SETLabs Briefings
VOL 7 NO 7
2009

Cloud Computing Identity Management

By Anu Gopalakrishnan

Online security concerns are on the rise and a robust identity management is what cloud needs now

Latest technology facilitates different service providers to unite their efforts to address a broader business space. It is possible that consumers hold multiple accounts with the service providers like e-bay, Gmail, etc. The visibility and scope of attributes for every identity has to be verified against a central trusted policy framing authority, assumed by the systems. In such a system, much is at stake if identities are not handled with extreme precaution. Such scenarios are common to high-end applications hosted on cloud computing environment. Identity management (IDM) assumes an upper hand in the whole area of cloud security. Cloud computing is an amalgamation of various technologies to meet the demands of an interdependent maze of software and services. This necessitates several IDMs, based on various technologies to inter-operate and function as one consolidated body over a cautiously shared user space. Hence IDM in clouds projects a number of new dimensions that traditional IDMs cannot meet.

Most cloud vendors have a simplified proprietary IDM solution with shortcomings that have to be understood. The challenge in this area is that there are considerable efforts towards outsourcing the IDM that gave birth to the concept of identity-as-a-service (IaaS) [1]. IaaS vendors focus on comprehensive, interoperable and quick-to-deploy solutions.

UNDERSTANDING THE NEW DIMENSIONS OF IDM IN CLOUDS

The evolution of cloud computing from numerous technological approaches and business models such as SaaS, cluster computing, high performance computing, etc., signifies that the cloud IDM can be considered as a superset of all the corresponding issues from these paradigms and many more. An IDM in cloud has to manage – control points, dynamic composite/decommissioned machines, virtual device or service identities, etc. Cloud deployments are dynamic with servers launched or terminated; IP addresses dynamically reassigned; and services started or decommissioned or re-started. So, as traditional IDM, merely managing users and services is not sufficient. When a deployment or service or machine is decommissioned, the IDM has to be informed so that future access to it is revoked. IDM should ideally store its details till it becomes active. Meanwhile access to its relevant stored data has to be monitored and granted by the defined access level for that mode as mentioned in SLA. Traditional IDM is not directly amenable for cloud computing due to these peculiarities of cloud.

Today's cloud requires dynamic governance of typical IDM issues like, provisioning/de-provisioning, synchronization, entitlement, lifecycle management, etc.

IDENTITY LIFECYCLE MANAGNEMENT

Lifecycle management incorporates an integrated and comprehensive solution for managing the entire lifecycle of user identities and their associated credentials and entitlements. Functionally, it is divided into two components – the provisioning component and the administrative component. Administrative component defines delegations rules, providing self-service components to change personal details or make requests to the users. Delegation of administrative rights to local group or process-in-charge is crucial for a volatile and dynamic cloud based scenarios. Decentralizing the tasks will reduce the load on the authenticator component and also save time in making access control decisions. Figure 1 illustrates the various components of lifecycle management.

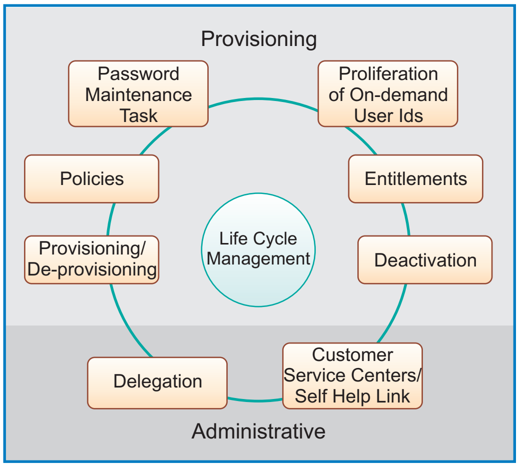
Provisioning
Password Maintenance Task
Proliferation of On-demand User Ids
Policies
Entitlements
Life Cycle Management
Provisioning/ De-provisioning
Deactivation
Delegation
Customer Service Centers/ Self Help Link
Administrative

Figure 1: *The Identity Life cycle Management*
Source: *Infosys Research*

Provision and De-provisioning

In cloud, provisioning means just-in-time or on-demand provisioning and de-provisioning stands for real time de-provisioning. Just-in-time provisioning indicates the federation of user accounts without sharing prior data, based on some trust model. Service Provisioning Markup Language (SPML) provides XML based structures for representing provisioning or de-provisioning requests intended for identity lifecycle management [2]. SPML can make use of Service Administered Markup Language (SAML) assertions and facilitate a complete trust model between senders and receivers. SAML defines an XML based framework for exchanging security information for enabling SSO or identity federation regardless of the underlying architecture. OASIS Security Services is currently working on developing a SAML 2.0 profile for SPML. SAML can help SPML to establish trust and quantity, a subject against which the SPML provisioning request is targeted. This makes just-in-time provisioning and real time de-provisioning possible.

Real time de-provisioning of a user account has to synchronize instantaneously with all participating service providers. Any delay in de-provisioning could lead to security vulnerability. Some of the issues like — ways in which de-provisioning of one user affects the other federated identities in cloud are matters of judgment on the functionality of the application deployed on the cloud.

Entitlement

Entitlement refers to the set of attributes that specify the access rights and privileges of an authenticated security principal. Lack of interoperable representation of this information poses a challenge as the information needs to be exchanged among different cloud based service providers. In the absence of interoperable format, expensive and customized syntactic translation components are needed. The semantic aspect still remains to be tackled.

While some applications like SalesForce have built-in control for entitlement and authorization control for multiple attributes, others require the help of OAuth or similar such technologies [3].

Proliferation of On-demand User ID

Proliferation of on-demand user ID is a big concern in cloud computing IDM as the occurrence of multiple identities for the same user in multiple service providers' security repositories cannot be ruled out. A simple way to overcome this problem is by the adoption of OpenID mechanism [4]. OpenID works by making one primary user id as the key to authenticate a single end user with multiple service providers. However, the difficulty in this approach lies in the trust propagation and development of trusted relationships [5].

Synchronization services help expedite the roll-out and expansion of federated identity management capabilities by enabling services in cloud to federate accounts and other data necessary to build up trust relations.

CLOUD ARCHITECTURE

Cloud architecture plays an important role in choosing your IDM, SaaS or the all-in-one Platform-as-a-Service (PaaS) [6]. SaaS requires only application access, whereas PaaS will require system access (for accessing the underlying platform) as well as application access (for accessing the hosted application on the underlying platform). Both require a common IDM that can integrate well into the existing authentication mechanism. The third type of cloud architecture is Infrastructure-as-a-Service (IaaS), which is not mentioned explicitly, since the IDM requirement of PaaS and IaaS are comparable. Consider one of the most common SaaS IDM implementation using ping identity. Ping identity works by deploying the technology behind the firewall and making the identities exportable [7]. This IDM mechanism allows integration of a number of authentication mechanisms such as Microsoft Windows based authentication, LDAP authentication, CA site minder, etc. It is deployed on top of the existing authentication infrastructure and the deployment is quite efficient and fast. It uses SAML to transfer credentials. It can be perceived as a layer of abstraction over the traditional IDM that fights the challenges of IDM. This aspect of it makes this IDM architecture easy to deploy and dynamic.

PaaS is commonly defined as the delivery of a computing platform and solution stack as a service. It includes workflow capabilities for application design, application development, as well as application services such as team collaboration, web service integration, etc. PaaS IDM automatically scales up to include all these features. This is illustrated in Figure 2 overleaf.

PaaS IDM has to address various functional modules like source control, test

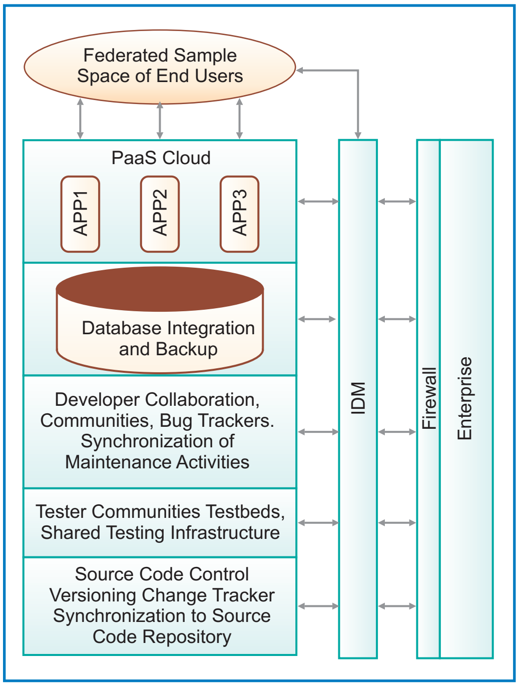
Federated Sample Space of End Users
PaaS Cloud
APP1
APP2
APP3
Database Integration and Backup
Developer Collaboration, Communities, Bug Trackers. Synchronization of Maintenance Activities
Tester Communities Testbeds, Shared Testing Infrastructure
Source Code Control Versioning Change Tracker Synchronization to Source Code Repository
IDM
Firewall
Enterprise

***Figure 2:** PaaS IDM*
***Source:** Infosys Research*

modules, development communities, etc. For the sake of simplicity, the PaaS IDM could adopt a Role-Based Access Control (RBAC) system to handle each of this and its user space. An RBAC system for source control will allot minimum set of privileges to the developer accounts and essential services, depending on the interdependency of the applications hosted on the platform. For test communities, IDM manages tester accounts, privileges, auto-run test suites and knowledge collaboration portals of the tester communities required for hosting a test bed. In case of development communities, IDM manages the collaboration of developer communities, access and privilege of each group of developer, the bug tracker system, etc. The cloud could also expect IDM to handle the database challenges, by controlling the access and synchronization with the in-premise segments. In addition to all these, IDM handles the SaaS based challenges of federated user space.

Due to vender lock-ins, the primary limitation with PaaS happens to be a fact that the complex IDM solution designed for PaaS is rendered useless while migrating to another cloud. A simple slice of IDM requirements are plotted here to illustrate the complexity of the PaaS IDM.

USER CENTRIC ACCESS CONTROL

The traditional model of application-centric access control, where each application keeps track of its collection of users and manages them, is not feasible in cloud based architectures. This is more so, because the user space maybe shared across applications that can lead to data replication, making mapping of users and their privileges a herculean task. Also, it requires the user to remember multiple accounts/passwords and maintain them. Cloud requires a user centric access control where every user request to any service provider is bundled with the user identity and entitlement information [8]. User identity will have identifiers or attributes that identity and define the user. The identity is tied to a domain, but is portable. User centric approach leaves the user with the ultimate control of their digital identities. User centric approach also implies that the system maintains a context of information for every user, in order to find how best to react to in a given situation to a given user request. It should support pseudonyms and multiple and discrete identities to protect user privacy. This can be achieved easily by using one of the open standards like OpenID or SAML.

FEDERATION OF IDENTITIES

On the internet, it is likely that each user ends up with multiple credentials and multiple access permissions across different applications provided by different service providers. These fragmented logins present a challenge to the users and service providers, in forms of synchronization of shared identities, security, etc. There is a strong need for an intrinsic identity system that is trusted across the web and within enterprises and unambiguously identifying users.

Federation of identities maintained by the multiple service providers on the cloud is very critical to cloud based service composition and application integration. An expected issue in this regard is the naming heterogeneity. Different SPs use different factors for authentication like account number, email ID, PayPal ID, etc. Also, when transactions traverse multiple tiers of service hosted in clouds, the semantics of the context of identity information has to be properly maintained, constrained and relaxed as per specific needs. Consider a complete transaction cycle for an e-bay purchase, based on PayPal account. It traverses from e-bay to supplier, through various tiers in supplier's domain to get approvals, release and shipping. Then it goes through PayPal to approve, validate, release the pay, bill the amount to the customer, etc. For each step, the federation authority decides the essential attribute of the customer to be shared with each department.

The user identity mapping in the previous environments have been one-to-one, or in other words, user ID to single user profile. In cloud architectures the mapping challenge is many-to-one, one-to-many and pseudonyms. Pseudonyms are for privacy protection details, when a user does not want his identity to be tracked as he crusades various domains.

Another issue is the trust relation setup between the service providers of the federated world. Currently it is based on policy files framed by the local authority, depending on various factors like the domain trust information automatically fed in by the trust authorities. This is not a scalable or flexible model that can meet cloud computing demands. Cloud scenarios require dynamic trust propagation and dynamic authorization.

VOLATILITY OF CLOUD RELATIONS

In a traditional model, the IDM is based on the long-term relation of a user to an organization or trust domain. In cloud, which represents the current e-commerce world, the relationships change dynamically and quickly, and the IDM has to incorporate all that. Any retrieval or cache of the volatile data has to be done cautiously. The possible damage of using old data should be studied. Like, if the user has changed his password login with old password, it should be restricted and locked in all the applications that are participating in the identity federation. Live data fetching, domain name resolution, canonicalization of the data like URL, account IDs, etc., are the challenges.

SCALABILITY

Cloud requires the ability to scale to hundreds of millions of transactions for millions of identities and thousands of connections – with short/rapid deployment cycles. Performance has to be N+1 scalable across the globe and deployments agile and quick (weeks not quarters/years). With the software today it takes ~6 months to make a single SAML/SSO connection and it doesn't address the access control and compliance issues. Open Cloud Manifesto states that clouds have to dynamically scale up and down, so that nobody needs to hoard resources to handle peak hours [9].

INTEROPERABILITY

The mass expects the cloud to provide a IDM solution that can interoperate with all existing IT systems and existing solutions as such or with minimum changes. Seamless interoperation with different kinds of authentication mechanism such as the Microsoft Windows authentication, SSO, LDAP, SAML, OPENID and OAUTH, OpenSocial, FaceBookConnect, etc., is what is expected of cloud. The syntactical barriers have to be bridged. It requires an authentication layer of abstraction to which any model of authentication can be plugged in and off dynamically.

TRANSPARENCY

Security measures assumed in the cloud must be made available to the customers to gain their trust. There is always a possibility that the cloud infrastructure is secured with respect to some requirements and the customers are looking for a different set of security. The important aspect is to see that the cloud provider meets the security requirements of the application and this can be achieved only through 100% transparency. Open Cloud Manifesto exerts stress on transparency in clouds, due the consumer's apprehensions to host their applications on a shared infrastructure, on which they do not have any control [9]. Transparency can be achieved by complete audit logging and control.

PATTERNS IN CLOUD IDM

Based on the insights gained so far three patterns in cloud IDM can be concluded. The ideal scenarios for each pattern are also mentioned.

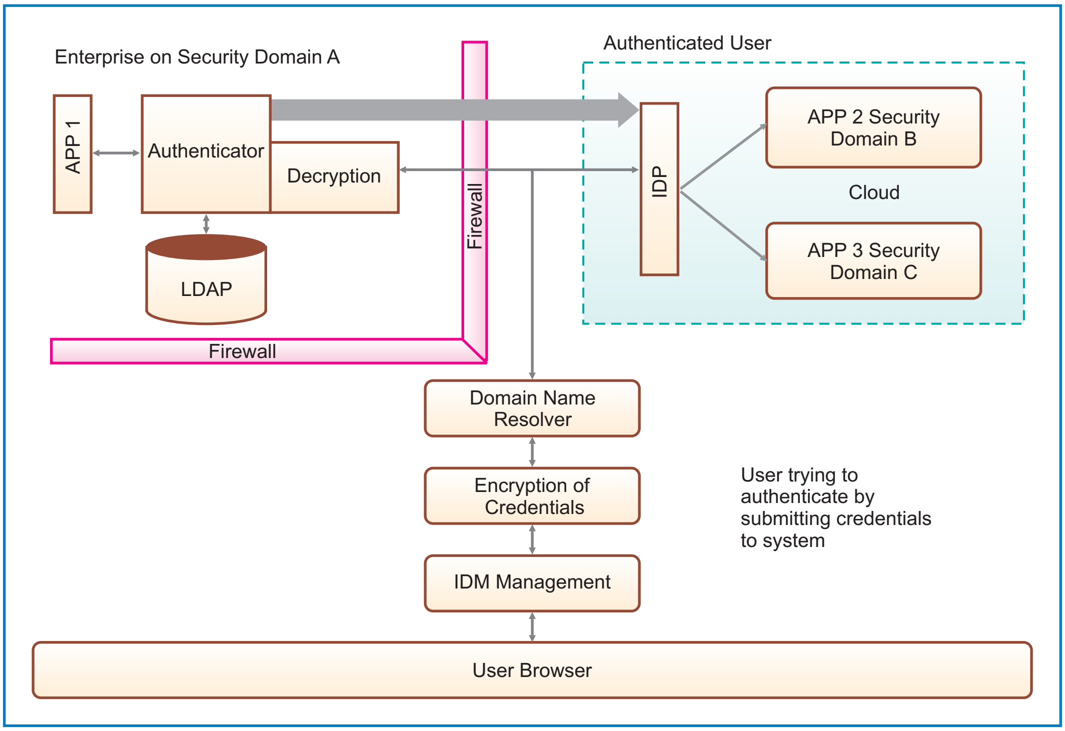

***Figure 3:** Trusted IDM Pattern* **Source:** *Infosys Research*

Trusted IDM Pattern

This pattern is intended for a smaller or even for a private cloud that requires security. Scalability is definitely not a feature of this cloud. But Google App Engine (appengine.google.com) that follows this pattern assures that the scalability is not a major concern at the moment as the number of requests that could be tunneled through simultaneously is quite large. The main feature of the pattern is that the authentication is always performed within the firewall. The credentials are submitted to the IDM component and it takes care of encrypting and tunneling the credentials through a secure channel to the authenticator. IDM is independent of the authentication mechanism. Hence deployment and integration is fast and efficient. Once the user is authenticated in by any authentication mechanism, then rest of the participating servers trust the user. The attributes of the user can be shared using some mechanism like SAML. Authorization can be effectively handled by XACML. A basic model of this pattern is illustrated in Figure 3 on page 50.

External IDM

This pattern is very similar to the initial pattern but for the fact that the credentials are submitted directly to the authenticator [Fig. 4]. The credentials can be collected by a different browser window, channeled by SSL. The pattern is intended for a public cloud. The IDM concentrates only on domain resolution and triggering of the authenticator to resolve the authentication. This is the architectural pattern adopted by ping identity. In ping identity, domain resolution is done by referring to a spreadsheet of valid users that is always kept updated. It can also be achieved through other mechanisms like standard domains name

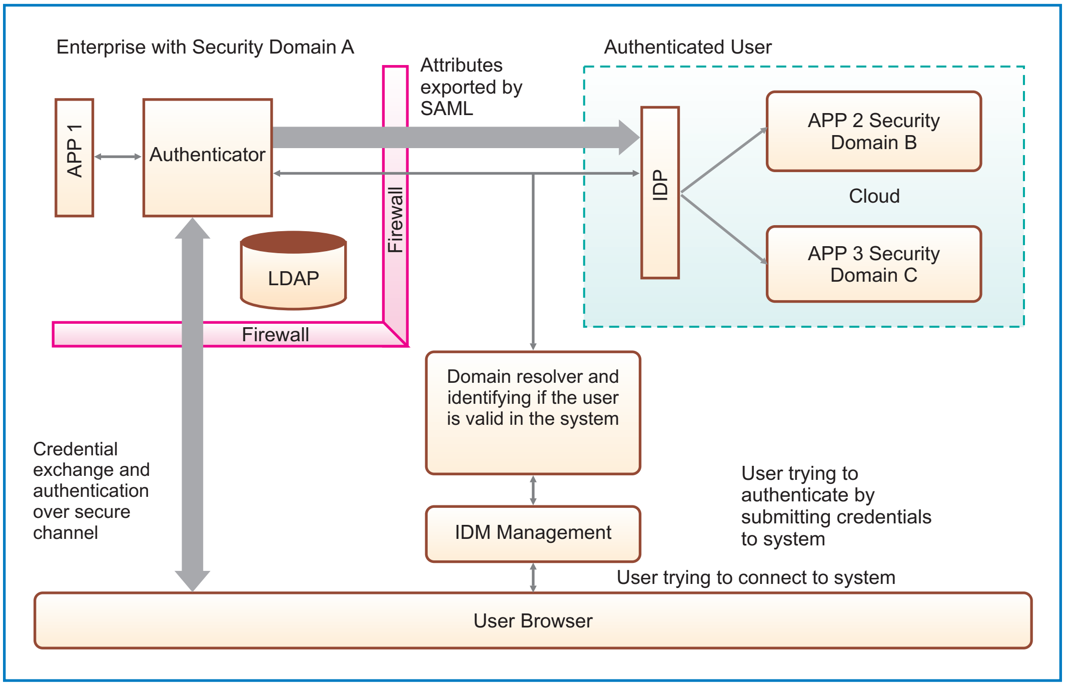
Enterprise with Security Domain A
Attributes exported by SAML
Authenticated User
APP 1
Authenticator
IDP
APP 2 Security Domain B
Cloud
APP 3 Security Domain C
LDAP
Firewall
Firewall
Domain resolver and identifying if the user is valid in the system
Credential exchange and authentication over secure channel
User trying to authenticate by submitting credentials to system
IDM Management
User trying to connect to system
User Browser

Figure 4: External IDM

Source: Infosys Research

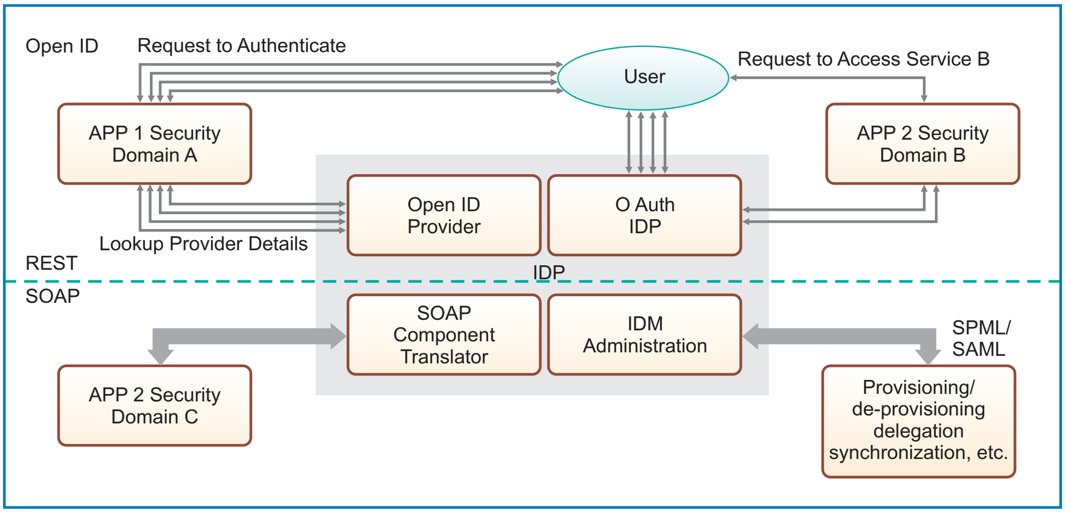

Figure 5: *Interoperable IDM* ***Source:*** *Infosys Research*

resolution, discovery or YADIS protocol, or XRDS query, etc., depending on the underlying technology used. The same drawback of pattern 1 exists in pattern 2 also. Scalability is an issue. Symplified (www.symplified.com) is vendor on cloud IDM, whose solution has close resemblance to this pattern.

Interoperable IDM Pattern

This pattern illustrates a cloud to cloud scenario, using OpenID and OAuth. The identity mechanism used, will understand and interoperate multiple identity schemes. OpenID is an open and decentralized standard for user authentication and access control, by allowing users to logon to multiple services with the same digital ID. Any service provider can authenticate the user in to the system. OAuth is again an open protocol that enables a user to grant permission to a consumer site to access a provider site without any sharing of credentials [10]. SPML is used for XML based IDM LC. This is extremely useful for an e-commerce web world where there are multiple service providers based on a common user space. The central identity system, understands all technologies used for authentication like SAML, OpenID, OAuth, etc. Let us assume that the central identity system to be collection of modules, each handling a technology, taking to a common user space and a policy database. The information is converted to different formats, depending on the technology used like OpenID, or SAML, or WS-Security and conveyed to the participating service providers [Fig. 5].

A brief comparison of the three patterns is shown in Table 1.

CONCLUSION

Of the emerging technologies cloud computing has a lot of substance. The huge set of challenges it has brought with it has to be captured and tamed to produce more benefits. Choice of IDM design for any cloud should be tailored to suit the definition of that particular cloud and open to any kind of enhancements the cloud is bound

Features	Trusted IDM Pattern	External IDM	Interoperable IDM
Security of Credentials	Very Secure	Submitted to IDP Network	Depends on Authentication Mechanism
Interoperability	Interoperable, since it is oblivious of the underlying authentication mechanism	Interoperable	Interoperable to any Authentication Mechanism and Technology
Type of cloud the pattern is best suited for	Private Cloud	Can be used in public clouds since the credentials are always submitted directly to the authenticator module and secrecy is maintained	Huge Public Clouds over Multiple Technologies
Scalability	Not Scalable Easily	Not Scalable Easily	Scalable
Speed of Deployment and Implementation	Very Fast	Fast	Speed depends on the number of technologies required
Examples of this Pattern	Google App Engine's SDC	Ping Identity	Proposed Design

Table 1: *Summary of the Patterns* ***Source:*** *Infosys Research*

to have in future. Essentially the design should be capable of incorporating any number of trust domains and of maintaining an effective shared user pool. As the next generation IDM IaaS, a user centric identity management is intended to be a complete all-round solution addressing all possible issues of cloud IDMs [11]. It may be the answer to the growing complexity of IDMs. The intent is to take away the complexity of IDM away from the enterprises, thereby allowing them to direct their energy and resources on their own functions, while the IaaS vendors provide the best solution or IDM based on their expertise.

REFERENCES

1. Open Cloud Manifesto, Spring 2009. Available at http://www.opencloudmanifesto.org/opencloudmanifesto1.htm
2. RSA's contribution to Cloud security guidelines. 2009. Available at http://www.cloudsecurityalliance.org/guidance
3. OAuth. Available at http://oauth.net/ OpenID Authentication 2.0 Final, 2007. Available http://openid.net/specs/openid-authentication-2_0.html
4. Illustration of OpenID based on Plaxo's use of Yahoo OpenID. Available at http://www.plaxo.com/api/openid_recipe
5. Luis M Vaquero, Luis Rodero-Merino, Juan Caceres and Maik Lindner, A Break in the Clouds: Towards a Cloud Definition, Cloud Architectures, Vol 39 No 1, Jan 2009. Available at http://delivery.acm.org/10.1145/1500000/1496100/p50-vaquero.pdf?key1=1496100&key2=0736171521&coll=GUIDE&dl=GUIDE&CFID=50720541&CFTOKEN=61415293
6. Ashish Jain, A blog on Ping Identity, Jan 12, 2009. Available on http:// itickr.com/?cat=29
7. Service Provisioning Markup Language Specification, version-1, June 2003. Available atxml.coverpages.org/PSTC-CS-SPMLCORE10.pdf

8. Christian Emig, Frank Brandt, Sebastian Kreuzer and Sebastian Abeck, Identity as a Service – Towards a Service-Oriented Identity Management Architecture, Lecture Notes in Computer Science, 2007. Available on http://www.springerlink.com/content/5865u474424qw751/. ◎

Author's Profile

ANU GOPALAKRISHNAN

Anu Gopalakrishnan was a Product Technical Architect in SETLabs, Infosys.

For information on obtaining additional copies, reprinting or translating articles, and all other correspondence, please contact:

Telephone: +91-40-67048455

Email: SetlabsBriefings@infosys.com

Infosys acknowledges the proprietary rights of the trademarks and product names of the other companies mentioned in this issue of SETLabs Briefings. The information provided in this document is intended for the sole use of the recipient and for educational purposes only. Infosys makes no express or implied warranties relating to the information contained in this document or to any derived results obtained by the recipient from the use of the information in the document. Infosys further does not guarantee the sequence, timeliness, accuracy or completeness of the information and will not be liable in any way to the recipient for any delays, inaccuracies, errors in, or omissions of, any of the information or in the transmission thereof, or for any damages arising there from. Opinions and forecasts constitute our judgment at the time of release and are subject to change without notice. This document does not contain information provided to us in confidence by our clients.