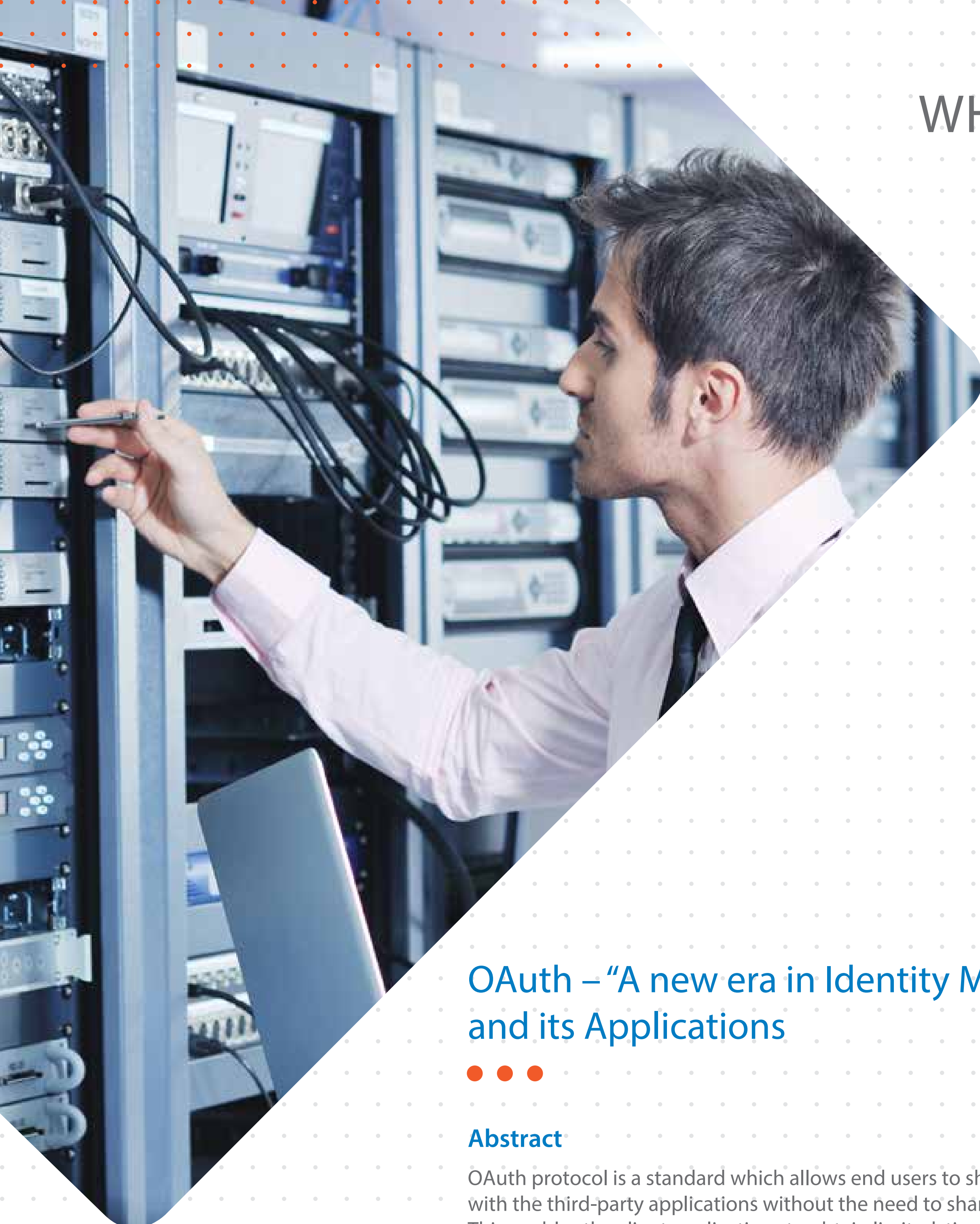

WHITE PAPER

OAuth – "A new era in Identity Management" and its Applications

Abstract

OAuth protocol is a standard which allows end users to share their web resources with the third-party applications without the need to share their credentials. This enables the client applications to obtain limited, time-bound access to HTTP service on behalf of the owner of the web resources. Here, the owner of the web resource does not share his credentials with the client application; instead he directly authenticates with the HTTP service and provides delegated access to client application for specific purpose. This whitepaper describes OAuth protocol – its overview, how it can help in Identity Management and its applications in various fields, saving significant amount of effort and cost, with an effective and centralized mechanism

Infosys
be more

Overview of OAuth protocol

As mentioned above, OAuth protocol can be leveraged by third party applications to obtain limited access to web resources on behalf of resource owner.

The traditional client server architecture allows the client to access server resources by providing their credentials - username and password. OAuth standard has introduced a third entity as resource owner.

OAuth basically defines four roles – client, authorization server, resource owner and resource server. The resource server hosts the protected resources and resource owner is the owner of these resources. The client here is the third party application requesting access to the protected resources on resource owner's behalf. Authorization server issues tokens to the client after resource owner is authenticated and owner grants authorization for delegated access to the client. The resource server and authorization server may be a single entity.

Thus, it facilitates the end users to authorize third party applications to obtain access to their resources with the use of tokens and without the need to share their credentials. It also enables the server to establish the identity of the client requesting the access. The client has to first obtain the authorization from the resource owner to access their resources. This permission is granted in the terms of tokens – wherein the token becomes a substitute for the user credentials. Unlike user credentials, tokens correspond to limited access and are time-bound.

In this process of delegation, resource owner first authenticates directly with the authorization server and then authorization server issues token to the client application. Then the client makes the request to the server for the resources, using two types of credentials – first to establish the identity of the client requesting the access and then to identify the resource owner who has authorized the delegated access to the client.

To begin with, the client makes a request for temporary credentials (here it is assumed that client has initially registered with the HTTP service for client credentials) to the server. The server responds with the temporary credentials. Then the client redirects to the server site to obtain approval of the resource owner. The server challenges resource owner to provide his credentials. Here, the resource owner provides his credentials (to server site and not to the client site) and once authenticated, the server asks resource owner to approve grant of access to the

client. If the resource owner authorizes the access, then redirection is made back to the client site – now client requests the authorization server for token using set of his temporary credentials. The server responds with the token after validation. Now client can access the protected resources on behalf of resource owner using this token. The client sends the request to resource server using this token and server responds with the requested resource after validation. The client can continue to access the protected resources using same token for the duration of the authorization granted by the resource owner.

A very well-known example of this whole process is that of photo printing site. The photo printing site can be assumed to be acting as client and the photo sharing site can be taken as the authorization server and resource server. The user (resource owner) has his photos uploaded in his profile on photo sharing site. Here, the user can grant authorization to photo printing site for access to his profile photos for printing. Additionally, photo printing site can be allowed to read user attributes from his profile on photo sharing site. For example, photo printing site can fetch user mailing address and can send the printed photos on that address – this in turn enhances user experience, as he gets the photos printed without much extra effort and without sharing his credentials.

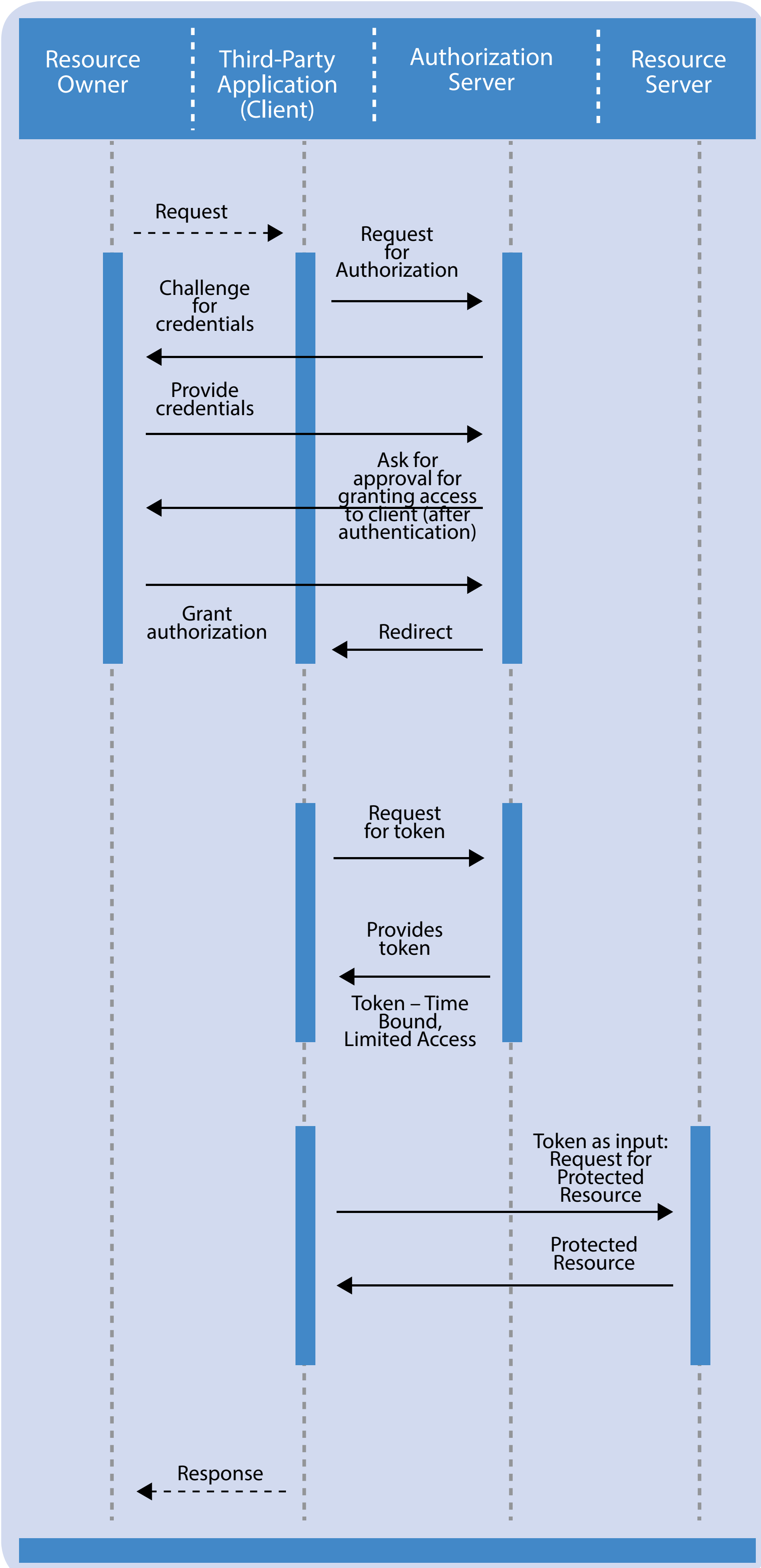

Fig 1: OAuth Flow

OAuth in Identity Management

Any enterprise planning to start any online service, for example online shopping store, can avoid the burden of setting up of identity store for management of their customer's identities. Instead, they can leverage OAuth standard and rely on any other existing digital identity store of any other enterprise, government etc.

They can tie up with other enterprise(s) and can redirect their customers to other sites for authentication – this way they can save on infrastructure to maintain user identity store and need not force their customers to open a new account. It can prove to be a win-win situation for all. The enterprise with existing identity store can charge the identity and authentication service it provides to other enterprises and also, the users need not maintain separate accounts for each site.

In India, citizens are being allotted Aadhar IDs - this can prove to be beneficial for many. As these IDs are now being increasingly adopted as proof of identity and address, financial institutions like banks can think of offering instant online account creation using OAuth wherein government can provide login credentials to citizens for their Aadhar site against Aadhar ID and users willing to open instant online account can visit bank site which will redirect to Aadhar site for authentication. Once user is authenticated with Aadhar site, he can approve the request to share his personal details like DOB, name, father's name, address etc. with the bank site. The bank site can create the account of the user instantly using these attributes. As Aadhar IDs are issued only after verifying user identity and address (after user submits relevant proofs for both), banks need not get these proofs again from customers and can create account at their end based on details fetched from Aadhar site which are authenticated and verified. Customers can thus avoid visiting bank branches for filling application form, submitting hard copies of identity and address proofs and can create account online instantly at the comfort of their homes. Banks also saves on paper costs, operational costs, time required to open their customers' accounts and need not keep a record of hard copy of proofs submitted by their customers. This will make the process centralized with different services linked to single unique ID provided by the government.

Applications

As mentioned in previous section, OAuth can be leveraged for identity management and banks can use Aadhar IDs to open instant online account etc. There can be many other applications and benefits of this standard which can make our life easier by saving on time and costs.

We all know that banks and financial institutions issuing credit cards and granting loans to customers update credit history of their customers, like payment status, with credit bureau which keeps record of all such transactions. Whenever customer visits any bank for loan, credit card etc., bank can have a look on his credit history with the bureau to arrive at the credit decision. Now the other quick alternative to this process can be using OAuth which can provide end customers more control of their data. Here, users can visit the bank site (from where he wishes to apply for fresh loan/credit card) and can redirect to his previous bank site for authenticating and sharing his corresponding payment history, outstanding amount etc. – this process will enable banks to arrive at the credit decision much faster.

Another benefit for financial transactions can be related to online usage of credit cards. Nowadays, people are increasingly using credit cards for online shopping, ticket booking, paying bills etc. But by doing so, they stand at a risk of sharing the credit card information like credit card

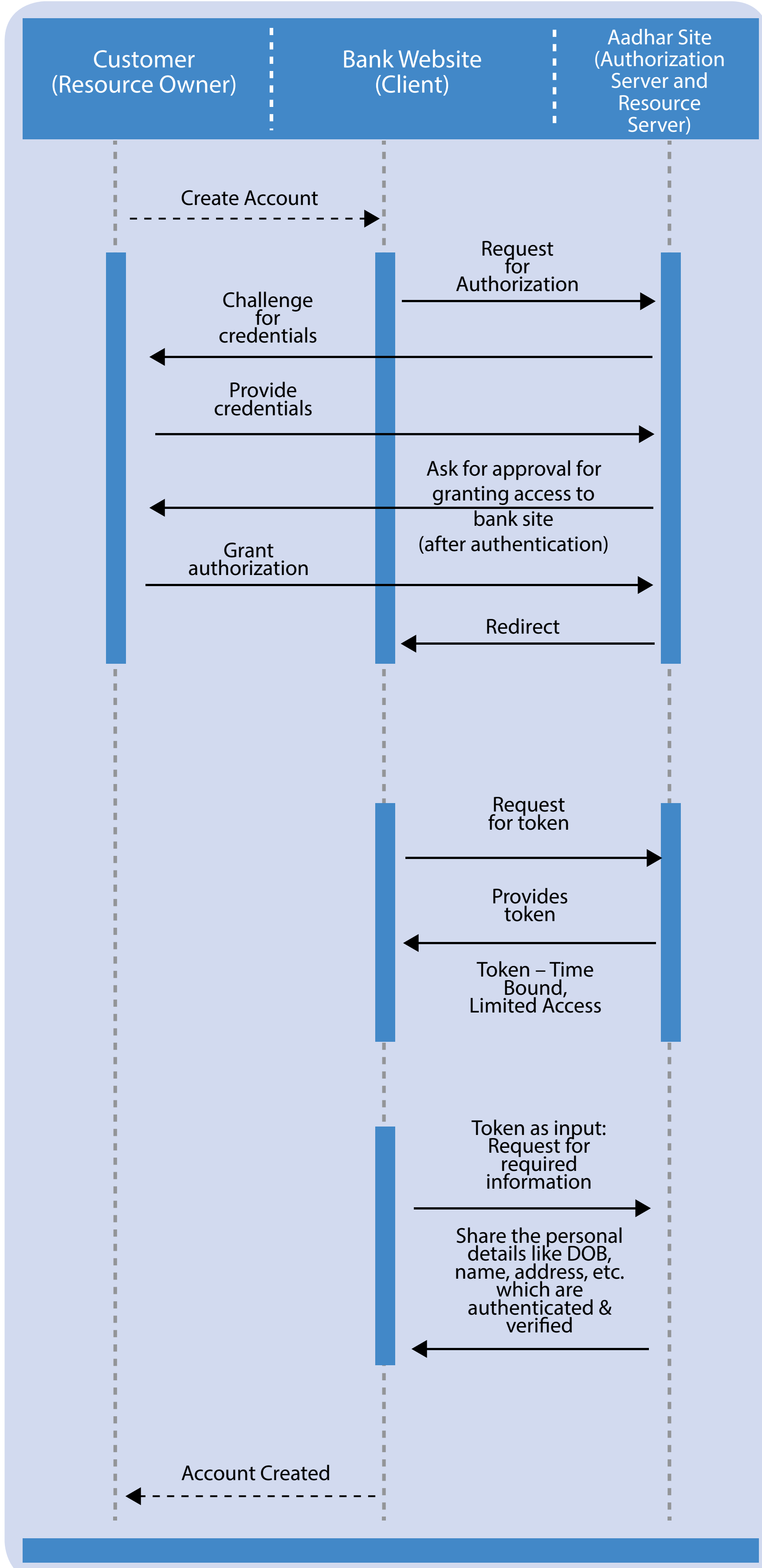

Fig 2: Online Bank Account Creation

number, CVV number, expiry date etc. with the vendor, which can be misused. Instead, banks and other financial institutions can act as OAuth providers and customers can directly login to bank's site and approve the payment through credit card to the vendor.

This standard can be used to centralize the scattered information of the users to a single repository which can be referenced by many and considered as authentic. For example: Google Health (which has been currently discontinued) was a centralized health information service by Google which allowed users to login into their accounts at partnered health service providers and sharing their health records like lab tests etc. with Google Health. Such central repository can be shared with any hospital the user visits in future – thus avoiding the need to conduct the tests again and saving on the costs involved.

There can be numerous such applications like sharing the school certificate online with the college authorities for admission, verifying the claims made by customer wherein the customer can collate their data and present in aggregate form, which is authenticated and trusted.

Thus this standard can make various processes quick, convenient and provide substantial savings on costs and time.

References

- http://tools.ietf.org/html/rfc5849
- http://tools.ietf.org/html/draft-ietf-oauth-v2-31

About the Author:

Ramanpreet Singh Lamba

Raman is working as Technology Architect with Infosys. He has more than 9.5 years of experience in IT industry. His areas of specialization include Access Management and web security. He has wide experience in working on multiple complex projects for Software Development and Maintenance. He took his degree in Computer Science Engineering in 2004. He can be reached at

Ramanpreet_Lamba@infosys.com

For more information, contact askus@infosys.com