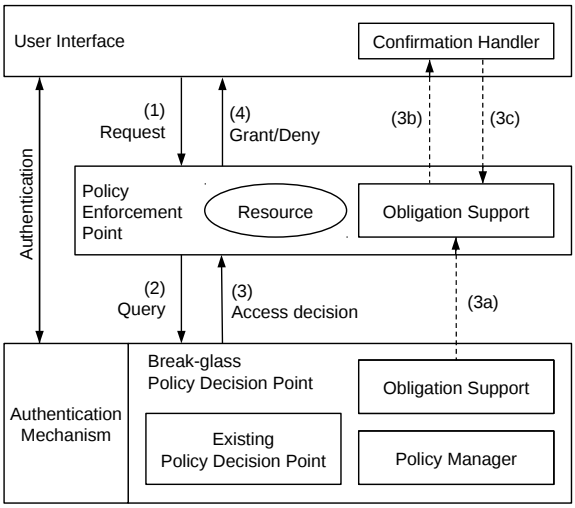

Fig. 2. The break-glass architecture and message flow [3]. Upon access request from user (1), the Policy Enforcement Point consults the Policy Decision Point (2). In normal operation, regular access decision based on existing policy is made (3) and access is either granted or denied (4). In break-glass operation, the special policy is invoked (3a) and access is granted after prompting the user to fulfill the required obligation (3b) and receiving confirmation (3c).

break into in emergencies. This practice is extremely vulnerable to misuse if the account falls into malicious hands.

A proper break-glass policy can be integrated into access control models without affecting normal operation [3], by carefully specifying how to recognize an emergency situation and allow selective access to necessary resources (Figure 2). The Rumpole model [15] uses the notions of *competence* to encode information on the user's capability to access the resource without causing harm, and *empowerment* to encode whether contextual constraints are met (e.g., whether the access will break critical policies such as separation of duties). Each can reflect one of the four values "true", "false", "conflict", or "unknown", to further provide evidence to support the access control decision.

In all cases, break-glass should be invoked along with a strict accountability function (logging and auditing), which should be made transparent to users in order to discourage abusing the permission beyond the emergency requirement.

3.3 Violation Management

Isolated situations that do not constitute an emergency may also call for violation of normal access restrictions in order to achieve a higher operational goal [16]. Suppose an IT support staff needs to troubleshoot a crash in the email application, but does not have the permission to look at the client's email ex-

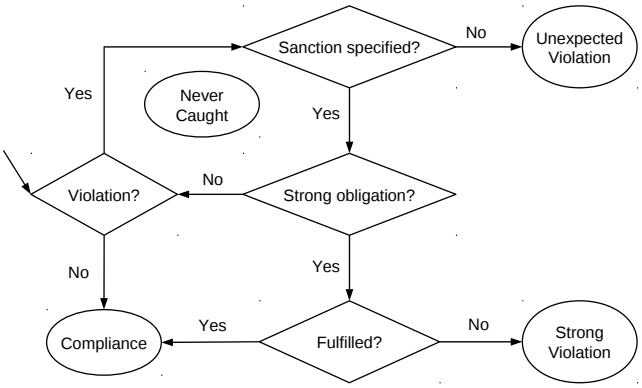

Fig. 3. Logical flow of violation management (simplified from [4]). "Compliance" captures the situation where no violation occurs, or all resulting sanctions are fulfilled. Violation occurs when a non-permitted event happens, or a violable obligation is not fulfilled. If no sanction is specified for a violation, it is concluded as "Unexpected Violation". If the sanction is a strong obligation (i.e., not violable), it must be fulfilled to achieve compliance, else it is a "Strong Violation". If the violation leads to infinite sequence of unfulfilled weak (violable) obligations, it is concluded as "Never Caught".

changes. Following the legitimate procedure of escalating the problem to higher authority might delay resolution more than simply asking the client for that permission. For security interests, such acceptable workaround should be properly accounted and audited as an "allowed" violation.

One proposed approach to manage violation in access control is to define *sanctions*, which are obligations that users must perform to justify policy violations [4]. Compliant behaviour is considered achieved as long as corresponding sanctions are applied whenever violations occur. The system can then distinguish between malicious attempts and justifiable infringements by checking whether or not sanction-obligations are fulfilled. The implementation requires the mechanism to check (1) the occurence of violation, (2) the existence of sanction corresponding to the violation, and (3) the enforcement of sanction (Figure 3).

Another violation management approach is to enhance the access control model with on-demand escalation and audit [33]. The principle is to carefully couple information access, audit, violation penalties and rewards, so that self-interested employees may obtain more information than strictly needed in order to seize more business opportunities while managing security risks responsibly.

4 Conclusion

Deployment of access control begins with defining the security policies, which may require knowledge of the concerned resources (access method, criticality, etc.), the potential users, and the nature of security breaches to prevent. Any

special requirement in the handling of specific situations, such as priority rules to apply during emergencies, should also be identified. Based on this, a suitable access control model can be selected and configured with the defined policies. Mechanisms required to support the workings of the model, including all necessary augmentations, are then installed. Each of these implementation steps should be verified to ensure that all policies are correctly put in place.

In practice, an organization may find it more hassle-free to purchase enterprise resource management systems that come bundled with access control. Factors such as migration cost and user-friendliness will then affect the choice, and it may instead adjust policies to fit the available means. Even then, the organization should ensure that its security objectives are met via careful configuration, and consider adopting additional means of security enforcement to fill any perceived gap.

We can be certain that access control, along with the challenges in enforcing it, will continue to evolve as information systems keep up with both technological advances and interaction trends. In face of this, it would seem prudent to never fully rely on any single solution, but to assume that breaches may and will happen, and to have both preventive and curative measures ready for them.

References

1. Barker, S.: Action-status access control. In: SACMAT. pp. 195–204 (2007)
2. Brucker, A.D., Brügger, L., Kearney, P., Wolff, B.: An approach to modular and testable security models of real-world health-care applications. In: SACMAT. pp. 133–142 (2011)
3. Brucker, A.D., Petritsch, H.: Extending access control models with break-glass. In: SACMAT. pp. 197–206 (2009)
4. Brunel, J., Cuppens, F., Cuppens, N., Sans, T., Bodeveix, J.P.: Security policy compliance with violation management. In: FMSE. pp. 31–40 (2007)
5. Cheng, P.C., Rohatgi, P., Keser, C., Karger, P.A., Wagner, G.M., Reninger, A.S.: Fuzzy multi-level security: An experiment on quantified risk-adaptive access control. In: 2007 IEEE Symp. on Security and Privacy. pp. 222–230 (2007)
6. Damiani, M.L., Bertino, E., Catania, B., Perlasca, P.: GEO-RBAC: A spatially aware RBAC. ACM Trans. Inf. Syst. Secur. 10 (2007)
7. Ferraiolo, D.F., Sandhu, R.S., Gavrila, S., Kuhn, D.R., Chandramouli, R.: Proposed NIST standard for role-based access control. ACM Trans. Inf. Syst. Secur. 4, 224–274 (2001)
8. Frank, M., Buhmann, J.M., Basin, D.: On the definition of role mining. In: SACMAT. pp. 35–44 (2010)
9. Fu, C., Li, A., Xu, L.: Hierarchical and dynamic security access control for collaborative design in virtual enterprise. In: IEEE ICIME. pp. 723–726 (2010)
10. Hu, H., Ahn, G.J., Kulkarni, K.: Anomaly discovery and resolution in web access control policies. In: SACMAT. pp. 165–174 (2011)
11. Hu, V.C., Ferraiolo, D.F., Kuhn, D.R.: Assessment of access control systems. Tech. Rep. NIST Interagency Report 7316, NIST (September 2006)
12. Johnson, M.L., Bellovin, S.M., Reeder, R.W., Schechter, S.E.: Laissez-faire file sharing: access control designed for individuals at the endpoints. In: NSPW. pp. 1–10 (2009)

13. Ledru, Y., Qamar, N., Idani, A., Richier, J.L., Labiadh, M.A.: Validation of security policies by the animation of Z specifications. In: SACMAT. pp. 155–164 (2011)
14. Ma, J., Logrippo, L., Adi, K., Mankovski, S.: Risk analysis in access control systems based on trust theories. In: Proc. 2010 IEEE/WIC/ACM Int'l Conf. on Web Intelligence and Intelligent Agent Technology - Volume 3. pp. 415–418 (2010)
15. Marinovic, S., Craven, R., Ma, J., Dulay, N.: Rumpole: a flexible break-glass access control model. In: SACMAT. pp. 73–82 (2011)
16. Massacci, F.: Infringo ergo sum: when will software engineering support infringements? In: FoSER. pp. 233–238 (2010)
17. Molloy, I., Li, N., Li, T., Mao, Z., Wang, Q., Lobo, J.: Evaluating role mining algorithms. In: SACMAT. pp. 95–104 (2009)
18. Nguyen, N.D., Le, X.H., Zhung, Y., Lee, S., Lee, Y.K., Lee, H.: Enforcing access control using risk assessment. In: Proc. 4th European Conf. on Universal Multiservice Networks. pp. 419–424 (2007)
19. Ni, Q., Bertino, E.: xfACL: an extensible functional language for access control. In: SACMAT. pp. 61–72 (2011)
20. Ni, Q., Bertino, E., Lobo, J.: Risk-based access control systems built on fuzzy inferences. In: ASIACCS. pp. 250–260 (2010)
21. OASIS: eXtensible Access Control Markup Language (XACML) Version 3.0. Committee specification 01, OASIS (August 2010), `http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf`
22. Ouyang, K., Joshi, J.B.D.: CT-RBAC: A temporal RBAC model with conditional periodic time. In: IPCCC. pp. 467–474 (2007)
23. Park, J., Sandhu, R.S.: The UCON$_{ABC}$ usage control model. ACM Trans. Inf. Syst. Secur. 7, 128–174 (2004)
24. Rescorla, E., Lebovitz, G.: A survey of authentication mechanisms version 7. Internet-draft, Internet Engineering Task Force (February 2010), `http://tools.ietf.org/search/draft-iab-auth-mech-07`
25. Salim, F., Reid, J., Dawson, E.: An administrative model for UCON$_{ABC}$. In: Proc. 8th Australasian Conf. on Information Security - Volume 105. pp. 32–38. Australian Computer Society, Inc., Darlinghurst, Australia (2010)
26. Salim, F., Reid, J., Dawson, E.: Authorization models for secure information sharing: A survey and research agenda. ISeCure, The ISC Int'l Journal of Information Security 2(2), 69–87 (2010)
27. Samarati, P., Vimercati, S.D.C.d.: Access control: Policies, models, and mechanisms. In: Foundations of Security Analysis and Design. pp. 137–196. Springer-Verlag, London, UK (2001)
28. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-based access control models. Computer 29(2), 38–47 (1996)
29. Sandhu, R.S., Samarati, P.: Access control: Principles and practice. IEEE Communications Magazine 32, 40–48 (1994)
30. Sinclair, S., Smith, S.W.: What's wrong with access control in the real world? Security & Privacy 8(4), 74–77 (2010)
31. Tripunitara, M.V., Carbunar, B.: Efficient access enforcement in distributed role-based access control (RBAC) deployments. In: SACMAT. pp. 155–164 (2009)
32. Vaidya, J., Atluri, V., Warner, J., Guo, Q.: Role engineering via prioritized subset enumeration. IEEE Trans. Dependable and Secure Computing 7(3), 300–314 (2010)
33. Zhao, X., Johnson, M.E.: Access governance: Flexibility with escalation and audit. In: Proc. 43rd Hawaii Int'l Conf. on System Sciences. pp. 1–13 (2010)