POSTER: Using Quantified Risk and Benefit to Strengthen the Security of Information Sharing *

Weili Han[1,2], Chenguang Shen[1], Yuliang Yin[1], Yun Gu[1], Chen Chen[1]
{wlhan, 08301010162, 08302010056, 07302010076, chenc}@fudan.edu.cn
1. Software School, Fudan University
2. Key Lab of Information Network Security, Ministry of Public Security
Shanghai, P. R. China

ABSTRACT

Risk and benefit are two implicit key factors to determine accesses in secure information sharing. Recent researches have shown that they can be explicitly quantified and used to improve the flexibility in information systems. This paper introduces the motivation and a technical design of Quantified riSk and Benefit adaptive Access Control (QSBAC) to strengthen the security of information sharing. The paper also introduces the key issues to design policies in QSBAC.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection - Access Controls; D.2.9 [**Software Engineering**]: Software Configuration Management

General Terms

Security, Management

Keywords

Quantified Risk, Quantified Benefit, Secure Information Sharing, QSBAC

1. INTRODUCTION

With the development of the information technologies, including cloud computing and social network service systems, information is currently being shared on a larger scale. People can almost access and share their favorite information in any place, at any time and with any device.

However, one of the most important challenges of information sharing is how to assure its security [6]. That is, the parties sharing the information should be protected according to security policies which could be preset explicitly, though policies are sometimes hidden in application scenarios. For example, in case of personalization recommendation, if a user is willing to share more private data, he or she will obtain a more accurate result. But, the risk of the leakage of private data will also arise. In this case, the policies are hard to be preset.

Quantified risk and benefit can be used to strengthen the security of information sharing, because:

- Each security policy implicitly considers risk and benefit. Risk [1][3] and benefit [7] are usually two key factors in security, and hidden in each step of the access control processes. If we use a policy, where *a user allows a website to know his or her age*, as an example, the user obliviously will accept all risks and benefits brought by the website's access to his or her age data. The acceptance could be determined because the risk is negligible, e.g. the website is an online banking site; or because the benefit is big enough, e.g. the website is a social network service site where the user can share information with his or her active friends.

- It is suitable to deal with emergent and dynamic applications where risk and benefit aware policies can be explicitly set [1][3][2]. If quantified risk and quantified benefit of an access request can be measured, the decision result will be supported by more directed evidence. This is especially important in emergent or dynamic application scenarios, where the preset policies cannot meet all application scenarios. Usually, the undefined application situations will lead to the use of a default policy, for example, *reject all requests for confidentiality or integrity* or *allow all requests for availability*. Obviously, such default policies are not very satisfying. Risk and benefit aware policies can make decision result based on explicit measurement of risk and benefit, which is a promising way to manage dynamic application scenarios.

This paper introduces a technical design of Quantified riSk and Benefit adaptive Access Control (QSBAC), discusses the key issues of the QSBAC policies, which include how to use fuzzy set and fuzzy logic, and evaluates the policy engine of QSBAC.

The rest of this paper is organized as follows: section 2 introduces the steps to enforce QSBAC; section 3 discusses the key issues to design the policies of QSBAC; section 4 evaluates the policy engine of QSBAC; section 5 investigates the related work; and section 6 summarizes the paper.

2. STEPS TO ENFORCE QSBAC

QSBAC considers two types of quantified risk: risk of allowing access (RAA) and risk of denying access (RDA), and two types of quantified benefit: benefit of allowing access (BAA) and benefit of denying access (BDA).

The process of enforcing QSBAC consists of the following steps:

*A full version of this paper is available at http://old.homepage.fudan.edu.cn/w̃lhan/paper/qsbac.pdf

CCS'11, October 17–21, 2011, Chicago, Illinois, USA.
ACM 978-1-4503-0948-6/11/10.

S1: *Set Risk Mitigation Actions (RMA) and Benefit Incentive Actions (BIA)*: Risk mitigation actions refer to how actions or modules in the system reduce the risk when the information sharing access request[1] is allowed or denied, and benefit incentive actions refer to how actions or modules in the system stimulate the benefit when the request is allowed or denied.

S2: *Set quantified risk and benefit adaptive policies*: A QSBAC-enabled system will be policy-driven. That is, security administrators will preset QSBAC policies. Note that, the QSBAC policies are designed not to replace traditional security policies, such as role-based access control policies, but to be a supplement of them to deal with the undefined situations, especially emergent or dynamic application situations.

S3: *Quantified risk and benefit measurements*: The four variables (RAA, RDA, BAA and BDA) will be measured by preset measurement functions. Due to the complexity of processing the measurements and its domain-specific feature, this paper does not discuss the details. Note that, during the measurements applicable risk mitigation actions and benefit incentive actions will be considered.

S4: *Determine a request according to the policies and measurements*: When a request comes, the PDP (Policy Decision Point) will evaluate the request and return a result which includes a decision (allow or deny the request), necessary risk mitigation actions, and necessary benefit incentive actions.

S5: *Implement the reponse with risk mitigation actions and benefit incentive actions*: Traditionally, if a request is allowed, a secure system will implement the request; Otherwise, the secure system would do nothing. But in a QSBAC-enabled secure system, the system would implement the risk mitigation actions and benefit incentive actions included in the decision result. The actions could be transactional and long-term. Therefore a more complex session control mechanism is required for secure information sharing.

S6: *Record the runtime context as a data source to optimize the risk mitigation actions, the benefit incentive actions, the policies and the measurement functions of quantified risk and quantified benefit.*

3. POLICIES OF QSBAC

We extend XACML (eXtensible Access Control Markup Language)[5] to express policies in QSBAC. XACML is becoming a standard of access control policy language. XACML mainly consists of the following tag elements: <PolicySet> mainly consists of sub policy sets, policies, and obligations; <Policy> mainly consists of the following elements: a <Target>, a set of <Rule>, a rule-combining algorithm-identifier, an <ObligationExpressions>, an <AdviceExpressions>; and the <Rule> element mainly consists of the following elements: a <Target>, an <Effect>, a <Condition>, an <ObligationExpressions>, an <AdviceExpressions>.

Our extension, referred as to QSBAC-XACML, includes the following extra elements:

[1]For simplicity, we use 'request' to represent 'Information sharing access request' in the rest of this paper.

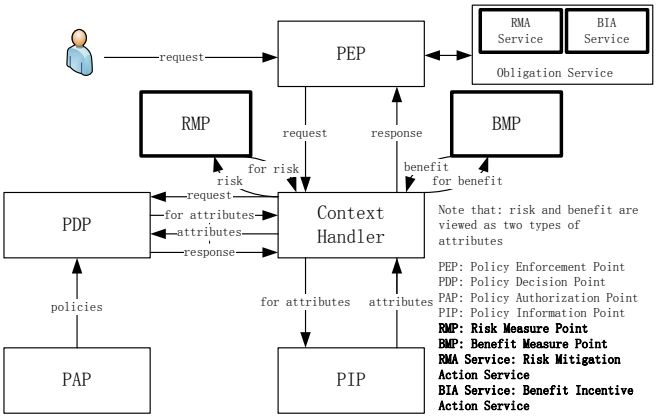

Figure 1: Dataflow Diagram of QSBAC-XACML

- Four variables: RAA, RDA, BAA, BDA, the corresponding measurement functions, and membership functions of fuzzy sets.
- Two types of obligation services: Risk mitigation actions and benefit incentive actions.
- The <FuzzyRule> element, which consists of <FuzzyConditions>.
- The <FuzzyConditions> element in the <FuzzyRule> element. The following code is an example of the <FuzzyConditions> element:

```
<FuzzyConditions
  InferenceAlgorithmIdentifier="an:infer:MINMAX">
  <FuzzyCondition type="RAA" value="high" />
  <FuzzyCondition type="BDA" value="low"/>
</FuzzyConditions>
```

  This element represents a policy which holds when the RAA is high, BDA is low, then the effect (allow or deny) on the target of <FuzzyRule> will work. It also suggests that the fuzzy inference model is "MINMAX".

Note that, we use fuzzy sets rather than thresholds to define the means of measured quantified risk and benefit. The main difference between the two methods are as follows: if a fuzzy set based method points out an assigned variable is *high*, that means the assigned variable, e.g. 29°C of temperature, belongs to the *high* fuzzy set, and a percentage of membership, e.g. 70% *high*, will also be assigned according to a membership function. But the threshold based method will only return the result *high* with no percentage of membership. Moveover, after assigning the percentage of membership, the system will infer a result according to the fuzzy logic rather than boolean logic.

4. PROTOTYPE AND EVALUATION

Figure 1 shows the data flow of QSBAC-XACML, which is extended from XACML. Here, we add the following new features:

- RMP and BMP: RMP is responsible for measuring risk (RAA and RDA), whereas BMP is for measuring benefit (BAA and BDA). In QSBAC quantified risk and

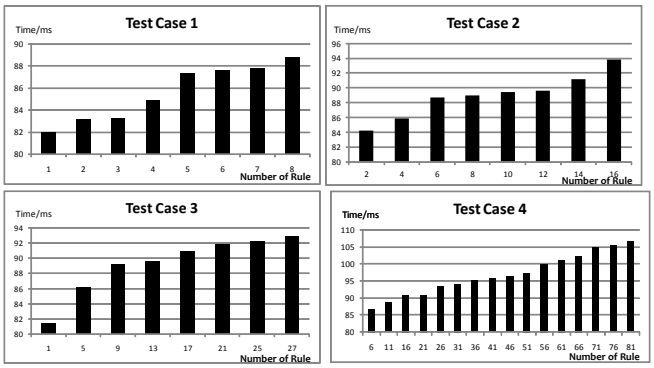

Figure 2: Performance Evaluation

benefit are viewed as two types of attributes in the data flow.

- Two types of obligation services: RMA Service and BIA Service. RMA Service enforces the risk mitigation actions in the response, whereas BIA Service enforces the benefit incentive actions in the response.

The work flow of QSBAC-XACML is as follows. The PEP (Policy Enforcement Point) is responsible for acquiring an access request and transferring it to the Context Handler. Then the PDP (Policy Decision Point) is invoked by the Context Handler to make a decision for the request. The PDP may retrieve necessary attributes from the Context Handler for evaluation, and finally return the decision result to the request. Afterwards the Context Handler will return the result back to the PEP. Meanwhile, the PEP will enforce the result and, if necessary, collaborate with the Obligation Service to enforce obligations.

Based on Enterprise XACML Implementation [4], we implemented a prototype.

We designed four test cases to evaluate the performances of the inference engine in the PDP, and the results are shown in Figure 2, where X axis of each diagram refers to the number of fuzzy conditions, and Y axis refers to the time the PDP responses to a request. We can conclude that the prototype of QSBAC is feasible both in design and performance. There can be at most three fuzzy sets (*high*, *middle*, *low*) and four variables (RAA, RDA, BAA, BDA). Because the number of variables and fuzzy sets can be cut out, we choose two fuzzy sets for each three variables in Test Case 1 (2^3=8); two fuzzy sets for each four variables in Test Case 2 (2^4=16); three fuzzy sets for each three variables (3^3=27) in Test Case 3; and three fuzzy set for each four variables (3^4=81) in Test Case 4.

5. RELATED WORK

Risk is always a hot spot in the field of information security. The JASON report[3] introduced a road map to leverage risk to enlarge and assure horizonal integration in information sharing. Cheng et al. [1] proposed FuzzyMLS where they used quantified risk as a factor to improve the flexibility of multi-level security systems. Han et al. [2] used quantified risk to strengthen the security of workflow systems.

Besides the above work, benefit is also researched to enlarge and assure the information sharing. Zhang et al. [7] proposed BARAC where risks of information leakage and benefits of information sharing are balanced. But they only used boolean logic. Moreover, the risks and benefits organized as vectors must be comparable and the system is required to be profitable in their method. This assumption would limit their applicable scenarios.

Different from the above works, QSBAC introduces the four quantified variables, and uses fuzzy set and fuzzy logic to infer a decision based on risk and benefit variables, risk mitigation actions, benefit incentive actions, and preset risk and benefit adaptive policies. In QSBAC, the measured risks and benefits only meet the requirements of the memberships of fuzzy set, and are not necessarily comparable among them.

6. CONCLUSION AND FUTURE WORK

This paper introduces QSBAC where we use four variables, RAA, RDA, BAA, BDA, as main factors to determine an information sharing request. We analyze the motivation of QSBAC, propose how to design the policies of QSBAC, and introduce the prototype and performance evaluation. The analysis, prototype and performance evaluation show QSBAC can help security administrators to easily deal with undefined situations in emergent or dynamic application scenarios, and the performance is promising.

In future, we will introduce QSBAC into several application scenarios, including dynamic network management and identity combination in online banking.

Acknowledgement

This paper is supported by the 863 high-tech project (Grant NO: 2011AA100701) and Key Lab of Information Network Security, Ministry of Public Security (Grant NO: C11601). Thank Mrs. Min Li for her English polish.

7. REFERENCES

[1] P. Cheng, P. Rohatgi, C. Keser, P. A. Karger, G. M. Wagner, and A. S. Reninger. Fuzzy multi.level security: An experiment on quantified risk adaptive access control. In *SP'07*, pages 222 – 230, CA, USA, May 2007. ACM.

[2] W. Han, Q. Ni, and H. Chen. Apply measurable risk to strengthen security of a role-based delegation supporting workflow system. In *POLICY 2009*, pages 45–52, 2009.

[3] JASON. Horizontal integration: Broader access models for realizing information dominance. Technical Report JSR-04-132, MITRE Corporation, http://www.fas.org/irp/agency/dod/jason/classpol.pdf, 2004.

[4] Ppzian. Enterprise xacml implementation. In *http://sourceforge.net/projects/java-xacml/*, 2008.

[5] E. Rissanen. Extensible access control markup language (xacml). *OASIS Standard*, April 2009.

[6] M. Srivatsa, D. Agrawal, and S. Reidt. A metadata calculus for secure information sharing. In *CCS'09*, IL, USA, 2009.

[7] L. Zhang, A. Brodsky, and S. Jajodia. Toward information sharing: Benefit and risk access control (barac). In *POLICY 2006*, pages 45–53, London, Ontario, Canada, June 2006.