Context Aware Role Based Access Control Using Security Levels

Michal TRNKA

Dept. of Computer Science and Engineering, Czech Technical University, Technick a 2, 166 27 Praha, Czech Republic

trnkami1@fel.cvut.cz

Abstract. *Security of applications is very challenging and extensive problem. To keep with the the trend of personalized applications and with increased interest in context aware applications it needs to be adapted for those applications. It this paper I present how to add context awareness elements into role based access control. I describe already existing solutions and point out their key ideas and I propose own solution, which is very lightweight, universal and allows instant enhancement of current RBAC even in current applications. The solution is based on security levels which are assigned to users based on context. Security levels represents how the users can be trusted and are determined during users login procedure. The levels are used as additional security constrain so to access resources in application user need to have not only right permission granted through roles, but also to have corresponding level.*

Keywords

Role based access control, Context-aware security, Security, Security levels.

1. Introduction

Contemporary applications move toward context-awareness [1, 2]. It is caused by emerge of the large amount of the mobile technologies [3] and users demand for personalized applications. Applications provide personalized context based on user's context or the application's context [4]. That brings completely new experience for the applications operators as well as for users. However securing the applications is done the old way. Usually users are assigned various roles in applications or permissions for resources and those security rules are independent on context. There is very few, if any, applications, which has security based on context. We can expect that users and application owners would take advantage of application security based on context to provide specific security rules based on users context.

Applications using CA security can be much less obtrusive for users. They can be asked for different authentication methods based on context, they can be authorized for same resource various ways depending on their context. They can even sometimes omit authentication because their context is trustworthy by itself (for example access from inner company network). Same as the users can profit from the context based authentication operators of the applications. They might define more strict security rules for suspicious users behavior (for example access to confidential resources in system from internet in night). Using context allows system administrators for more fine grained security rules, which would be otherwise unsustainable for maintenance.

Application operators and software developers are good aware of the added value of CA security. Even there are various proposals how to do CA security none of them is widely used. Reason why they are not widely used can be that they are either too complicated or they are too innovative and it is hard to incorporate them into existing solutions.

In this paper I will present solution, which extends standard role based security architecture with CA elements. This extension is based on giving users security level based on their context. Resources would require user to posses that level in addition to his normal rights to be accessed. That way we can extend any existing security architecture with CA elements.

2. Background

Large applications or information systems, which has more then one user needs some form of authentication and authorization. Such systems are here for many decades and are almost as old as computers itself. For example proposal for information system called Memex is mentioned in 1945 [5]. Therefore for many decades there exists problem with security of applications and it was solved various ways.

Two of the oldest principles for securing application resources is mandatory access control (MAC) [6] and discretionary access control (DAC) [6]. Those two principles does not define how the application security should be implemented, but rather define core principles in authorization. In MAC there is some authority, which grants permissions to all resources. In DAC on the other side can grant permission everyone with sufficient permission for the resource.

However granting permissions to every user in system is unpractical for larger amount of users. Role based access control (RBAC) [7] provides another level of abstraction. It that approach the permission is given to an abstract role and users are assigned roles. Typically there is in application many times less roles then permissions and the roles does not change significantly over time unlike users.

Nevertheless those authorization principles and method are very static. They does not reflect the changing state of the system and users. Once they are set they do not take in account any other factors and any fine tuning is if not impossible very difficult.

CA security can overcome those difficulties and provide even new experience for user and application operator. CA applications are much more personalized then the static ones and same comes for the security. Application can get a lot of information about user from context and therefore it does not annoy user with request for additional information. For example application knows his IP address and therefore does not need to re-verify him. The context is also valuable source of information for application owner. For example he might restrict some IP ranges, days of time, etc to access resources in application.

Also with the emerge of the context aware application there is naturally need of CA security. Ideas about CA applications are here since 90's [8]. Such CA applications adapts according to the location and time of user, the collection of nearby people, hosts, accessible devices, etc, as well as to changes to such things over time. A system with these capabilities can examine the user's environment as well as computing environment and react to changes to environment. Naturally such application needs security architecture which adopts all of the above mentioned principles to be fully CA. However there is now significant lack of standardized methods or best practices how to address problem of the CA security.

To illustrate how can context aware security improve applications consider following example. We have a information system in company. To make it more comfortable let users from inner company network access some noncritical resources. But if the user comes from internet or access some sensitive resources he needs to authenticate itself normal way. But not only users would benefit from it. CA security can determine suspicious users behavior. For example if users log in to system in short time frame from different parts of world it can rise flag and the incident can be investigated further. Or the company can set some access hours for various resources, to limit possibility of their abuse (for example limit access in non working hours).

3. Related Work

There has been multiple attempts to extend classic role based access control with context-aware elements and make RBAC more fine-grained.

One of the approaches is to add another set of roles to role based access control. Moyer [9] proposes creating two additional sets of object roles and environmental roles and tying permissions with trio of roles. Covington [10] simplify that to just one additional set of environmental roles. They are hierarchical composed and represent current state of system. Similarly to this approach Seon-Ho [11] suggests additional set of context roles.

Different solution proposes Sladić [12]. Roles are granted to user after his authentication based on context. That way user can obtain new roles based on context. The idea is further developed by Kulkarni [13] into Context-Aware RBAC. It also allows roles to be granted based on context but there is second layer of authorization architecture, which is responsible for granting and revoking roles when the context changes and therefore roles are dynamically reflecting context.

There is also possibility to solve that problem with adding another element not based on roles. Neumann [14] suggests adding context constrains to security policies. When the permission is checked used needs to posses not only the permission for resource (based on his role) but also fulfill context constrains. Similar approach by Mostéfaoui [15] proposes that security rules should consist of four elements - permission, role, context and authentication method.

Lima [17] adds another context dimension to current security rules. It would make security policy three dimensional with context, permission and role. Difference from xoRBAC [14] is that it takes context more abstractly and complexly. Corrad [16] suggest leave the roles completely and assign permissions to contexts. Both those approaches are interesting in that they somehow compare contexts and make decisions on how similar they are.

Remarkable idea is proposed by Hung [18]. He proposes three entities in security rules - object, user and activity. All of the entities have some credentials. If user want to perform action on object he needs to poses credentials required for both object and activity.

Another interesting idea is described by Wendong [19]. He suggest adding user security level in addition to RBAC and define needed security levels to perform actions.

4. Proposed Solution

Security policies in organizations are very consistent and are changing just slightly over the time. Most of the organizations does not want and even does not need any radical change. Therefore CA security must be another logical

step to evolve current security. This will allow us to build new security rules on existing and well proved solution and it also makes the solution more accessible for people who are familiar with current solutions.

I propose create security level, which is based on context in addition to traditional roles in RBAC. Level can be understood as quantification how is the user trustworthy and it is dynamically tied to user. The security level creates second security constrain beside traditional permission and therefore resources in application now can have two different kind of security rules - classic permission tied with role and security level.

As the context of the user and the application is changing the level needs to reflect that dynamic nature of context. There are several moments when the level can be calculated. First is to calculate level during user's account creation. However this does not reflect dynamic nature of context and therefore is unsuitable. Opposite extreme is to determine level on every security check. This would reflect changing context most reliably but it is too demanding for computational resources and time consuming as the context check might not be trivial. As the best compromise seems to determine level during user's log in into application. It decreases number of context check by several orders and in the same time it provides very accurate snapshot of the user's context. In case the context would be changing rapidly the user can perform relogin or even the application can force new level calculation manually.

Resolving level is done by context resolvers. Each resolver takes responsibility for checking one particular part of context. For example one resolver would determine network, which user came from. Another would check time of the day and so on. Every resolver would return, which level it grants to user. As the security resolver is written withing application it has access to users information (e.g. his request, information about him stored in database, etc) as well as it can use information about the application (e.g. number of requests, number of users) and even about the machine the application is running on (e.g. load of the machine, location of the server). The level does not need to be set in resolver and it does not decide just if to give it or not, the resolver itself makes decision, which level to grant. After every resolver performs its inner logic and determines level on its own the highest level is used as the final user's security level.

Level representation by itself is very abstract. Only need for level is to be comparable with other levels to know if given level is higher or lower then required one and also to determine the highest one. Therefore it is not important whether level is represented by number, string or even some more complex structure. This leaves a lot of space for customization for a given application.

The proposed solution has many advantages. The most important ones are:

- Lightweight - it does not require any complex structures in application nor it does not consume significant system resources.
- Easy to use - it just requires adding another type of constrain to resources that need to poses CA security.
- Voluntary - if someone wants to use plain RBAC he can and just to chosen resources he might add level restrictions.
- Scalable - there is not any predefined set of levels nor there is no limit in amount of levels in application.
- Universal - the solution can be modified and used with other security architectures, not just with RBAC.

However the solution poses few limitations, which needs to be worked further on. Among them the most significant are:

- Hard to determine exact context - sometimes can happen that some resource should be accessible just from given context. For example some resources are accessible only during the day and some just during the night. Such scenario is impossible to secure with proposed solution.
- Levels are linear - structure of the levels is strictly linear and therefore it is impossible to build some tree or even more complex structure of levels. Often happen that there are multiple context rules, which are granted different set of right. Levels can't model for example geographical situation when users from same state have some rights but people in different location of the state got additional specialized rights.

5. Conclusion

In this paper I focused on the area of the CA security with focus on RBAC architecture. As is covered by related works the main issue of CA security is no standardized nor effective best practice. There are multiple approaches how to add CA elements into RBAC. However they suffer from multiple inconveniences. They are either too complicated and therefore they significantly decreases one of main advantage of RBAC which lays in its simplicity to maintain and develop. In some cases they also demands a lot of computational resources or even change the RBAC so extensively that it can be hardly called RBAC anymore.

I introduced a innovative solution based on adding another security constrain beside classic permissions tied to roles. The constrain is called security level and it is based on a context. Basically level describes how much user can be trusted. To access resource in application user is required not only to posses permission through roles but also to have

corresponding security level. This approach keeps advantages of RBAC and extends them further with CA elements. Even the solution has some flaws its advantages makes up for it.

In future I want to focus on transfer of security levels to other security architectures because the solution seems to be very flexible and I want to utilize advantaged of the described solution as much as possible. Apart from that I will explore options how to overcome linearity of the levels and how to model more complex context security constrains and generally reduce downsides of the proposed work.

Acknowledgements

Research described in the paper was supported by the Grant Agency of the Czech Technical University in Prague, under grant No. SGS14/198/OHK3/3T/13.

References

[1] ABOWD, Gregory D; DEY, Anind K; BROWN, Peter J; DAVIES, Nigel; SMITH, Mark; STEGGLES Pete. Towards a better understanding of context and context-awareness. In: Handheld and ubiquitous computing. 1999. 304307.

[2] MACIK Miroslav; CERNY Tomas; SLAVIK, Pavel. Context-sensitive, cross-platform user interface generation. Journal on Multimodal User Interfaces 8, no. 2 (2014): 217-229.

[3] HARTER, Andy; HOPPER Andy; STEGGLES, Pete; WARD, Andy; WEBSTER, Paul. The anatomy of a context-aware application. Wireless Networks 8, no. 2/3 (2002): 187-197.

[4] HONG, Jongyi; SUH, Eui-Ho; KIM, Junyoung; KIM, SuYeon. Context-aware system for proactive personalized service based on context history. Expert Systems with Applications 36, no. 4 (2009): 7448-7457.

[5] BUSH, Vannevar and As We May Think. "The atlantic monthly." As we may think 176.1 (1945): 101-108.

[6] SANDHU, Ravi. Access control: The neglected frontier. In: Information Security and Privacy. 1996. 219227.

[7] HITCHENS, M.; VARADHARAJAN, V. Design and specification of role based access control policies. Software, IEE Proceedings -. 2000, 147 (4), 117-129. DOI 10.1049/ip-sen:20000792.

[8] SCHILIT, Bill; NORMAN Adams; WANT, Roy. "Context-aware computing applications." In Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on, pp. 85-90. IEEE, 1994.

[9] MATTHEW, Moyer; ABAMAD, Mustaque. Generalized role-based access control. In: Distributed Computing Systems, 2001. 21st International Conference on., pp. 391-398. IEEE, 2001.

[10] COVINGTON, Michael J.; LONG, Wende; SRINIVASAN, Srividhya; DEV, Anind K.; AHAMAD, Mustaque; ABOWD, Gregory D. Securing context-aware applications using environment roles. In: Proceedings of the sixth ACM symposium on Access control models and technologies. 2001. 1020.

[11] PARK, Seon-Ho; HAN, Young-Ju; CHUNG, Tai-Myoung. Context-role based access control for context-aware application. In High Performance Computing and Communications, pp. 572-580. Springer Berlin Heidelberg, 2006.

[12] SLADIĆ, Goran; MILOSAVLJEVIĆ, Branko; KONJOVICĆ, Zora. Context-sensitive access control model for business processes. Computer Science and Information Systems/ComSIS. 2013, 10 (3), 939972.

[13] KULKARNI, Devdatta; TRIPATHI, Anand. Context-aware role-based access control in pervasive computing systems. In: Proceedings of the 13th ACM symposium on Access control models and technologies. 2008. 113122.

[14] NEUMANN, Gustaf; STREMBECK, Mark. An approach to engineer and enforce context constraints in an RBAC environment. In: Proceedings of the eighth ACM symposium on Access control models and technologies. 2003. 6579.

[15] MOSTÉFAOUI, Ghita Kouadri; BRÉZILLON, Patrick. A generic framework for contextbased distributed authorizations. In Modeling and Using Context, pp. 204-217. Springer Berlin Heidelberg, 2003.

[16] CORRAD, Antonio; MONTANARI, Rebecca; TIBALDI, Daniela. Context-based access control management in ubiquitous environments. In: Network Computing and Applications, 2004.(NCA 2004). Proceedings. Third IEEE International Symposium on. 2004. 253260.

[17] LIMA, Joao Carlos D.; ROCHA, Cristiano C.; AUGUSTIN, Iara; DANTAS, Mrio AR. A Context-Aware Recommendation System to Behavioral Based Authentication in Mobile and Pervasive Environments. In: Embedded and Ubiquitous Computing (EUC), 2011 IFIP 9th International Conference on. 2011. 312319.

[18] XUNG, Le Xuan; HASSAN, J.; RIAZ, AS.; RAAZI, SMK.; WEI-WEI, Y.; CANH, Ngo Trong; TRUC, Phan Tran Ho; LEE, Sungyoung; LEE, Heejo; SON, Yuseung; and others. Activity-based security scheme for ubiquitous environments. In: Performance, Computing and Communications Conference, 2008. IPCCC 2008. IEEE International. 2008. 475481.

[19] WENDONG, Zhang; KAIJI, Zhang. A role-based workflow access control model. In: Education Technology and Computer Science, 2009. ETCS09. First International Workshop on. 2009. 11361139.

About Authors...

Michal TRNKA received his Bachelor's and Master's degree from Faculty of Electrical Engineering of Czech Technical University in Prague, where he currently studies Ph.D. program. His area of interest is software engineering and especially application security.