A Generalized Model for Internet-Based Access Control Systems with Delegation Support

Utharn Buranasaksee, Kriengkrai Porkaew, and Umaporn Supasitthimethee

School of Information Technology
King Mongkut's University of Technology Thonburi, Bangkok
54500702@st.sit.kmutt.ac.th, {porkaew,umaporn}@sit.kmutt.ac.th

Abstract. In the web environment, web browsers use HTTP/HTTPS to communicate between users and web/application servers. However, many internet activities require interactions among three parties without compromising confidentiality. For example, an e-commerce transaction requires a buyer to authorize an e-commerce website to withdraw money from the buyer's bank account at an internet banking website. Although several existing works have been proposed to solve this problem, they are done in ad-hoc manners or lack of some important properties. This paper proposes a model, called PRA (Provider-Requestor-Authorizer), for generalizing three-party communication in the web-environment in order to identify desirable properties that can be used to measure the goodness of protocols for and classify them. We found that PRA model can generalize three-party communication protocols to a single model from conceptual level to implementation level.

Keywords: design, implementation, distributed access control, distributed system, classification, delegation.

1 Introduction

Since the invention of the Web, web-based applications have become ubiquitous because of its numerous advantages over traditional applications. One of the main reasons is that the users do not need any program other than web browsers which are commonplace. Unlike installing and using programs from unknown sources, with web browsers and HTTP/HTTPS, users do not need to worry about security. However, HTTP/HTTPS supports only communication between two parties. In some internet applications, three-party communications are necessary, such as e-commerce applications where there are a buyer, an e-commerce website and an internet banking website. These three parties need to communicate to one another in order to perform a buying transaction by delegating a right from one party to another without compromising confidentiality. For instance, the e-commerce website does not need to know about the bank account number of the buyer and the bank does not need to about what the buyer buys. Three-party communication can be built on top of two-party communication by having 3 pairs of two-party communication. Though there are many existing protocols and studies that support three-party communication [1-8], they are usually

K. Singh, A.K. Awasthi, and R. Mishra (Eds.): QSHINE 2013, LNICST 115, pp. 975–988, 2013.

done in ad-hoc manners. Some protocols [9] even have security holes. Unlike the existing studies that focus on specific problems, this paper proposes a model that generalizes three-party communication called PRA model (Provider-Requestor-Authorizer) and identify desirable properties for three-party communication in the web environment so that it can be used to measure the goodness of such protocols and classify them. In addition, this paper also proposed implementation roles in PRA model which is an implementation level view of three-party communication in order to identify and generalize the process and required features of the protocol.

The remainder of this paper is organized as follow. In Section 2, we present the related work about the existing protocols. Then, the details of PRA model is discussed in Section 3. In Section 4, implementation role in PRA model are identified. We also analyzed the existing Internet-based access control protocol with delegation by PRA model and implementation roles in Section 5. Finally, the conclusions and future work are drawn in Section 6.

2 Related Work

There are many types of existing protocols that emulate three-party communication on the web. One type of these protocols is that a user is authorized by one system to access another system such as Single Sign On (SSO). SSO allows a user to use only one account to access multiple websites. In this type, OpenID [1] is one of the most popular SSO protocols available on the web. It is an open standard allowing anyone to be OpenID provider. Another protocol in SSO is Shibboleth [2]. Unlike OpenID, Shibboleth gained much attention in institution group since the protocol assertion has home institution feature. However, in enterprise, SAML [3] protocol was proposed to handle SSO and cross system authorization in XML format. SAML protocol was designed to be extensible to support many solutions, but the protocol does not gain much attention on the Internet due to its complexity.

Another type of three-party protocols is that a user is authorizing one system to perform a task at another system on a user behalf. Other than SSO, Gonzalez et al. [4] proposed an OAuth based protocol called Reverse OAuth. In Reverse OAuth, a user is authorized to access the resource on third party website while OAuth protocol [5, 6] allows a user to authorize one system to perform the tasks on another system without sharing user's credential. Due to its simplicity, OAuth is one of the most popular cross authorization protocols on the Internet. To support more in distributed system, Schiffman et al. [7] proposed DAuth protocol by introducing an agent to handle master token and sub-token to each sub-component in distributed system. In some situation, when there are complex delegation rules to delegate the user from many systems, xDAuth protocol [8] introduced a central agent called Delegation Service to handles predefined rules of delegation so as to reduce redundancy of delegation rules.

Though many of existing protocols are well-known and on the market, they were designed to focus on specific problems such as single sign on, cross system authorization, etc. None of the existing protocols were proposed to support the extension to

solve general problem which could be different approach of authorization. For example, OAuth solves the problem of a user authorizing one system to access the resource on another system while OpenID solves the problem of a system authorizing a user to access another system. This study focuses on a generalized model called PRA model as well as its implementation guide that can be used to design generalized three-party communications protocol.

3 PRA Model

PRA model is a generalized model of three-party communication that proposes basic communication and required steps in order to make the interactions among the nodes complete without compromising confidentiality. In the Internet communication, when a node located in one domain requests the resources located on another domain, it would require third node that has accessible rights to grant the access. Therefore, we build Provider-Requestor-Authorizer model (PRA model) which based on the roles in the Internet communication.

According to PRA model, each scenario will have at least three roles involved in access control system with delegation support: Service Provider, Service Requestor and Service Authorizer. Each node performs the workflow according to its role. First, the Service Provider refers to the node that stores protected resource, or exposes some services. As we focus on Internet based access control model, Server Provider normally refers to a computer operator node since it electronically stores some resources on the Internet. Second, the Service Requestor refers to the node that receives the permission and accesses the resources or the services. Third, the Service Authorizer refers to the node that could physically or virtually own the protected resources located at the Service Provider node. According to our assumption, each node can either be human operator or computer operator. Therefore, there are 2^3 (2^N) = 8 scenarios that could occur in PRA model. Therefore, we identify the scenario according to Service Provider, Service Requestor, and Service Authorizer roles as the following:

3.1 CHH Scenario (Computer-Human-Human)

In CHH scenario, both Service Requestor and Service Authorizer are human operators while only Service Provider is computer operator. In this scenario, one user grants an access to another user on the same system. The grant typically is done in advance before an access, with no restriction and time limitation. Figure 1 shows that a user acting as Service Authorizer identifies himself, selects the sharing resource with privilege to another user. Then, another user acting as Service Requestor logs in, and requests for the shared resource afterwards. For example, a user shares a document to his colleague. After that, the user's colleague accesses the shared documents. The sharing usually allows user's colleague to edit the documents until the grant is revoked. The grant could also be adjusted according to privilege list depending on the resource type such as read only and read/write. There are many studies proposed the

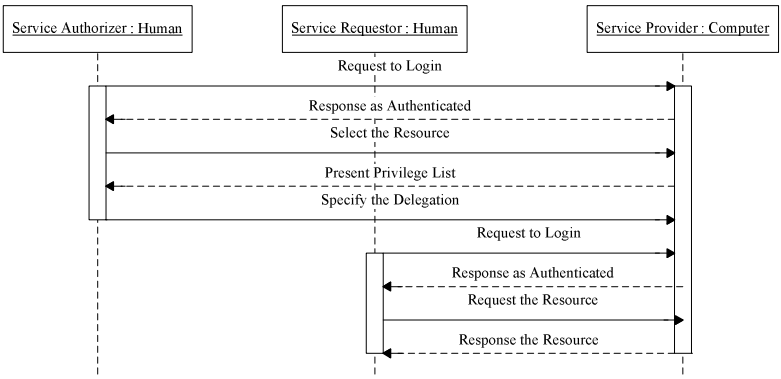

Fig. 1. CHH Scenario Basic Process

work on this scenario. For example, delegation in role based access control [10] and delegation in context-aware access control [11]. Furthermore, there are also many services in this scenario available on the Internet such as sharing documents on Google Docs [12], Facebook [13], etc. However, these previous studies were designed in ad-hoc manner. This scenario requires only Service Provider to have the interface for the user to share the resource and the storage to keep track the shares since the access control in this scenario does not have to pass secret information across the system.

3.2 CHC Scenario (Computer-Human-Computer)

In CHC scenario, both Service Provider and Service Authorizer are computer operators while Service Requestor is human operator. In this scenario, a user with the help of the server in one domain could get an access to protected resource in the server located on another domain. The process, as shown in Figure 2, starts by a user as Service Requestor requesting a resource on Service Provider. After that, a user is redirected to authenticate and to get authorized from Service Authorizer. Getting authorized, a user carries access token via URL redirection to Service Provider. Then, Service Provider verifies the authenticity of access token. If it is authentic, Service Provider returns a resource to a user. One of the situations that match this scenario is Single Sign On (SSO). SSO allows a user to use single credential as identification to access the resources on another systems. In enterprise, the applications could be run on many servers and the company maintain single identity provider to handle authentication process. When the user wants to access the application, he is redirected to identity provider to perform authentication. Furthermore, the logout status of the user needs to be propagated to recently used applications when the user logout. This scenario requires both Service Provider and Service Authorizer to have the same

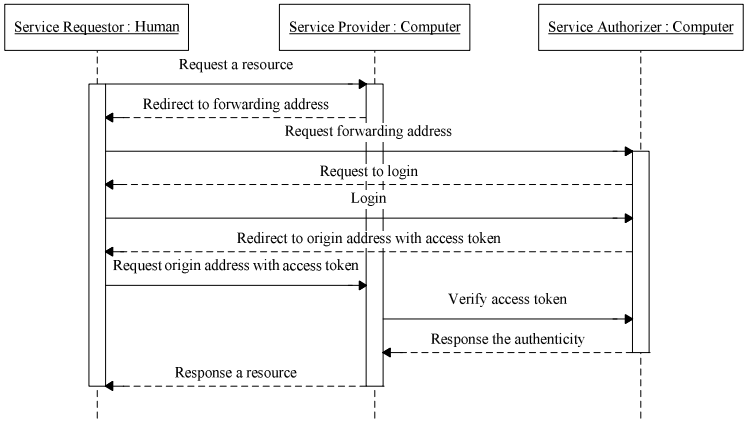

Fig. 2. CHC Scenario Basic Process

interface in order to communication to each other as well as Service Provider to trust Service Authorizer information. There are also many protocols available on the market such as OpenID [1], Microsoft Account [14], SAML [3], Shibboleth [2], etc.

3.3 CCH Scenario (Computer-Computer-Human)

In CCH scenario, both Service Provider and Service Requestor are computer operators while Service Authorizer is human operator. In this scenario, Service Requestor and Service Provider are likely to be located in different domains while the user could have an action in both parties so that the user has one computer perform on another computer on the user behalf. The process, as shown in Figure 3, starts by a user requesting Service Requestor to perform the task on Service Provider. Then, a user that carries requested task information from Service Requestor is redirected to Service Provider. Accessing Service Provider, a user is presented the login form. After logging in, a user is asked to authorize the task that Service Provider processes from the redirecting parameters. At this stage, a user could see requested operations and resources required by Service Requestor. After the operation is authorized, Service Provider returns access token and redirects a user to Service Requestor. Finally, Service Requestor uses that access token to access the resource on Service Provider, and renders the output to a user.

Since the authorizer in this scenario is human, the delegation is usually done in a real time rather than in advance. For example, asking to authenticate and authorize at Gmail website, Facebook fetches the user's contacts from Gmail and adds to Facebook friends immediately. From the nature of workflow in this scenario, the grant is given by once rather than giving always allowed. Therefore, the restriction of

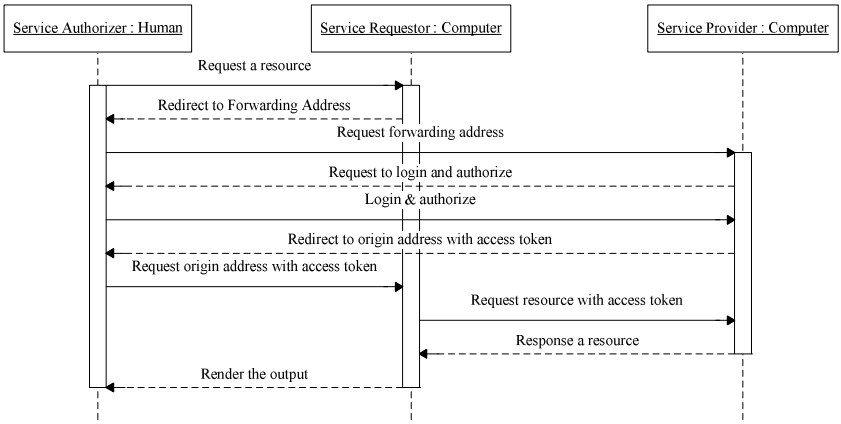

Fig. 3. CCH Scenario Basic Process

the delegation such as time limit should be a necessary requirement. Another issue in this scenario generally involves confidentiality of the user. Unless a proper delegation transfer is deployed, Service Requestor could gain over the access since it could misuse the user's credential in Service Provider.

Therefore, this scenario requires Service Provider to have appropriate method that could differentiate computer operator from human direct access from browser in addition to on-the-fly authorization interface for Service Authorizer. Furthermore, Service Requestor and Service Provider should know each other which parameters to pass to the other via the user browser without compromising integrity and confidentiality. There are also many protocols proposed in this scenario such as OAuth [5, 6], Reverse OAuth [4], DAuth [7], xDAuth [8], etc.

3.4 CCC Scenario (Computer-Computer-Computer)

Unlike another scenario, CCC scenario has all the nodes including Service Provider, Service Requestor, and Service Authorizer performing in computer operators. Therefore, this scenario has no human interaction. Comparing to another scenarios, the basic process in CCC scenario could be applied from both CHC and CCH scenarios by changing human to computer operator instead. Therefore, CCC scenario could have two types of the basic process depending on what role the first party that initiates the communication takes place. If the first party takes Service Requestor role, the basic process matches CHC scenario. If the first party takes Service Authorizer role, the basic process matches CCH scenario. Even though CCC scenario could also be applied from CHH scenario, we do not consider this as three-party communication. If Service Authorizer grants the privilege beforehand, there would be only Service Requestor interacting with Service Provider. Unfortunately, at the time of writing this paper, we have not found any standard three-party communication protocol proposed

since the communication is generally done in two-party since the application of three-party communication still is not clear.

Regarding another four categories in PRA model in which Service Provider is human, we are unaware of those scenarios and they are out of scope in this paper since we restrict that Service Provider roles would electronically store the protected resource online.

Though every parties in these scenarios perform the same concept of communication, the interactions of each party in these scenario does quite different when compare to that of another. For example, the interactions of Service Provider in CHC scenario is quite different when compare to that in CCH scenario. However, the protocols in each scenario also share some common implementation. Therefore, to generalize the process, we proposed implementation roles that could be used to generalize the function of three parties in PRA model.

4 Implementation Roles in PRA Model

By the nature of the protocols in CHH scenario, they generally involves the grant that is predefined in advance rather than immediate action and it does not involve cross system communication. For example, a user shares the document to another user. This kind of authorization is different from that in the other scenarios where the authorization is done on-the-fly and real time manner. Therefore, the generalization of the protocols in CHH scenarios is out of scope.

Implementation role refers to the role in PRA model that considers the implementation function such as redirecting a user, passing parameters, generating token, etc. rather than conceptual level such as authorizing node, requesting node, providing resource node. To illustrate this, comparing Figure 2 and Figure 3, the basic process in these two scenarios perform mostly the same pattern except that the ones that perform the same task are in different role. For instance, Service Provider in CHC scenario conceptually performs the same request/response as Service Requestor in CCH scenario does, and Service Authorizer in CHC scenario conceptually performs the same request/response as Service Provider in CCH scenario does. Therefore, we could see that, regardless which role the node takes in communication, each node has some specific interactions as well as information exchanging with another node as shown in Figure 4. To generalize the implementation requirements in the three scenarios, we proposed the implementation roles – Ticket Holder, Ticket Checker, and Ticket Master.

4.1 Ticket Holder

Ticket Holder refers to a user who first initiates the request to the resources or services. In this implementation role, a user would have nothing to implement on the server but hold the ticket via URL redirection between Ticket Checker and Ticket Master.

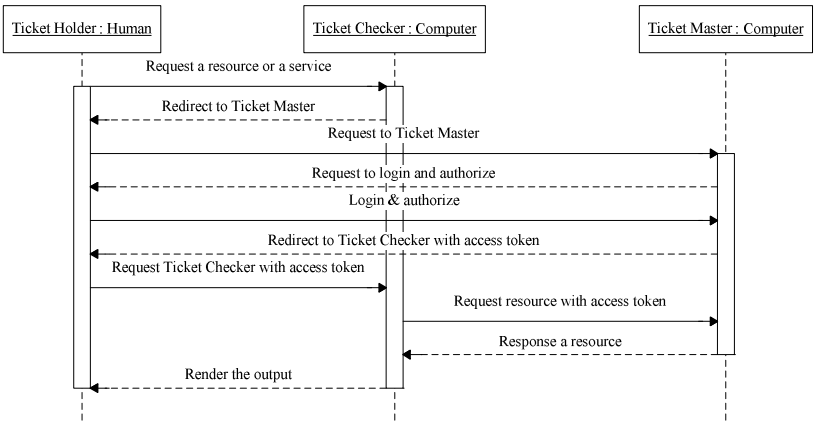

Fig. 4. Implementation Role in PRA Model

4.2 Ticket Checker

Ticket Checker refers to the node that obtains information from another node by access token so that it could use this information to perform some tasks or provide some services. Unlike Service Requestor role that conceptually refers to the node that directly accesses the final output, Ticket Checker in implementation role concerns the node that forwards the user to obtain access token, and uses access token to access some information on another server, and response a finishing resource or service to a user. In addition, Ticket Checker is responsible to verify the authenticity and integrity of access token and information. Therefore, when implementing Ticket Checker, the Ticket Checker has to provide a user interface to authenticate and request for a resource or a service. Furthermore, it should also be capable to request, verify, and use the token to integrate the service from Ticket Master.

4.3 Ticket Master

Ticket Master refers to a node that produces information to another node by generating the tokens so that it could allow another node to obtain information and process the output to the user. Unlike Service Provider role that conceptually refers to the node that electronically stores protected resources and that protect resources would be accessed by Service Requestor, Ticket Master in implementation role refers to the node that generates token to another. After that, whenever Ticket Checker uses generated token to obtain the resource, Ticket Master returns the resource or verify the authenticity of an access. Therefore, when implementing Ticket Master, it should be capable to generate, verify the token as well as maintaining generated tokens. In addition to providing user interface for Ticket Holder to authenticate, Ticket Master should provide different approach that is suitable format for Ticket Checker to fetch

the information when accessing with the generated token rather than returning a HTML document.

It is not only that both parties are capable to perform the required function, but also they should trust to each other. The trust might not be fully-trusted, but at least public-private key or shared secret key. For example, both parties could verify that the value in passing parameters have confidentiality, integrity and authenticity since Client could capture and change the value while redirecting. This could be done by deploying digital signature or encryption.

5 Example Analysis of Existing Protocols

We give the brief characterization of existing access control protocols, and explain the process of the protocol. Then, we apply PRA model into the protocol in order to categorize the protocol into suitable scenario. After that, we compare each protocol process to basic process in PRA model, and discuss how the protocol could be generalized to PRA model in implementation level using implementation roles. Due to space limitation, we pick some of the protocols in each type and analyze in this paper.

5.1 OpenID

OpenID is a decentralized open authentication protocol that allows users to identify themselves on any OpenID-enabled website. In OpenID context, there are three nodes which are OpenID provider, relying party, and user. The process of OpenID protocol,

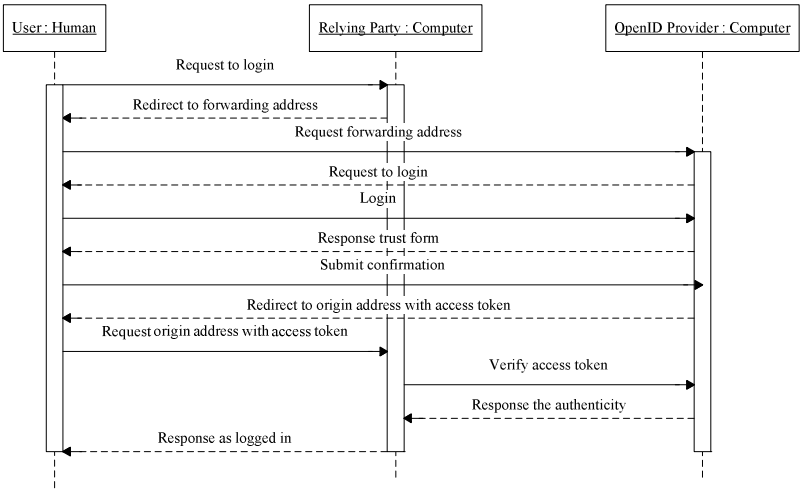

Fig. 5. OpenID Protocol Process

as shown in Fig. 5, starts by a user wanting to access the resource on relying party by providing the OpenID account information. Then, relying party redirects a user to OpenID provider page. Then, a user gets authenticated and confirms the trust to relying party at OpenID provider. After that, relying party verifies if the account is not blacklisted, the user is allowed to access the resource on relying party based on OpenID account. Therefore, in OpenID protocol, OpenID acts as Service Authorizer that authorizes end user as Service Requestor to protected resources located at relying party which acts as Service Provider. Thus, OpenID matches CHC scenario.

Comparing to CHC scenario basic process, OpenID protocol does quite the same as CHC basic process does except that relying party does not only request for authenticity from OpenID provider, but also the user's account information. Furthermore, when compare to implementation roles, relying party and OpenID provider could take Ticket Checker and Ticket Master role accordingly while a user takes Ticket Holder role.

5.2 OAuth 2.0

OAuth 2.0 is one of the most popular web authorization protocols. OAuth protocol was proposed to allow delegation the resources without giving credential. In OAuth context, there are three nodes which are Service Provider, OAuth consumer, and the user. The process of OAuth protocol, as shown in Fig. 6, starts by OAuth consumer requesting the resource at service provider in exchange of request token and redirecting user to authorize. After the user authorizes the request token at service provider, the service provider returns access token to OAuth consumer so that OAuth consumer could use this access token to access protected resource afterwards.

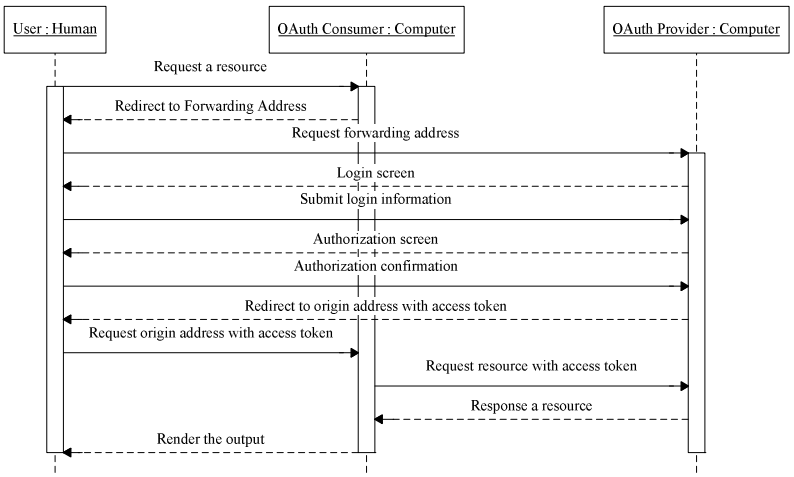

Fig. 6. OAuth Protocol Process

According to PRA model, the user acts as Service Authorizer that authorizes OAuth consumer as Service Requestor to access the resource at Service Provider. Therefore, OAuth protocol matches CCH scenario. However, when redirecting a user to OAuth provider, OAuth consumer also has to specify required permission, called scope, in addition to forwarding address and request token.

When applying implementation roles to OAuth, OAuth consumer could be implemented in Ticket Checker role and OAuth provider could be implemented in Ticket Master role. Furthermore, OAuth protocol process does the same as CCH scenario basic process does since there are only three parties.

5.3 Reverse OAuth

Reverse OAuth is a delegation access control protocol that based on OAuth protocol to support different requirement. Unlike OAuth which allows a user to grant one system to access another system, Reverse OAuth allows a system to grant a user to access another system on-the-fly using token. In Reverse OAuth, there are three nodes which are student, web tool, and LMS system. The process of Reverse OAuth, as shown in Fig. 7, starts by a student requesting a service at web tool via LMS system. Then, a student authenticates himself at web tool using request token generated from LMS system. Then, web tool asks LMS system if a student who uses this request token have privilege to access web tool. If so, a student could have an access to web tool. According to PRA model, LMS system acts as Service Authorizer that authorizes student as Service Requestor to access web tool as Service Provider. Therefore, Reverse OAuth matches CHC scenario in PRA model.

According to implementation role, a student as a human takes Ticket Holder role. LMS system does the job generating an access token to a student. Therefore, LMS system takes Ticket Master role while web tool that uses access token to render a finishing output to a student takes Ticket Checker role. Comparing to CHC scenario basic process, Reverse OAuth protocol has more complicate process several points. First, a student could not just directly access to the web tool, he needs to request to LMS system before requesting to web tool. This is because Gonzalez et al. designed the protocol to support controlling usage of the web tool such as time period, scope of usage. However, with CHC basic process, Ticket Master could still validate the right of the user before generating an access token to Ticket Holder anyway. Second, the authenticity and the authorization of access token are checked by Ticket Checker before passing to Ticket Holder. However, in CHC basic process, the authenticity and authorization of access token whenever Ticket Holder requests a service. This is just a design differentiation since PRA model could be adjusted to flexible enough.

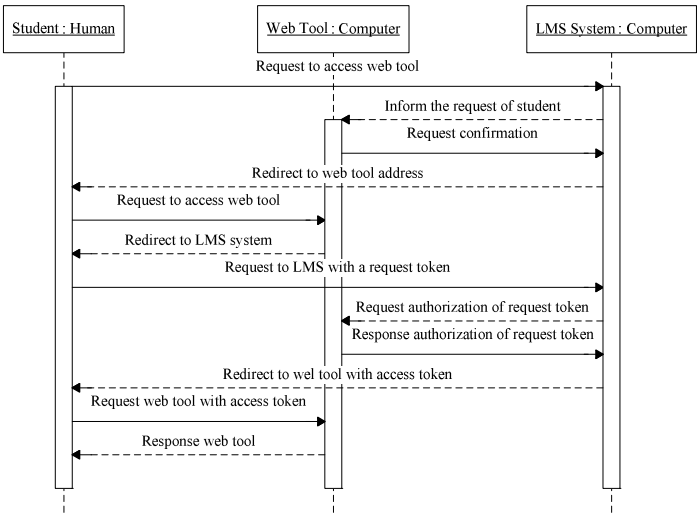

Fig. 7. Reverse OAuth Protocol Process

5.4 DAuth

DAuth is an authorization delegation access control for distributed web application. The protocol assumes that one web application would have many sub-components and addresses the problem of fine-grained authorization on each sub-component by introducing a new component called DAuth Agent. Therefore, there are four types of node in DAuth protocol which are Server Provider, DAuth Agent, consumer components, and user. The process starts by, as shown in Fig. 8, a user requesting a resource to DAuth Agent. Then, DAuth agent requests for a request token from Service Provider and redirect a user to authenticate and authorize at Service Provider. After authorizing the request token, Service Provider generates an access token and passes it to DAuth Agent via a user. Obtaining a master access token, DAuth Agent acts as a proxy of consumer components by requesting access sub-token and transmits to consumer component. Whenever consumer component needs to access the resource at Service Provider, it just requests access sub-token from DAuth Agent.

According to PRA mode, the user acts as Service Authorizer that authorizes both DAuth Agent and consumer components to access the resource at Service Provider. Therefore, DAuth protocol matches CCH scenario. Since DAuth has 4 parties involved in the protocol, the process in the protocol is more complicated than that of CCH basic process. It could be called as additional features to basic protocol which allows DAuth Agent to handle distributed components.

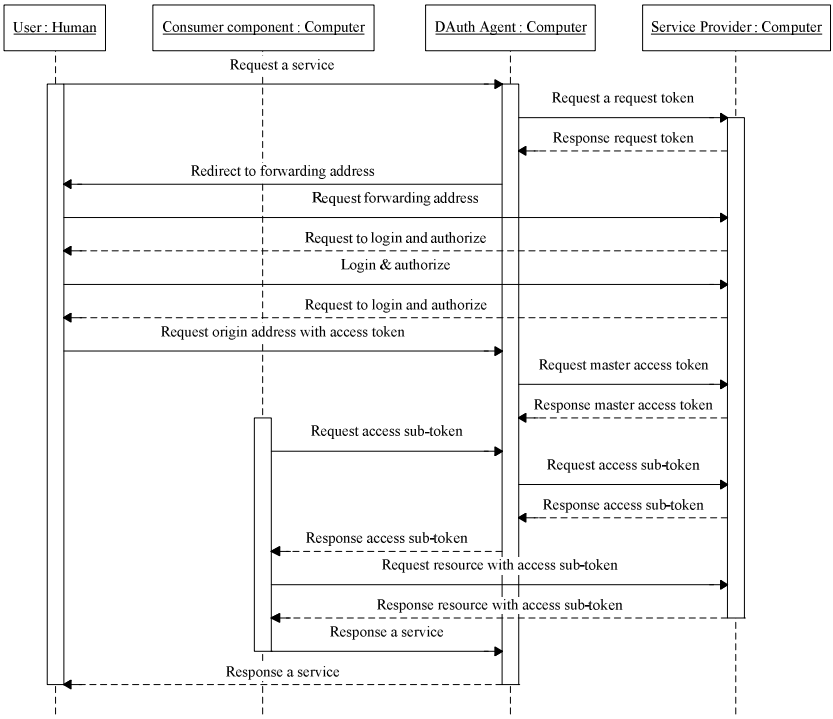

Fig. 8. DAuth Protocol Process

According to implementation roles, both consumer components and DAuth Agent that are located in the same domain operate by obtaining the token from Service Provider and using it to access the resource. Therefore, both of them act as Ticket Checker. Second, Service Provider that generates request token and access token to another party would take Ticket Master role. Finally, a user takes Ticket Holder role since the generated token is passed via URL redirection. This could be seen as CCH scenario basic protocol with sub token extension.

6 Conclusion and Future Work

We have proposed a generalized model for Internet-based access control systems, called PRA model, which built for three-party communication. The model categorizes access control into scenario. The access control protocols that fall into the same category have the same approach of handling authorization. After proposing PRA model, we found that CHC, CCH, and CCC scenarios could be generalized the process by introducing implementation role. According to implementation role in PRA model, the node that takes particular role would have specific guideline of implementation regardless which scenario the protocol is. After reviewing the existing

protocols by listing the process of the protocol and applying both the concept and implementation roles of PRA model into each protocol, we found the common implementation of each protocol that could be categorized to the same category in PRA model. Furthermore, the existing protocols that fall into CHC, CCH, and CCC protocols could also be generalized to PRA model in terms of both concept and implementation.

Acknowledgement. This work was supported by the Higher Education Research Promotion and National Research University Project of Thailand, Office of the Higher Education Commission and IBM Faculty Award.

References

1. OpenID Authentication 2.0, http://openid.net/specs/openid-authentication-2_0.html (accessed 30 June 2012)
2. Morgan, R.L., Cantor, S., Carmody, S., Hoehn, W., Klingenstein, K.: Federated Security: The Shibboleth Approach. In: EDUCAUSE Quarterly, vol. 27, pp. 12–17 (2004)
3. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0., https://www.oasis-open.org/committees/download.php/35711/sstc-saml-core-errata-2.0-wd-06-diff.pdf (Accessed 30 August 2012)
4. González, J.F., Rodríguez, M.C., Nistal, M.L., Rifón, L.A.: Reverse OAuth: A solution to achieve delegated authorizations in single sign-on e-learning systems. Computers & Security 28, 843–856 (2009)
5. OAuth Core 1.0a, http://oauth.net/core/1.0a/ (accessed 30 June 2012)
6. The OAuth 2.0 Authorization Framework, http://tools.ietf.org/html/draft-ietf-oauth-v2-30 (accessed 30, June 2012)
7. Schiffman, J., Xinwen, Z., Gibbs, S.: DAuth: Fine-Grained Authorization Delegation for Distributed Web Application Consumers. In: IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY), pp. 95–102 (2010)
8. Alam, M., Zhang, X., Khan, K., Ali, G.: xDAuth: a scalable and lightweight framework for cross domain access control and delegation. In: Proceedings of the 16th ACM Symposium on Access Control Models and Technologies, SACMAT 2011, pp. 31–40. ACM, New York (2011)
9. OAuth 2.0 Threat Model and Security Considerations, http://tools.ietf.org/html/draft-ietf-oauth-v2-threatmodel-07 (accessed 20 August 2012)
10. Crampton, J., Khambhammettu, H.: Delegation in Role-Based Access Control. In: Proceeding of the 11th European Symposium on Research in Computer Security, pp. 174–191 (2006)
11. Toninelli, A., Montanari, R., Kagal, L., Lassila, O.: A Semantic Context-Aware Access Control Framework for Secure Collaborations in Pervasive Computing Environments. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 473–486. Springer, Heidelberg (2006)
12. Google Docs, http://www.google.com/google-d-s/b1.html (accessed 30 August 2012)
13. Facebook, http://www.facebook.com (accessed 30 August 2012)
14. Microsoft account, https://account.live.com/ (accessed 30 August 2012)