RiskPol: A Risk Assessment Framework for Preventing Attribute-Forgery Attacks to ABAC Policies

Carlos E. Rubio-Medrano, Ziming Zhao and Gail-Joon Ahn
The Center for Cybersecurity and Digital Forensics
Arizona State University
[crubiome,zzhao30,gahn]@asu.edu

ABSTRACT

Recently, *attribute-based access control* (ABAC) has emerged as a convenient paradigm for specifying, enforcing and maintaining rich and flexible authorization policies, leveraging attributes originated from multiple sources, e.g., operative systems, software modules, remote services, etc. However, attackers may try to bypass ABAC policies by compromising such sources to forge the attributes they provide, e.g., by deliberately manipulating the data contained within those attributes at will, in an effort to gain unintended access to sensitive resources as a result. In such a context, performing a proper risk assessment of ABAC policies, taking into account their enlisted attributes as well as their corresponding sources, becomes highly convenient to overcome *zero-day* security incidents or vulnerabilities, before they can be later exploited by attackers. With this in mind, we introduce *RiskPol*, an automated risk assessment framework for ABAC policies based on dynamically combining previously-assigned trust scores for each attribute source, such that overall scores at the policy level can be later obtained and used as a reference for performing a risk assessment on each policy. In this paper, we detail the general intuition behind our approach, its current status, as well as our plans for future work.

CCS CONCEPTS

• **Security and privacy → Access control**;

KEYWORDS

Attribute-based Access Control; Risk Management, Attribute Forgery; Policy Bypassing

ACM Reference Format:
Carlos E. Rubio-Medrano, Ziming Zhao and Gail-Joon Ahn. 2018. *RiskPol*: A Risk Assessment Framework for Preventing Attribute-Forgery Attacks to ABAC Policies . In *Proceedings of 3rd ACM Workshop on Attribute-Based Access Control (ABAC'18).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3180457.3180462

1 INTRODUCTION

Contemporary software systems have increased in size and complexity, evolving from small, monolithic, closed and proprietary infrastructures into a series of big, dynamic, distributed, heterogeneous and highly-interconnected modules that, besides providing their intended functionality as efficiently as possible, also relieve developers from fully implementing code from scratch, allowing for them to focus instead on leveraging existing solutions to better meet their needs. As an example, there is nowadays a plethora of third-party *application programming interfaces* (APIs), web services, dynamic libraries, and so on that are provided by a considerable amount of independently-run organizations, e.g., companies, institutions, government agencies, etc., thus depicting an emerging trend that is likely to stay in the foreseeable future.

In such a context, authorization policies may certainly benefit from leveraging security-related information that is provided by these sources to write rich and flexible policies that, besides meeting very specific needs, may also be evaluated and enforced in more efficient ways. With this in mind, *attribute-based access control* (ABAC) [11] has recently gained the attention of both academia and industry as a convenient way to specify, store, evaluate and enforce authorization policies by representing this security-related information as well-defined constructs known as *attributes*.

However, despite the inherent benefits introduced by this emerging approach, some security concerns still exist, as modern software infrastructures are known to be the target of attacks that leverage existing and previously-unknown security vulnerabilities [8]. Moreover, *zero-day* attacks are now becoming more frequent, leaving security officers with little or no time to respond, thus having devastating consequences [3]. In the context of ABAC policies, attackers may try to leverage vulnerabilities in third-party software to deliberately modify attributes at will, thus allowing them to bypass authorization policies and gain unintended access to protected resources as a result.

As a palliative solution, reference systems such as CVE [18] can be used to alert security officers of recently discovered vulnerabilities in commonly-used software, allowing them to perform risk assessments on possibly-affected systems, thus paving the way for counter-measures to be deployed. However, there is a need to automate such an assessment procedure as much as possible, as manual or semi-automated inspections are time-consuming and highly-error prone, thus losing efficiency and valuable time that can be leveraged by exploitation attacks.

In order to address these concerns, this paper proposes *RiskPol*, a collaborative, distributed, and automated risk assessment framework for protecting ABAC policies. Initially, numerical scores representing *trust* as a qualitative perception on the security state of a given software system, are to be assigned to third-party attribute sources. Later, these scores are transfered to the attributes they provide, allowing for a consolidated score to be calculated for each

ABAC'18, March 21, 2018, Tempe, AZ, USA

ACM ISBN 978-1-4503-5633-6/18/03…$15.00
https://doi.org/10.1145/3180457.3180462

ABAC policy, taking into account its inner *structure*, e.g., number of policy rules, boolean conditions, etc. This way, as the score assigned to the source of an attribute A changes, e.g., as a result of security vulnerabilities or attacks on the source itself, so does too the score assigned to A, thus ultimately affecting the overall risk scores of all the policies attribute A is enlisted in. In addition, the initial assignment of trust scores for attribute sources may be delegated to collaborating third-parties known as *valuators*, which may actively report the presence of security vulnerabilities and attacks and may be better suited to actively change risk scores of attribute sources that get compromised.

With all this in mind, this paper makes the following contributions: First, we explore the problem of attribute-forgery: deliberately manipulating attributes to bypass ABAC policies, which may be ignited by allowing attackers to compromise the different sources of such attributes. Second, we propose a collaborative approach, as well as a supporting framework, to mitigate immediate security incidents and zero-day attacks derived from such attribute-forgery attacks. Third, we describe a case study we are actively working on as a part of our experimental plan, which is expected to relate attributes, their corresponding sources, and their impact on assessing security risks for ABAC policies.

This paper is organized as follows: we start by providing some basic background on ABAC in Section 2 and we move on to further discuss the problem being addressed by our work in Section 3. Next, we provide a description of our approach in Section 4, and continue to propose an experimental plan, which includes implementation details, in Section 5. We revise related work in Section 6 and propose some additional ideas for future work in Section 7. Finally, Section 8 concludes this paper.

2 BACKGROUND

Attributes and Sources. In ABAC, an authorization request is granted upon the satisfaction of constraints, a.k.a., rules, involving *attributes*: properties, characteristics, or traits of subjects, objects, and even environment conditions that are relevant under a given security context [11]. Attributes are leveraged by *policy makers*, who are in charge of crafting policies by establishing relationships between attributes, *access entities*, e.g., end-users and protected resources, and access rights, commonly known as *permissions*. Following the description provided by the U.S. National Institute of Standards and Technology (NIST) [11], dedicated infrastructures may be introduced in the foreseeable future allowing for attributes to be defined, created and assigned to access entities. Such infrastructures, hereafter referred in this paper as *attribute sources*, or simply *sources* for short, may be in turn deployed by different independently-run organizations such as companies, government agencies, non-profit corporations, etc., and may be implemented as operative system modules, dedicated application software, remote services, etc. This way, a given source may provide different attributes, and may be run by a single or a conglomerate of organizations in a collaborating scheme. In addition, a given organization may run different sources at once. This way, leveraging attributes from distinct sources may greatly increase the flexibility of ABAC,

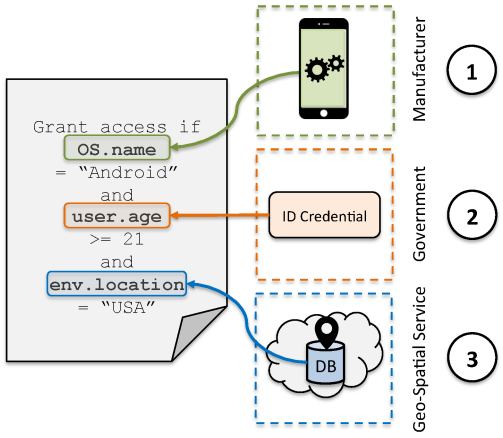

Figure 1: An ABAC policy depicting attributes from different sources. During policy evaluation time, attribute `OS.name` is provided by a device manufacturer, e.g., by means of a OS native call (1). In addition, attribute `user.age` may be in turn obtained from an ID credential issued by a local government (2). Finally, the `env.location` attribute may be retrieved from a remote Geo-Spatial service that calculates the location country based on a set of GPS coordinates (3).

e.g., easier policy specification and enforcement: no need to manually assign attributes to entities, no need for entities to hold many different attributes at once.

Defining Trust and Risk. For the purposes of this paper, we leverage the definition of trust provided by Gambetta [9]: *"Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent will perform a particular action, both before [we] can monitor such action (or independently of his capacity of ever to be able to monitor it) and in a context in which it affects [our] own action"*. In the context of ABAC policies, such definition may include a perception on the overall security state of the attribute creation and assignment processes (actions) as carried on by each source (agents). This includes any supporting software, hardware as well as any business logic that allows for the source to create and assign attributes to access entities. Such a perception may also include the way security guidelines and best practices are implemented within the organizational domains defined by the organizations running the sources.

We also leverage the definition of risk as stated by Vaughn et al. [19]: *"Risk is the probability that a particular threat will exploit a particular vulnerability of the system."* As hinted in Section 1, such a definition in the context of ABAC policies may be extended to the probability of attackers (threats) exploiting security vulnerabilities in the attribute creation and assignment infrastructures depicted by the sources (systems). In addition, we also consider the probability that, once a given attribute source has been compromised, attackers may try to manipulate its attributes at will to specifically bypass an ABAC policy (or policies) guarding sensitive resources for an specific organization.

Running Example. Fig. 1 presents a sample ABAC policy restricting access to a mobile application to end-users who are 21

years or older of age, are using a mobile phone running the Android OS, and are physically located in the United States. In such a policy, attributes are obtained from different organizational sources, each of them implementing its own attribute creation and assignment infrastructure, which is in turn protected by an independently-run security domain. During policy evaluation time, attributes may be effectively retrieved from those sources and used for policy evaluation, e.g., using a dedicated software module commonly known as *policy information point* (PIP) in the literature [14].

3 PROBLEM STATEMENT

Attribute-Forgery Attacks. In the context of ABAC, an attribute whose value can be deliberately modified without proper consent from its originating source may not provide strong security guarantees, as attackers may be allowed to modify the attribute's value at will to meet the requirements defined in a given policy, thus effectively bypassing it in unintended ways. Even in locally-run domains, attributes may be the subject of such attacks, e.g., changing file and system attributes such as names, current time, location, etc., as a result of the unintended actions carried on by dedicated attack agents, e.g., malware. Referring back to Fig. 1, attackers may try to manipulate attributes by compromising their creation and assignment infrastructures. As an example, a dedicated malware may try to intercept native OS calls such that the value of the `OS.name` is changed. Moreover, attackers may also try to compromise the remote Geo-Spatial server providing the `env.location` attribute, such that it always returns a location within the United States despite the current location of the end-user. For the purposes of this paper, we assume the evaluation and enforcement infrastructures of ABAC policies, as well as their runtime attribute-collection modules, e.g., the PIP module discussed before, stay out of reach and cannot be compromised by attackers.

Trusting Attribute Sources. Despite the inherent benefits of multiple-sourced ABAC, many approaches in the literature assume a single, always-trusted source exists or all existing sources are fully trusted all the time. However, such an assumption may not be always feasible in practice. Therefore, there is a need to provide an approach for policy makers to place a degree of trust in the attribute sources they ultimately rely on, and to estimate the level of risk for a given policy when such a perception of trust is decreased, e.g., as a consequence of the source being compromised or a new vulnerability being discovered in the attribute creation and assignment process. In the context of our running example, policy makers should be able to assess when their policies become risky as a result of any of their enlisted attributes being potentially compromised, so proper counter-measurements, to be further discussed as a part of future work in Section 7, can be deployed as a result.

4 *RISKPOL*: A TRUST-BASED RISK ASSESSMENT FRAMEWORK

As hinted in Section 3, policy makers and security officers should be allowed to maintain a perception on the security state of the attribute sources they leverage for their ABAC policies, as those sources may, in fact, be the target of dedicated attacks, or may suffer from security-related vulnerabilities affecting their attribute creation and assignment capabilities. In addition, such a perception

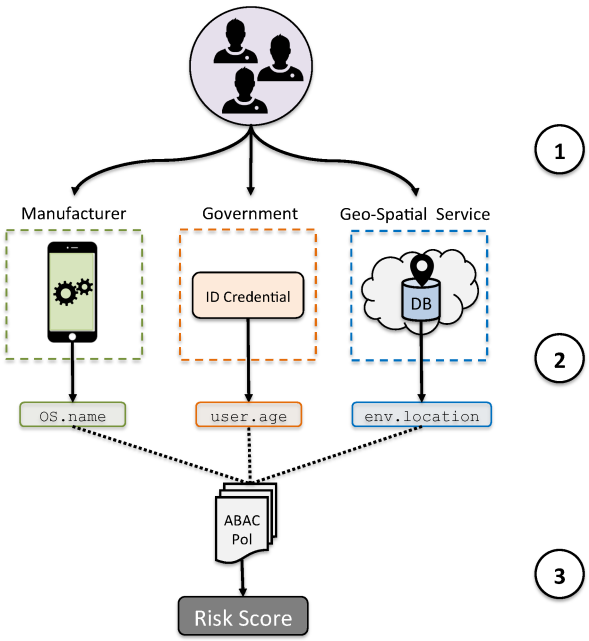

Figure 2: *RiskPol*: a framework for risk assessment for protecting ABAC Policies: an initial score is determined for attribute sources, either by risk assessors directly or in a collaborative fashion by third-party entities known as *valuators* (1). Later, such scores are forwarded to the attributes the sources provide (2), and are ultimately combined together, using a mathematical model based on the ABAC policy structure, to create a policy-level risk score (3). Policy makers may then use such policy-level scores as a foundation for customized risk assessment and the deployment of counter-measures against vulnerabilities and incidents.

should be taken into account when crafting ABAC policies, such that only attributes from good-standing sources are used, allowing for them to properly assess the risk involved when the security state of a given source is perceived to have deteriorated at a given moment of time. With this in mind, we present *RiskPol*, an automated framework allowing for both policy makers and security officers to become *risk assessors* for the ABAC policies under their control, such that unexpected security incidents can be properly addressed and mitigated, thus potentially preventing unintended access to protected resources.

Modeling Trust. Following the definition provided in Section 2, trust can be modeled as a numerical value to be defined in the context of a given implementation, which may maintain a consistent numerical scale to allow for calibrating and comparing different values of trust between distinct sources, allowing for the attributes produced by a given source to be trusted as much as the trust value assigned to it.

Source-level Scores. Following this approach, trust in attribute sources can be represented by a *source-level* score, which can be initially assigned by risk assessors. As an example, a sample trust scale may include values in the range [0, 5], being 5 the highest

score indicating *complete* trust and being 0 the lowest score denoting no trust at all (distrust). This way, referring back to Fig. 1, the device manufacturer, which provides the `OS.name` attribute, should be trusted to properly retrieve the value of the OS running on the device by means of a dedicated kernel-level service that can be queried through a native OS call. Initially, such a source may receive a trust value of 5. However, if such a source is eventually found to be affected by a newly-discovered vulnerability, e.g., it is possible to change the name of the OS as retrieved by the native call, risk assessors may change their perception of trust with respect to such a source as a result, e.g., reducing it to an intermediate such as 3 or a low trust value such as 1. Later in this section, we discuss an alternative approach for initially assigning source-level scores, which involves a set of collaborating third-parties, potentially allowing for extended reliability and convenience.

Policy-level Scores. In our *RiskPol* approach, trust is passed from sources to the attributes they provide, and from those attributes to ABAC policies. Such a process starts by retrieving the attributes enlisted in a given policy. Then, for each attribute, its corresponding source-level score is retrieved and later used to calculate an overall *policy-level* score. This idea is graphically depicted in Fig. 2. A policy-level score can be then calculated by leveraging the inner *structure* of each ABAC policy, e.g., the logical operators and the number of attribute-based rules it contains, to intelligently combine the scores obtained for each attribute. For such a purpose, an illustrative set of possible option may include, but may not be limited to, the following:

- A policy-level score can be calculated as the average of the scores obtained for each of the rules listed in the policy. The score for each rule is then calculated as the average of scores depicted by each of the attributes it contains. Leveraging our running example, and assuming an initial source-level assignment of scores of the form {(Manufacturer = 5), (Government = 4), (Geo-Spatial Service = 4)}, the trust scores for the attributes listed in such a policy can be then listed as {(`OS.name` = 5), (`user.age` = 4), (`env.location` = 4)}, allowing for the policy-level scored to be set as the average of the attributes of the unique rule contained within the policy, that is, (5 + 4 + 4) / 3 = 4.33.
- An alternative approach may include calculating the policy-level score as the average of the scores obtained for each of its listed rules, as mentioned before. However, the score of each rule can be alternatively calculated as either the maximum or the minimum value of the scores of all the attributes listed in it. Assuming the same configuration {(`OS.name` = 5), (`user.age` = 4), (`env.location` = 4)} mentioned before, the policy-level score of our running example can be then calculated as the score obtained for its unique rule, that is, max(5,4,4) = 4.
- In addition, a policy-level score can be calculated as the maximum or minimum of the scores obtained for all the rules listed in a given policy. Conversely, each policy rule score can be calculated as the average of the scores of the attributes such a rule contains.
- Policy-level scores may be also obtained by calculating the maximum or minimum score of all the rules listed in a given policy, as just mentioned, but calculating the scores of each rule as maximum or minimum of the scores assigned to its enlisted attributes, thus depicting an approach in which the overall policy-level scores is then calculated as the maximum or minimum score of all the attributes listed in the policy.
- Finally, policy-level scores may leverage the rule combination algorithms of the *extensible access control markup language* (XACML) [14], the well-known *de facto* language for authorization policies. As an example, an XACML combination scheme, e.g., deny overrides, permit overrides, etc., may be used to combine the rule-level scores of a given ABAC policy, assuming those scores where first obtained by following one of the techniques just mentioned in this section.

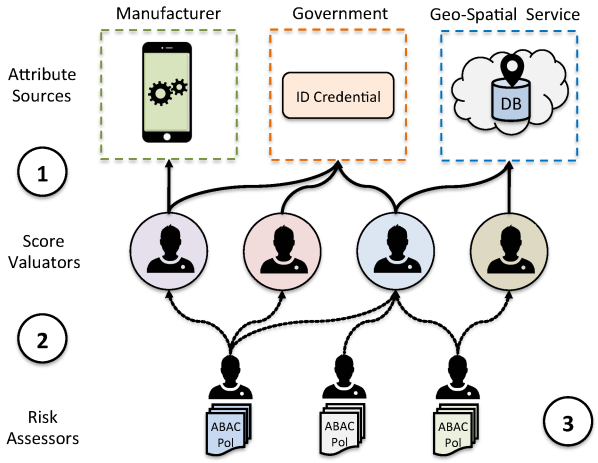

Figure 3: A graphical depiction of a stock market model for risk assessment: score valuators maintain a source-level score, which is updated depending on the perception of the current security state of attribute sources (1). In addition, risk assessors may rely on the scores provided by different valuators for the attribute sources they ultimately leverage for their ABAC policies (2). Finally, risk assessors may implement a *k-out-of-n* approach for collecting information from valuators, which may be in place for updating source-level scores and ultimately the policy-level scores of their ABAC policies (3).

As it will be discussed in Section 5, we plan to provide a experimental testbed so we can obtain evidence and further compare these combinatorial approaches for risk score calculation at the policy level.

Risk Assessment. Following the attack model described in Section 2, when an attribute is compromised, all policies referring to it become compromised too. To mitigate such a potential problem, a proper risk assessment strategy may help understand the consequences that a compromised attribute may have on a given policy, along with the resources such a policy guards. Using our policy-level scores as a foundation, risk assessors may be able to determine when a given policy becomes risky as a result of changes in the trust perception of their attribute originating sources. As an example, a policy may be deemed as risky if its policy-level score goes beyond

a predefined numeric threshold. This way, risk assessors may be able to actively react to unexpected security incidents even before they have real consequences within their local security domains. As it will be further discussed in Section 7, once a policy has been regarded as risky, a series of preventive counter-measures can be potentially deployed as a result.

As an example, assuming the initial scores for our running example have been set as {(Manufacturer = 5), (Government = 4), (Geo-Spatial Service = 4)} and {(`OS.name` = 5),(`user.age` = 4), (`env.location` = 4)} respectively, and assuming the policy-level score has been calculated as the average of the policy's attributes, that is, (5 + 4 + 4) / 3 = 4.33, a change on the source-level score of one of the attributes, e.g., the score for the `OS.name` going down to 3, may force the policy-level score to be updated to 3.67, thus below a predefined threshold, e.g., 3.75, which would then deem the policy as risky. As it will be later discussed in Section 7, as a part of our future work, we plan to explore the development of techniques for assisting risk assessors on determining score thresholds for their ABAC policies.

Score Valuators. Finally, as mentioned before, risk assessors may delegate the initial assessment of source-level scores to a series of third-party, well-deputed organizations we call *risk valuators*, or simply *valuators* for short, which, besides having updated knowledge on the attribute creation and assignment infrastructures implemented by sources, may also be able to promptly assess when a recently-discovered vulnerability, or a security incident, may have an impact on the overall trust perception of a given source. In practice, assessors may be allowed to choose n different valuators so they can implement a *k-out-of-n* strategy for score updates, e.g., allowing for k valuators to suggest a change in a given score before such a change is actually implemented. This way, risk assessors and valuators engage in a *stock market* model [4], graphically depicted in Fig. 3, in which information on updated source-level scores is distributed in proactive, expedite, and continuous ways, thus possibly improving the overall risk assessment process we have introduced in this section.

Referring back to our running example, risk assessors may rely on different valuators to provide source-level scores for the attributes they leverage. This way, when a given source is found to exhibit a serious security vulnerability, e.g., the OS kernel module retrieving the `OS.name` attribute, valuators may provide an updated trust score for it. Later, risk assessors may implement their own strategy for effectively updating their *local* source-level scores, e.g., the *k-out-of-n* approach discussed before, so they can recalculate their policy-level scores using the new data and determine if a given policy is on risk due to the security news that have just developed. Stock markets may be in turn implemented as a collection of web services provided by valuators, allowing for risk assessors to continuously query them for updates. We discuss some implementation ideas in Section 5.

5 IMPLEMENTATION AND EXPERIMENTAL PLAN

As mentioned in Section 1, we aim to provide evidence of the suitability of our *RiskPol* approach to be effectively deployed in practice. With this in mind, we are currently working towards refining and providing an implementation of the ideas discussed in Section 4. In addition, we have devised an experimental plan based on a case study leveraging realistic ABAC policies as well as a set of simulated vulnerabilities and attacks, such that our approach can be effectively evaluated using different scenarios for risk assessment.

Implementation. We plan to provide an implementation of the stock market architectural model depicted in Fig. 3. With that in mind, we aim to provide the following software components: first, we plan to develop support for implementing a risk valuator as a web service that can be queried by a series of different risks assessors. In addition, we plan to provide support for calculating the policy-level scores of ABAC policies, that is, by first extracting the attributes enlisted in them and retrieving their corresponding source-level scores from the risk valuators as just mentioned. In the context of XACML, management of such scores can be done by implementing a customized PIP module, which may continuously ping a series of previously-defined valuators for score updates. Finally, we aim to provide an initial risk assessment module for ABAC policies that may allow practitioners to obtain policy-level scores as well as to establish a set of score thresholds, which may be useful for alerting when a given policy becomes risky as a consequence of recent security events as discussed before.

Experimental Goals. We have devised the following objectives for our experimental plan: first, we aim to provide evidence of realistic ABAC policies consuming attributes provided by different sources, thus depicting the paradigm we have discussed in Section 2 and throughout this paper. Second, we aim to leverage those policies, along with the set of their corresponding attribute sources, to evaluate the approach introduced in Section 4, concretely, the different calculation techniques proposed for obtaining policy-level scores, such that we can compare them in the context of risk assessment and elaborate on their observed effectiveness and shortcomings. Finally, we aim to encourage the deployment of our approach in practice by providing some insight on existing and future attribute sources, the implementation of risk valuators, as well as some general considerations for risk assessment techniques in the context of ABAC policies.

Case Study. In order to achieve these goals, we aim to perform a case study leveraging a set of ABAC policies collected from the literature [20], as well as from realistic authorization scenarios that may be better served with the ABAC paradigm. Later, for each of the collected policies, we aim to identify the set of attributes that may better fit them, as well as their possible attribute sources, which should include the identification of the software infrastructure that may be in place for providing the attribute, e.g., OS modules, external software, web service, etc. As a subsequent step, we plan to incorporate the information obtained from those sources into our implementation of a stock market and perform a series of experiments simulating a change in trust scores based on a set of case scenarios simulating security incidents and the discovery of vulnerabilities. As an example, we may take existing reports on security vulnerabilities listed as CVE entries as an inspiration for developing a set of case scenarios involving affectations to ABAC policies, e.g., an attribute being potentially compromised as a result of a newly-discovered vulnerability. In addition, we aim to leverage this methodology to evaluate the policy-level calculation techniques listed in Section 4, and to obtain some insight into the

development of additional techniques we may have not devised in this paper. Finally, we aim to create a catalog of attributes, sources, valuators, and trust scales/scores for risk assessors to choose from, which, besides serving as a convenient platform for deploying our approach in practice, can be also used as a foundation for the ideas for future work we detail in Section 7.

6 RELATED WORK

Risk Models for Authorization. Previous work has focused on incorporating risk models for dynamic permission assignment based on the current risk state of the system. This way, before assigning a permission to an end-user, the system evaluates the risk of doing so based on recent (possibly real-time) information [7]. Following this paradigm, Ni et al. [13] introduced an approach leveraging a fuzzy engine for estimating risk before releasing a permission to a given user. In the context of role-based access control (RBAC) [17], Bijon et al. [2] proposed an extension to the core RBAC model that includes a so-called *risk-threshold* as a part of RBAC user *sessions*, allowing for a given session to be dropped in case the calculated risk value goes beyond a predefined threshold, thus potentially preventing user-based abuse of already-authorized permissions. Similarly, Chen et al.[6] presented an approach leveraging XACML as the policy language for expressing RBAC policies, extending the language with specific construct to model risk as well. In the context of attribute-based authorization, Kandala et al. [12] presented an approch combining the concept of risk and ABAC, developing a model based on UCON [15] extensions. Our *RiskPol* approach is also intended to incorporate a perception on the security state of a given system before an authorization policy can be evaluated and enforced, such that the system becomes aware of recent events, e.g., vulnerabilities or incidents, that may have an impact on the overall authorization process. However, our approach is intended to protect policies themselves from attacks that may originate from compromised attributes, instead of evaluating risk for each user in isolation before or during the time the authorization process takes place. In our approach, a risk assessment performed over a whole policy may affect all users being served by it, e.g., by applying one of the proactive actions discussed in Section 7. Therefore, our policy-level approach differs from previous approaches on risk assessment that calculate risk at the user-level only.

Credential-based Risk Analysis. Risk assessment approaches have been also proposed for credential-based access control. As an example, Chapin et al. [5] introduced a trust management logic that provides formal risk assessment by associating risk levels with authorization elements, allowing for tolerable levels of risk to be rigorously enforced. In addition, Goodrich et at. [10] provided a solution for an authenticated dictionary for attribute-based credentials, allowing for attribute sources to collectively publish information to a common repository, which can be later queried by other parties through the network. While these approaches have influenced to our *RiskPol* approach, ABAC comprises a wider model that may include credentials as an implementation subset. As an example, credentials may be used to securely communicate attributes between sources and policy evaluation engines. Also, they may provide proof of a correct attribute-user assignment as stated by the source. Moreover, ABAC provides a wider theoretical model in which attributes may be also retrieved by other implementation strategies. As an example, while certain attributes may benefit from a cryptography-based protection while in transit, some implementations, e.g., an XACML PIP module, may simply retrieve the attribute directly from sources by implicitly trusting both the communication channel, e.g., a native OS call, as well as its originating source (the OS), as it is depicted in the `OS.name` attribute included in our running example, shown in Fig. 1.

Distributed Risk Assessment and Attribute Dictionaries. Finally, Aven [1], introduced a risk assessment framework modeling both security and safety in the concept of information technology infrastructures. As with our architectural approach depicted in Fig. 3, information about security incidents is generated by a set of trusted partners and distributed actively to remote enterprise, thus allowing for the fast and efficient dissemination of security-related issues. Our *RiskPol* approach is inspired by such a concept as it also relies on strong collaborative settings for sharing information that can be valuable for risk assessment. However, our approach goes a step further by providing means for each risk assessor to automatically calculate policy-level scores by leveraging the information previously-shared by the risk valuators, thus providing the foundations for an automated approach for efficient and expedited risk assessment.

7 FUTURE WORK

Besides refining the approach introduced in Section 4, and performing the experimental plans we also discuss in Section 5, we have envisioned some paths to explore in the future as a continuation of the work detailed in this paper.

Policy Enhancement. An alternative approach would include *enhancing* the now-risky policy with additional attributes or rules, such that the overall trust score goes back to a value above the risk threshold. As an example, previous approaches have already explored the possibility of dynamically enhancing authorization policies with additional attributes that are automatically retrieved from users [16]. A generalization of these schemes may include replacing the now-risky policy completely for another one, e.g., all attributes in the new policy do not appear in the now-risky one, until a proper source-level score is restored. Obviously, these actions may need to be carefully crafted beforehand, taking into account assignments of users-permissions-resources such that no usability issues are introduced by these changes, e.g., a user may be denied access to a resource because he/she fails to be assigned a newly-introduced attribute.

Determining Risk Score Thresholds. As introduced in Section 4, our *RiskPol* approach depicts a basic risk assessment approach that is mostly based on the calculated policy-level scores of ABAC policies. While convenient, such a scheme still requires risk assessors to manually determine a proper risk score threshold for each of the policies under their control. With this in mind, we plan to provide a solution to assist risk assessors on calculating proper thresholds for their policies. As an example, a future approach may inspect the inner structure of an ABAC policy, e.g., number of rules, attributes per rule, etc., to better determine which attributes are more relevant when it comes to granting access to a protected resource. Later, the source-level scores for such attributes

may be taken as a reference for determining a proper policy-level score threshold. In addition, we aim to incorporate more advanced models for risk assessment into our approach. As an example, it may be possible that only a subset of the attributes produced by a given source may be on risk due to a vulnerability discovered in a recent version of the attribute creation infrastructure. In such a scenario, only the attributes produced since the last infrastructure update may need to have their source-level scores updated, leaving all others untouched. Such a scheme may rely on external metadata information on attribute creation to be provided by sources and being maintained by the ABAC policy evaluation infrastructure.

Advanced Risk Assessment Models. In addition, as stated in Section 2, a risk assessment model must consider the probability that, once a given attribute source has been compromised by attackers, they will try to specifically target the set of ABAC policies protecting the resources of a given organizational entity. In our *RiskPol* approach, as described in Section 4, we have taken a rather conservative approach by assuming that every single ABAC policy whose enlisted attributes are compromised may automatically become at risk, that is, the probability of attackers attacking any of those policies is the same. With this in mind, future risk assessment models may consider adding extra parameters to allow for assessors to introduce the probability that attackers may try to specifically target their organizational ABAC policies, assuming a previous security vulnerability or incident within their corresponding attribute sources has been found, thus possibly producing a more customized and accurate assessment as a result.

8 CONCLUSIONS

In this paper, we have discussed the problem of bypassing ABAC policies by deliberately manipulating the attributes listed on them. Such a problem is aggravated by the fact ABAC policies are expected to consume attributes originated from many different, independently-run sources. In order to address this problem, we have presented an ongoing work on providing a collaborating and distributed framework, called *RiskPol*, which is intended to properly assess the risks involved in trusting the attribute creation and assignment infrastructures provided by heterogeneous sources, such that attribute-forgery attacks can be effectively mitigated. As a next step, we aim to refine the general approach we have discussed in Section 4, and complete the experimental process we have detailed in Section 5. Finally, we plan to publish our results later this year.

ACKNOWLEDGMENTS

This work was partially supported by a grant from the National Science Foundation (NSF-IIS-1527268) and by a grant from the Center for Cybersecurity and Digital Forensics at Arizona State University.

REFERENCES

[1] Terje Aven. 2007. A unified framework for risk and vulnerability analysis covering both safety and security. *Reliability Engineering and System Safety* 92, 6 (2007), 745 – 754.

[2] Khalid Zaman Bijon, Ram Krishnan, and Ravi Sandhu. 2012. *Risk-Aware RBAC Sessions.* Springer Berlin Heidelberg, Berlin, Heidelberg, 59–74.

[3] Leyla Bilge and Tudor Dumitras. 2012. Before We Knew It: An Empirical Study of Zero-day Attacks in the Real World. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS '12)*. ACM, New York, NY, USA, 833–844.

[4] Katherine Campbell, Lawrence A. Gordon, Martin P. Loeb, and Lei Zhou. 2003. The Economic Cost of Publicly Announced Information Security Breaches: Empirical Evidence from the Stock Market. *Journal of Computer Security* 11, 3 (April 2003), 431–448.

[5] Peter Chapin, Christian Skalka, and X. Sean Wang. 2005. Risk Assessment in Distributed Authorization. In *Proceedings of the 2005 ACM Workshop on Formal Methods in Security Engineering (FMSE '05)*. ACM, New York, NY, USA, 33–42.

[6] Liang Chen, Luca Gasparini, and Timothy J. Norman. 2013. XACML and Risk-Aware Access Control. In *Proceedings of the 10th International Workshop on Security in Information Systems (ICEIS 2013)*. 66–75.

[7] Nathan Dimmock, András Belokosztolszki, David Eyers, Jean Bacon, and Ken Moody. 2004. Using Trust and Risk in Role-based Access Control Policies. In *Proceedings of the Ninth ACM Symposium on Access Control Models and Technologies (SACMAT '04)*. ACM, New York, NY, USA, 156–162.

[8] Katheryn A. Farris, Sean R. McNamara, Adam Goldstein, and George Cybenko. 2016. A preliminary analysis of quantifying computer security vulnerability data in the wild. (2016), 9825 - 9842 pages.

[9] Diego Gambetta. 1988. Can We Trust Trust?. In *Trust: Making and Breaking Cooperative Relations*. Basil Blackwell, 213–237.

[10] Michael T. Goodrich, Michael Shin, Roberto Tamassia, and William H. Winsborough. 2003. *Authenticated Dictionaries for Fresh Attribute Credentials.* Springer Berlin Heidelberg, Berlin, Heidelberg, 332–347.

[11] Vincent C Hu, David Ferraiolo, Rick Kuhn, Adam Schnitzer, Kenneth Sandlin, Robert Miller, and Karen Scarfone. 2014. Guide to attribute based access control (ABAC) definition and considerations. *NIST Special Publication* 800 (2014), 162.

[12] S. Kandala, R. Sandhu, and V. Bhamidipati. 2011. An Attribute Based Framework for Risk-Adaptive Access Control Models. In *2011 Sixth International Conference on Availability, Reliability and Security*. 236–241.

[13] Qun Ni, Elisa Bertino, and Jorge Lobo. 2010. Risk-based Access Control Systems Built on Fuzzy Inferences. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS '10)*. ACM, New York, NY, USA, 250–260.

[14] OASIS Standard. 2013. eXtensible Access Control Markup Language (XACML) Version 3.0. (2013, January 22). (2013). http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html.

[15] Jaehong Park and Ravi Sandhu. 2004. The UCONABC Usage Control Model. *ACM Trans. Inf. Syst. Secur.* 7, 1 (Feb. 2004), 128–174.

[16] Carlos E. Rubio-Medrano, Josephine Lamp, Adam Doupé, Ziming Zhao, and Gail-Joon Ahn. 2017. Mutated Policies: Towards Proactive Attribute-based Defenses for Access Control. In *Proceedings of the 2017 Workshop on Moving Target Defense (MTD '17)*. ACM, New York, NY, USA, 39–49.

[17] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. 1996. Role-Based Access Control Models. *Computer* 29, 2 (Feb. 1996), 38–47.

[18] The MITRE Corporation. 2017. Common Vulnerabilities and Exposures. (2017). https://cve.mitre.org/

[19] R. B. Vaughn, R. Henning, and A. Siraj. 2003. Information assurance measures and metrics - state of practice and proposed taxonomy. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2003.* 10.

[20] Zhongyuan Xu and Scott D Stoller. 2015. Mining attribute-based access control policies. *IEEE Dependable and Secure Computing* 12, 5 (2015), 533–545.