A developer's basic guide to cloud identity management – edited by Dr. Bernard Bowitz, 2019

The days of placing virtual walls around applications and data are fading. Organizations looking for better, more agile security are choosing to integrate identity access and management (IAM) and governance directly into the applications.

Cloud application delivery of IAM systems is making this transition a lot easier and cheaper, but this shift also means that developers will now be primarily responsible for identity management. In this article, I'll introduce some of the activities, patterns, and task lists that will help developers get started with integrating cloud identity management.

Protecting the Crown Jewels: Privileged Account Management

Identity management in a nutshell

Identity management provides security and governance by only allowing individuals, services, and processes with an authorized *identity* to access the appropriate resources at appropriate times. Resources can be data, services, IoT devices, or any other computing entity. They also have identities. Every person, application, and device gets an identity, and administrators, developers, and applications create policies that define the limits of various relationships between identities. One identity might have no access, limited access, or unlimited access to a resource. It depends on how the policies are defined.*Get Started with Seamless App Sec in a Single Day*

The explosive growth of cloud identity management

In cloud environments, IAM is a commonly accepted best practice, since there's no practical way to secure resources without identity management. Clouds mean that resources (services, storage, compute, etc.) will be widely distributed. That's why there is no single piece of software that can centrally secure the systems, so IAM is a way to build authentication gates into each of the distributed resources, distributing security as well. The IAM system can then centrally manage security by communicating with the gates on each resource.

The well-known benefits of cloud computing such as adaptive pricing structure and faster, easier deployment are also the drivers of the cloud IAM market. As the primary pattern for security and governance in the cloud, IAM SaaS is growing in tandem with the increasing migrations to cloud applications.

According to an IAM market report from Markets to Markets, the IAM market will be worth $18.3 billion by 2019. The report defines and segments the IAM software market into various subsegments, with in-depth analysis and forecasting of revenues. It also identifies drivers and restraints for this market, with insights on trends, opportunities, and challenges.

Cloud security shifts to developers with IAM

Modern cloud applications are *identity-enabled* to provide the security and governance services that enterprises need. As I mentioned earlier, this means that, rather than placing virtual walls around applications and data, you integrate identity management and governance directly into the applications. This is a major shift for many organizations. It means that instead of sysadmins and operations, developers will need to build the bulk of these organizations' security and governance features.

The specific activities may include adding the ability to:

- Have cloud applications check the IAM system to determine access rights before proceeding with operations
- Filter through data that applications are allowed to use based on the identities of the data, the application, and the application user
- Encrypt data communications as needed, regardless of whether an authorized identity is accessing it or not
- Encrypt data storage as needed
- Dynamically determine access rights and limits while the application is running

There can be hundreds of patterns necessary in your application to make it identity-aware. By integrating the features and functions of an identity systems into your applications, this is very manageable.

Typically, the use of identities for access management, authentication, and governance follows this pattern.

1. Process start
2. Identity retrieval
3. Identity validation
4. Resource access using identity
5. Identity validation and understanding of limitations
6. Resource access with authentication and limitations
7. Release of resources

8. Process end

The ways in which applications leverage identity to access resources will be different for every organization, depending on application requirements and the features of the identity management system. However, most organizations want to provide secure APIs (usually RESTful APIs) that they can leverage from their applications, or even from their database.

Database access and exception handling

Access to the cloud or non-cloud-based data is a fundamental activity for cloud applications, and the use of identity is a way to ensure that data is accessed by the appropriate application, people, or services. Overall, access to the data is controlled at several levels: database, entity, object, instance.

It depends on the data model you’re looking to leverage, but the basic concept is that the IAM system will provide the cross reference to determine whether the person or application can access the entire database, or just a smaller section of data, possibly down to just a single instance or record of data. The IAM sits at a level above the application; however, the application must understand how to leverage the identity management system to deal with access and governance. If the application does not, then those attempting to access the data will receive a platform error message.

An application has to understand the identity of the application user first. Then the application should map out the access rights upon identifying and validating the user, and the application should never attempt to access data that it, or the user, has no rights to access. This is something only the developers can build into a system.

Exception handling should be built into the application as well. This is so applications know how to respond effectively when problems come up. For example, maybe the access rights are altered while the application is executing, or access to the database fails for some reason. The application must have automated capabilities to deal with such issues as they occur, never leaving the application in an unstable state and always keeping the user clearly informed.

Integrating IAM into automated testing

Automation is everything these days. The use of DevOps processes and tools, including automated testing systems, makes using identities much easier for developers. Automated testing systems are necessary to test and validate the use of identities within each application and database. This helps ensure that developers review every aspect of identity use and that any issues are externalized and fixed before the application goes into production.

The good news is that there are already testing tools available that are aware of security and governance dependencies using identity management systems. It’s just a matter of creating the testing scripts that are aware of your IAM system.

Don't let your guard down if you already have testing tools that are IAM-aware. There are a lot of typical issues you need to watch out for, including misalignment of identities and access rights within the application, which can cause issues during execution.

An application’s inability to deal with dynamic changes to the identity repository. For instance, if a user is pulled out of the system, all access right should be terminated, and the application stopped.

Building your own identity-enabled applications

Identities are no longer a part of the infrastructure; they're part of the application and the data, which means they are firmly in the hands of developers. However, identity management is a relatively new role for cloud and non-cloud developers. If most of your team isn't familiar with building identity-driven security and governance, now is the time for them to learn.

Here is a list of things that should be on your task list for now:

- Be sure to select cloud-aware identity management systems that are right for your applications and platform. This means understanding your requirements and using those requirements to come up with a sound solution that could include many different identity management components.
- Splurge on training. Developers need to become experts at identity management and using identities within the development process.
- DevOps needs to change. The use of identities means that automated integration and testing have to change as well. This means finding or building new tools and services and dealing with new ways to stage and deploy the applications. This could be the most costly thing you’ll do, in terms of time and money.
- Look to new approaches, such as centralized trust and new versions of industry standards, such as SAML. These approaches need to be on your radar or on your roadmap.

Developers have to face the fact that the new approaches to security and governance need to provide flexibility that we’ve never seen in the past. It doesn’t matter if we’re building and deploying containers or microservices on cloud or non-cloud platforms. The use of identity-based approaches and technologies are here to stay. Going forward, developers need to drive application development along with enterprise security and governance.