Global Journal of Computer Science and Technology
Network, Web & Security
Volume 12 Issue 13 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Analysis of Decentralized & Diverse Access Control (DDAC) Architecture

By Rajender Nath & Gulshan Ahuja
Kurukshetra University, Kurukshetra, India

Abstract - Access control refers to securing access to the resources and allowing access up to some defined level. This paper presents various approaches implementing access control in an open domain and carries an analysis of decentralized and diverse access control (DDAC) architecture. The DDAC architecture eliminates the role of centralized authority for managing and issuing users' credentials. It allows the users to keep the right of disclosure of their attributes under the sole control of them and also ensures that the users are not able to modify the confidential credentials which have been registered and verified by various trusted attribute providers. This paper explains the metrics for carrying the analysis and then presents a theoretical and experimental analysis of the DDAC architecture.

Keywords : *Access Control, DDAC, Attributes, Credentials.*

GJCST-E Classification: *D.4.6*

Strictly as per the compliance and regulations of:

Analysis of Decentralized & Diverse Access Control (DDAC) Architecture

Rajender Nath [α] & Gulshan Ahuja [σ]

Abstract - Access control refers to securing access to the resources and allowing access up to some defined level. This paper presents various approaches implementing access control in an open domain and carries an analysis of decentralized and diverse access control (DDAC) architecture. The DDAC architecture eliminates the role of centralized authority for managing and issuing users' credentials. It allows the users to keep the right of disclosure of their attributes under the sole control of them and also ensures that the users are not able to modify the confidential credentials which have been registered and verified by various trusted attribute providers. This paper explains the metrics for carrying the analysis and then presents a theoretical and experimental analysis of the DDAC architecture.

Keywords : Access Control, DDAC, Attributes, Credentials.

I. Introduction

Open and distributed nature of Internet assists users to use online services for the benefits of costs, time and efficiency. To avail theses services users are required to submit their credentials for the purpose of registration and further verification. The credentials supplied by a user may not be sufficient enough to grant the access to the requested service and a further verification may need to be carried by demanding some confidential and secret credentials from the user.

However user may wish to disclose only basic set of credentials in the form of attributes and may decide to refrain from disclosing the confidential and sensitive attributes to service portals for the concerns of safety and privacy. This creates a requirement for trusted agencies, which can maintain private and confidential information of users and allow this information to be used by service providers without compromising privacy and security of user specific information.

A significant research has been carried in the field of federated identity management, which makes possible to utilize the existing Identity management systems for realizing authentication and authorization decisions. In a federated system, Identity Provider (IdP) plays an important role and issues the certified credentials, which can be utilized at the service provider's (SP) end. The scalability of such system is limited due to the need of IdP to act as a central authority and maintain credentials of ever growing large number of requesters.

As more and more portals are offering online services, there is a strong need to provide authentication and authorization independent of any central authority. A decentralized environment must allow various attribute authorities to collaborate dynamically to produce a set of attributes, which are consumed by the service providers for providing services to the requesting users. The rest of this paper is structured as follows. Section II highlights various decentralized access control mechanisms. Section III describes in brief about the DDAC architecture and its components. Section IV presents the analysis of the DDAC architecture and finally Section V presents the conclusion

II. Related Work

With the increase in number of service requesters and service providers, there was an increase in the complexity related with access management activities. The researchers started considering attribute management frameworks, which worked without involvement of any central authority to manage or process the users' attributes.

Cantor et al. [1], Chappell [2], Klingenstein [3], Jill et al. [4] approaches relied on IdPs and SPs for issue and consumption of attributes. The establishment of trust between IdPs and SPs required them to become part of the federated identity management. In federated system every IdP could define its own attribute release policy for each SP within the federation. The IdP had the full authority to decide about which attributes could be released to a particular SP based on the concerned access control policy. The service requester had no right to specify about attributes that could be released to an SP.

Regina N. Hebig et al. [5] proposed a decentralized identity and attribute based access control approach. The authors described a prototype implementation with an architecture based on the standards XACML, SAML, WSPolicy, WS-SecurityPolicy and WS-Trust, which put the focus on sharing identity and attribute information across independent domains for the purpose of access control.

Author α : Rajender Nath, Department of Computer Science & Applications, Kurukshetra University, Kurukshetra, India.
E-mail : rnath_2k3@rediffmail.com
Author σ : Gulshan Ahuja, Research Scholar, Kurukshetra University, Kurukshetra, India. E-mail : ahujag_24@yahoo.com

A recent framework Aditi [6] for user centric identity federation enhanced the standard federated model with new IdP and SP components operated directly by users. These components were termed as user IdP and user SP, respectively to provide an interface between the user and the federation. In Aditi system, the user could obtain all attributes from the IdP and store them locally. Aditi addressed issues like redirection of user requests, use of cookies, removal of need for introduction of an SP to the identity federation, scalability, providing complete user control over his attributes, trust management in order to help the SPs to find out the trustworthiness of an IdP. In this approach all the attributes of the user were still kept with IdP and the user had to download all attributes from IdP to the card selector in order to utilize these attributes for authorization decisions. This provided users with full control over their attributes, which could be changed at the will of the user. Therefore ADITI framework was not well suited for service portals where users' attributes were required to be verified without control of users over their own attributes and independent of any centralized authority.

The problem evolved in relation to management of attributes in multiple federations. With the continuous and fast pace increase in the number of service requesters, the numbers of federations also increased. Each SP had to manage its linkages across multiple federations. This increased the complexity related with access management across multiple federations. Moreover, the IdPs still played the role of central authority for issuing and managing the attributes of users. There was a need for attribute management framework, which worked without involvement of any central authority to manage or process the user attributes.

The DDAC architecture presented by Rajender Nath et al. [7] considered the use and verification of diverse attributes for supporting online services in a decentralized manner. It allowed utilizing diverse attributes without involvement of any centralized agency for management and issue of access related attributes. In the next section, we outline in brief about DDAC architecture and its components.

III. DDAC Architecture Components

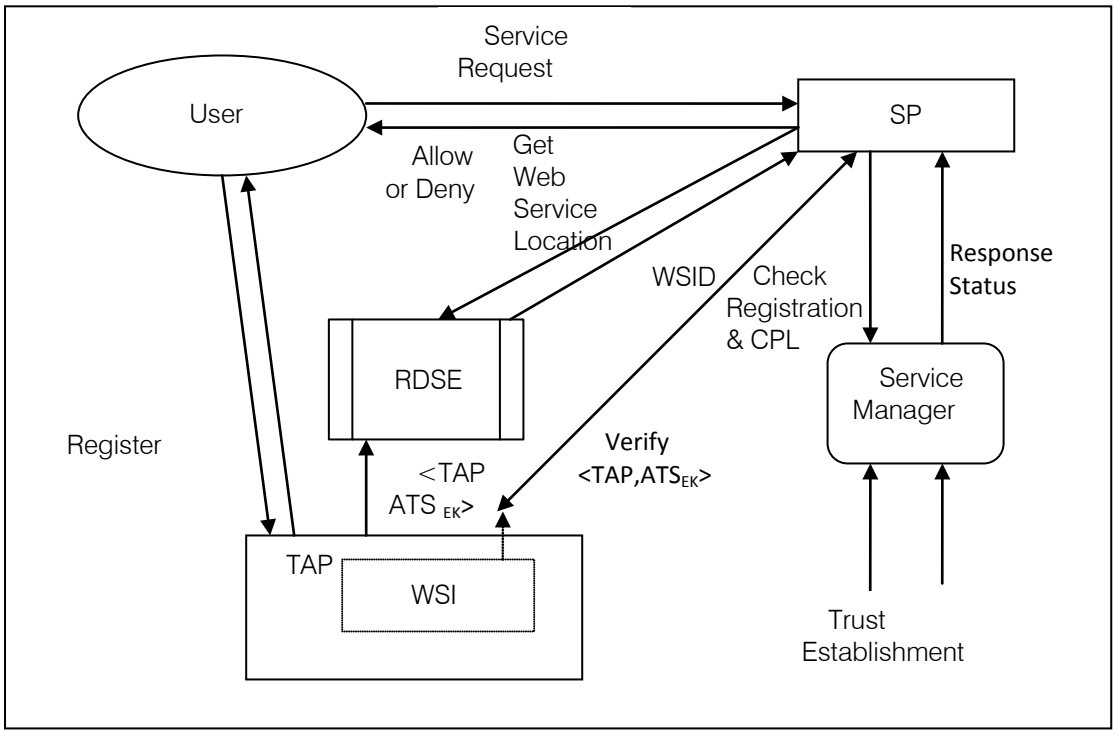

Fig.1.1 : Components of DDAC Architecture

The DDAC architecture allows safe integration of users, service providers (SPs) and attributes authorities (AAs) for disclosure & utilization of the attributes. The DDAC architecture permits SPs to verify about the user's attributes without involvement of any central authority. It eliminates the role of any central agency for issue and management of attributes and provides complete control over user's attributes. Figure 1.1 depicts the detailed view of the overall architecture.

The DDAC architecture mainly comprises of four components such as (a) user attribute management (UAM) module (b) resource description search engine (RDSE) (c) web service interface (d) service manager. The UAM is a requester side module, which implements

three basic operations - attribute creation, attribute-authority mapping, attribute storage. The RDSE provides both information collection and information retrieval functions for all SPs. It also provides the means for all trusted attribute providers (TAPs) to register their information, which is utilized by SPs to verify about users' attributes. Each TAP provides a web service, which is accessed by the SPs, for verification of users' attributes. The TAP provides the means for communication with the web service through a well defined service interface. The service manager is located on the SP's end and is composed of six components such as (a) Policy Store (b) Controller Module (c) Users' Registration List (d) Credibility Profile Level (CPL) Data Store (e) Credibility Verification Module (CVM) (f) CPL Computation Module.

The policy store keeps information about users' attributes along with the set of policies, which specify the rules and conditions under which access can be granted or denied. The Controller Module acts as the overall organizer for invoking and fetching response from the other components. Once a user sends a request for a service to an SP, the controller module intercepts the incoming request, invokes the credibility verification module (CVM) and directs it to process the service request. The CVM evaluates the registration time attributes against the registration list to verify whether user is already registered or not.

The Users' Registration List contains registration details about all those users who have already registered with an SP for accessing a service. The CPL data store contains the information about service access request related parameters. The CVM verifies the attributes against a data registry to check whether the requesting user is already registered or not. If the user is already registered, the CVM module invokes CPL computation module for calculating CPL value for the requesting user. Otherwise, the CVM module asks the user for registration and carries the verification through RDSE query. The CPL computation module computes the value for CPL based on service request related parameters

The next section carries the analysis of DDAC architecture and presents the performance results.

IV. Analysis of Ddac Architecture

To analyze the merits of the DDAC architecture three main parameters have been identified such as (a) Performance (b) Time Effectiveness (c) Cost Effectiveness. The analysis of the DDAC architecture based on the above mentioned parameters is presented below:

a) Performance

The DDAC architecture is implemented using Java Framework. The portal interface has been deigned using Java Server Pages. The experiment is conducted on a 2.4 GHz Intel Dual Core Pentium machine with 1 GB of RAM, Windows XP operating system. The attribute storage and retrieval services are provided by installing IBM Tivoli Directory Server for Windows on a remote site. A web service is implemented for receiving of verification request, query of attributes from Tivoli Server and generating response for the SP.

The working of the architecture is tested for two different cases

Case 1: For requests based on registration time attributes.

Case 2: For requests based on registration time attributes & another set of attributes stored with TAP.

The experimental details for first case are described as follows:-

The experiment is performed for 100 requests, where each access request contains only registration time attributes. For each access request, the types of registration time attributes and threshold values are varied. The CPL value is computed as per eq. 1, 2, 3 & 4 and is normalized in the range of <1, 10>.

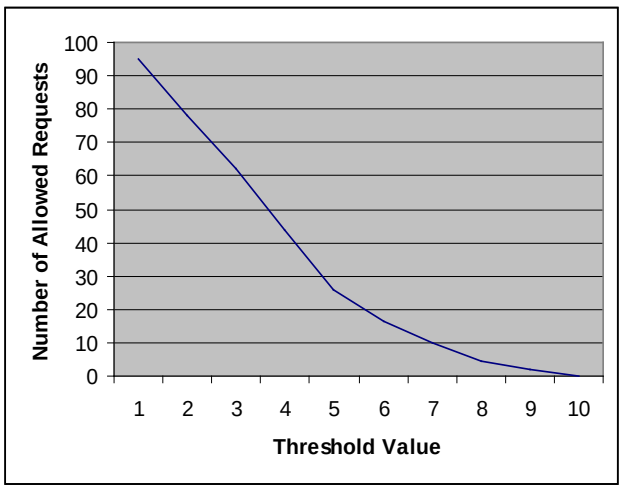

Figure 1.2 : Threshold Value vs. Number of Allowed Requests

The obtained results as per figure 1.2 highlight that with the increase in threshold value the number of allowed access requests also decrease. At mid of the total threshold range, there is found a sharp decline in the allowed number of requests. A further increase in the threshold value results in the rejection of most of the number of access requests as their computed CPL value comes out as below than the permissible limits.

The experimental details for second case are described as follows:

The experiment is performed for 100 requests, where each request contains registration time attributes and another set of attributes, which are maintained with TAP. For each access request the types of registration time attributes and TAP's attributes are varied. The experiment is conducted by varying the threshold values in the same intervals as in above presented case 1.

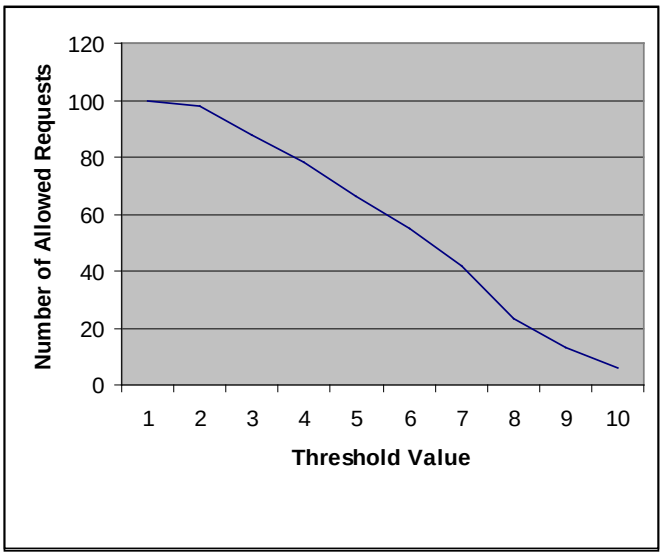

Figure 1.3 : Threshold Value vs. Number of Allowed Requests

As Per Figure 1.3, It Is Found That For The Same Threshold Levels; Most of The Access Requests Are Allowed. A Further Increase in the Threshold Value Results in a Steady Decline in the Number of Allowed Requests. At Maximum Level of Threshold Value, Some Requests Are Still Allowed. Thus it is found that the Ddac Architecture Performs Well as Compared to Existing Access Control Mechanisms.

b) Time Effectiveness

The use of DDAC architecture results in considerable saving in time required to deliver the required products to the requesters. The computation and use of CPL values for access request allows an SP to establish some degree of trust with the requesting client. The degree of trust further increases with the AR and PDR values associated with the same client.

The time effectiveness of the DDAC architecture is calculated as follows:

Assuming that there are N numbers of requests for purchase of products and out of total of N requests, for P requests the products are returned for valid reasons and for Q number of requests due to some defaults.

Total time T1 required to serve N requests, when no verification is carried, is computed based on time required to deliver the product (TD), time required to receive back the rejected product (TR) and time required to receive back the product in case of a default (TU).

$$T_1 = P*(T_D + T_R) + Q*(T_D + T_U)$$

Now, considering the case where the verification is carried based on CPL value, the time T2 required to serve N number of requests is computed as follows:

$$T_2 = P*(T_D + T_R)$$

The use of CPL value, leads to elimination of time caused by Q number of defaulting requests.

The time effectiveness value (TE), which describes the total saving in time, is computed as

$$TE = T_1 - T_2$$

The value of TE results in a significant amount of saving in time for the organization.

c) Cost Effectiveness

The DDAC architecture allows an SP to verify about the genuineness and validity of service requester. The services are provided only after ascertaining about the details about the requester.

The method employed in the architecture considers CPL as one important factor for serving users requests. The CPL is computed based on the service request related parameters such as the number of times requested items accepted by the user on delivery, the total number of items supplied, timely payment, number of times delay occurred during payments, the time delay in payment, the time allowed for payment etc. The values of these parameters for a user varies based on the past transaction details interactions with an SP. The DDAC architecture has been designed in a manner that it significantly reduces the request processing overhead based on the CPL value of a user. This results in a considerable saving in terms of costs of delivery.

V. Conclusion

This paper has presented a theoretical and practical analysis of the working of DDAC architecture. The DDAC architecture works well in a decentralized manner and provides means by which various attribute providers can dynamically collaborate to utilize users' attributes. The concept of CPL in DDAC architecture leads to reduction in the time required to verify service requests, based on the users' credibility values and previous experiences. The change in the value of one or more attributes can be easily carried by trusted attribute provider without any hassles of intimation to any other party. The trusted attribute providers only provide the location and signature of web service in resource descriptive search engine. The information about signature of web service in resource descriptive search engine remains unchanged and do not effect any operation even when there is a change in the value of one or more users' attributes.

References Références Referencias

1. S. Cantor, S. Carmody, M. Erdos, K. Hazelton, W. Hoehn, R. Morgan, T. Scavo and D.Wasley, "Shibboleth Architecture, Protocols and Profiles", Working Draft 02, "http://shibboleth.internet2.edu/", 2005.

2. D. Chappell, "Introducing Windows CardSpace", Microsoft MSDN website, http://msdn.microsoft.com/enus/library/aa480189.aspx, 2006.
3. N. Klingenstein, "Attribute Aggregation and Federated Identity", International Symposium on Applications and the Internet Workshops (SAINTW'07), pp. 26, 2007.
4. Jill Gemmill, John-Paul Robinson, Tom Scavo, Purushotham Bangalore, "Cross-domain authorization for federated virtual organizations using the myVocs collaboration environment", Concurrency and Computation: Practice and Experience, pp. 509-532, July 2008.
5. Regina N. Hebig et al., "A Web Service Architecture for Decentralized Identity and Attribute based Access Control", IEEE International Conference on Web Services, pp. 551 – 558, 2009.
6. Michal Prochazka, Daniel Kouril, Ludek Matyska, "User Centric Authentication for Web Applications", IEEE, pp. 67-74, 2010.
7. Rajender Nath, Gulshan Ahuja, "Decentralized & Diverse Access Control Architecture (DDAC) for Online Purchases", International Journal of Computer Applications (IJCA), Volume 30, No.1, p.p. 26-30, September 2011.

This page is intentionally left blank