From: www.cio.com

How Secure is That Cloud Vendor? Part Two

– David Taber, CIO

February 01, 2011

Last week, we explored the 7 basics of cloud computing vendor security, including identity, authentication, encryption, ILP/DLP and audit trails. Now here comes the deep dive: access control.

Depending on what your cloud application is doing, there will be several levels of access control. The simplest is at the file level: the operating system will enforce user/group/world or access control lists on every file in the system. Once a remote process has had credentials authenticated, the operating system will properly enforce the C/R/U/D privileges according to the running user's role or profile. Most operating systems also enforce access control on processes and system objects. This course-grained access management is basic and fairly uniform across cloud systems — not much to discover there.

In a similar way, DBMS systems will be fairly uniform in the way they enforce security across clouds. All DBMS security systems offer at least table-level enforcement of access control, and most offer column-level restrictions as well. Further, the DBMS will enforce locks at the transactional level to manage contention and prevent corruption. The cloud doesn't really change much here, so there's not much of a challenge at this level.

Things get a lot more interesting at the application level, as cloud application vendors have had to put a lot more work into their access control mechanisms from the outset. This is where some of the real work of multi-tenancy has taken place. But the access control enforcement models tend to be very different across cloud application categories, as well as among specific vendors. For example, the details of data access and sharing are a whole lot different in a social network than they are in an accounting system, an ecommerce engine, an ERP system, or a CRM system. This is where you'll find the access control challenge when trying to integrate across clouds.

To make clear examples, I'm going to be using specifics from Salesforce.com. Please don't interpret this as shilling for them — it's just that they are a common target for cross-cloud integration and I understand their security model best.

Salesforce.com's access control is defined and enforced through a series of filters that is deceptively simple. The filters each apply to groups or classes of user accounts:

• Profiles define whether a class of users can see a table, object, or area of functionality
• Profiles define whether a class of users can see a table column (object attribute)
• Roles define whether a class of users can see a record (row or instance)
• Record types define which Profiles can see individual cells within a record, and can be used to restrict access to nearly any function or object class.

There are some modifiers for these filters to allow delegation and to broaden the span of control for superusers, but the experience for most users is that applying all these filtering mechanisms provides a very fine level of access control. Sometimes to the point of irritation. Locking and filtering may be context or state dependent...and get in the way of some required business needs. So the system provides exception methods for data sharing, granted at either the individual or group level.

All this is within a single Salesforce instance: they also have a mechanism for bridging instances so that companies can share individual records with their business partners in a separate instance ("org") of SFDC.

The Wonders of Security

So here's the point: does any of your other cloud applications use this security model? Sure, there might be equivalent concepts...but the specifics will be different for every cloud application vendor.

When trying to integrate across clouds, the goal is to enforce the tightest security model of the applications involved with an object, context, or transaction. This can be a real challenge for your developers.

The last thing you want to do is try to emulate all the security logic outside of the applications. Even if your developers understood the model exactly (and trust me, this will be an ongoing point of confusion), there are too many things to get wrong — both in the coding and the ongoing configuration / administration of the security model.

But that's the issue: when integrating cloud applications, the external application classically needs to be able to access records for (or on behalf of) a wide range of users, if not a wide range of tables (objects). For example, an ecommerce system needs to be able to create and close opportunities in the CRM system for any sales territory. The external cloud ecommerce system doesn't need to impersonate the rep (in most situations, you wouldn't want it to), but it does need CRM privileges akin to the highest level of sales management to close the deal. The answer in this case is to have the ecommerce integration point use credentials that are equivalent to the relevant sales VP.

That example worked because the external system does essentially one thing, without being triggered by some internal request from the CRM system. What if, instead, the external system is providing a service that is confidential, and access should be highly user-dependent (e.g., viewing compensation records or commission checks)? In this situation, the external cloud service request must not be made directly from the user's browser (for example, presented through a screen mashup or triggered via Javascript and JSON), but instead be regulated though the data mechanisms provided by the user's "native" back-end system. The cloud with the looser security model should be accessed only through the regulation of the tighter model, using an API call-out from that cloud. In this specific example, the user's request to see commission data would call out from the CRM cloud to the commission cloud, and then pull the results into the CRM tables where the proper level of access control can be applied.

Essentially, the Access Control strategy must be thought through and resolved independently for each class of cross-cloud conversation. That said, for testability, maintainability and reliability, you'll want to have as much re-use of these security mechanisms as possible.

I Love Standards — There are So Many to Choose From

Because of the differences in access control models and enforcement mechanisms across clouds, providing a uniform level of access and auditability seems to be an evergreen challenge. Unfortunately, I don't see any standardization bodies that will make this problem go away — if anything, they may institutionalize it.

David Taber is the author of the new Prentice Hall book, "Salesforce.com Secrets of Success" and is the CEO of SalesLogistix, a certified Salesforce.com consultancy focused on business process improvement through use of CRM systems. SalesLogistix clients are in North America, Europe, Israel, and India, and David has over 25 years experience in high tech, including 10 years at the VP level or above.

Follow everything from CIO.com on Twitter @CIOonline.