How Adobe DRM Works in SimplyE

And problems that arise

Edited script of a discussion from the Library Simplified Slack #general channel.

Initial Discussion: Friday, Jan 18th, 2019

Paul Swanson [10:21 AM]
One thing I'd like to learn more about is how Adobe DRM works in SimplyE: a description of how the NYPL-hosted Adobe ID system works and what are its implications (activation limits, not being able to read a title in simplye that is checked out on the vendor platform, etc.), along with Adobe DRM licensing terms (renewal schedule, limitations, etc).

Robert Williams [10:59 AM]
There is a document in the Datalogics Knowledge Base and a Datalogics PDF that describe the Adobe Vendor ID service that SimplyE employs in general terms:

- https://kb.datalogics.com/display/KB/Vendor+ID+Workflow
- http://www.datalogics.com/pdf/VendorID-DS.pdf

Adobe's introductory page on its Adobe Content Server product has a colorful, simplified diagram of the fulfillment process in ACS. However, it shows only the use of a user-specific Adobe ID account. The Vendor ID service would be another panel that fits between the Retailer/Distributor and User panels, as it intermediates fulfillment between the user and Adobe (https://www.adobe.com/solutions/ebook/content-server.html). The actual process of how it works is a nasty looking trail of back-and-forth communication between at least four systems (including Adobe's signing service). As Mark and Winnie suggested, I'll leave the sweaty details of SimplyE specifics to `@leonardr`!

Leonard Richardson [11:13 AM]
Let's start with the most common case: you have a library barcode and you want to read a book that's encrypted with Adobe DRM (Adobe Content Server, ACS). The vendor has the Adobe Content Server that provides the encrypted ebook file ("bookbytes"). Your library has an ILS that knows whether your barcode+PIN are valid. The SimplyE Circulation Manager knows how to talk to the vendor and to the library's ILS.

The Adobe system doesn't know about barcodes or PINs, or even libraries. It doesn't communicate with the library's ILS. So that's the gap SimplyE has to cross: bridging the user account in the library ILS with the user activity in Adobe ACS.

Major Elements of the Loans Using Adobe DRM

From Adobe's perspective, opening an ACS DRM title has two steps (in a *very* general sense):

1. Acquire an ACSM file
2. Fulfill the ACSM file with an Adobe ID

At that point you have three things: an actual (encrypted) binary copy of the book, an ACSM file which acts as a 'padlock' on the file, and an Adobe ID which acts as the user's 'key'. So where do these things come from?

The ACSM File

Let's start with the ACSM file. From the SimplyE perspective, ACSM files come from ebook vendors. Overdrive, Axis 360, Bibliotheca, and other all use Adobe ACS and provide an ACSM file in response to a borrow request under specific circumstances. To get an ACSM file you have to:

a. convince the vendor that you are a bona fide patron of a certain library,
b. request a loan for a specific book licensed to that library,
c. ask for an ACSM file

While the user initiates a loan/borrow request in the SimplyE app, the Circulation Manager takes care of communicating with the vendor related to all three steps above. It asks the library ILS whether your barcode/pin are valid, it negotiates with the vendor the authority to act on behalf of the specific patron, it takes out the loan and it asks for the ACSM file.

So, this process provides the padlock, the ACSM file. But we've not yet mentioned the key to decrypt the bookbytes--the Adobe Account ID.

And this is where it gets complicated!

The Adobe ID/User Key

Understanding Adobe IDs can be confusing because there are different types of Adobe ID one might talk about in various circumstances. The oldest is the Adobe Account ID that is associated with a specific Adobe user account, where a user is directed to the Adobe website to register. You may remember that as an awkward bit of user education you had to provide in early ebook reading apps. It led to a desire by libraries *and* vendors to simplify the loan /purchase/reading process.

A second type of Adobe ID is related to enterprise/organizational use where a specific user is not identified. Again, this was developed as a way to simplify use of DRM-protected content without users having to register a third-party account.

Most recently, Adobe developed a product/toolkit (Adobe Vendor ID) for use by ebook providers so that the management of users is offloaded to the ebook provider--whether an ebook service like OverDrive or Bibliotheca, or a retailer like Barnes and Noble. Through a registration process, Adobe develops a "trust" in these third-party vendors or organizations to produce, or mint, user identifiers. This essentially hides the creation and management of user identifiers from the user experience, making it more seamless. User IDs created by the vendor/provider are known are still Adobe Account IDs, but ones produced by the provider.

In the Adobe Vendor ID product, each trusted party is assigned its own vendor ID. For the SimplyE project, NYPL has registered as a trusted party, and our Adobe Vendor ID is "NYPL". NYPL pays about $10k a year for this privilege -- basically Adobe keeping our name in a row of a database somewhere.

Therefore, NYPL can mint identifiers, which we refer to as "keys" in this discussion, for its patrons. But our library is not the only one who needs user keys. Every library in the SimplyE system needs to have keys minted. We can't ask every library in the system to pay $10k a year for the ability to mint keys. Fortunately, we don't have to. An Adobe Vendor ID covers everyone who uses a certain mobile app or service. So all the libraries who use SimplyE are covered. NYPL is allowed to mint keys for patrons of every library in the system.

In the remainder of this document, to reduce confusion, whenever I refer to ***Adobe ID*** or ***Adobe Account ID***, I am meaning *only the user keys we produce in the SimplyE system*. These IDs are distinct from any IDs produced by any other vendor (such as OverDrive for use in their own apps) and are not known by any third party.

An Adobe Account ID is a common UUID (Universally Unique Identifier) and has this format: `urn:uuid:039a0c7a-a51e-11e6-93ab-ebd28876c900`. With a 'key' like this assigned to your SimplyE app/reading device, you can open any 'padlock' (ASCM file, remember) that was not previously opened with a different key. Adobe doesn't know who you are and it doesn't know how you got that ACSM file. All it knows is which keys have been used to open which padlocks.

While NYPL's use of Vendor ID is a boon for all the participants in SimplyE, it does pose a problem in the loan process: how is NYPL supposed to know whether a patron's barcode and PIN are valid? For NYPL patrons, we can check NYPL's ILS. But we can't connect to every ILS in the system. And we don't even want to _handle_ the barcodes and PINs of patrons of other libraries. That would be a security nightmare.

The SimplyE Short Client Token

This is where the Short Client Token comes in. If you go to a library's Circulation Manager URL with a path of `/patrons/me` (e.g., for NYPL: https://circulation.librarysimplified.org/NYNYPL/patrons/me), you'll be prompted with a login dialog. Enter the barcode and PIN for a patron and you'll receive a JSON document that includes several elments. Beside the NYPL Adobe Vendor ID, you'll also see a Short Client Token element in this format: `"drm:clientToken": "KLBRA|1547836658|a77d4156-0434-11e9-8c35-0a8b31d0b954|Xs5ObZk64;;0SKM5kvSW0kswT53lSYn0WKRK5Hr60mr@"` (again, this is just a sample, not real). This string is a cryptographically signed assertion that means the following:

> "Hi, I'm a public library, you might know me as `KLBRA`. I'd like to tell you about one of my patrons. I won't give their real name, but you can call them `a77d4156-0434-11e9-8c35-0a8b31d0b954`. Anyway, at 6:37PM UTC on January 18 2019 (`1547836658`), this patron successfully authorized with my ILS. Bye!"

The `Xs5ObZk64;;0SKM5kvSW0kswT53lSYn0WKRK5Hr60mr@` bit is a cryptographic signature that proves that this Short Client Token actually came from the library `KLBRA`.

The Loan Process

With this in place, we can tell the whole story.

- A patron logs in to their library (in the Circulation Manager) via SimplyE
- The Circulation Manager returns the library's Short Client Token to SimplyE
- SimplyE sends this Short Client Token to Adobe along with the Adobe Vendor ID associated with NYPL, `NYPL`
- Adobe dispatches the Short Client Token to a server NYPL controls, the Library Simplified Library Registry (https://libraryregistry.librarysimplified.org/), and asks that server either to find an existing key (an Adobe Account ID) or to mint a new one if necessary
- NYPL's server validates the Short Client Token it receives

At this point we know that a certain patron of the library `KLBRA` recently validated their credentials against that library's ILS. That's good enough for us, so we either create a new Adobe Account ID, or we look up the one we created earlier, and send it to Adobe.

- Adobe trusts whatever we say, so it just passes the Adobe Account ID back to the client, SimplyE

Now the patron has a 'key', and that's the most difficult part. Next they just need to borrow a book as mentioned above. That gives them the 'padlock'--the ACSM file. Once they have both, they can download, open, and read an ebook encrypted with Adobe's ACS DRM.

So there's one little mystery remaining: how does the NYPL Library Registry server know that the Short Client Token is valid? How does it know that there's a real library called `KLBRA`? What does `KLBRA` even mean? What if someone is just making fake Short Client Tokens?

Note that the server that handles this stuff is the Library Registry (https://libraryregistry.librarysimplified.org), which keeps the canonical list of every library in the SimplyE system. Each new library is added to that list through a registration process on the Circulation Manager hosting the library collections. As part of that process, the Library Registry creates a short name (e.g. `KLBRA`) and a shared secret (e.g. `f05226dcb6679c48bc85e2b64e0ede9d`) for the new library. Both are generated randomly. Both values are passed back to the Circulation Manager and stored in its database.

How do we verify that the Short Client Token is real in the loan request? When the Circulation Manager creates the Short Client Token, it puts the library's short name at the front, and it uses the shared secret *to create the cryptographic signature*, which it puts at the end. If someone made up a fake Short Client Token, NYPL would know, because Library Registry would fail to find a library with that short name (if it was made-up) or it would fail to match the token because the source of the fake wouldn't have access to the shared secret to create the token.

This is a complicated system but it has some really big advantages:

1. There's one 10k cost for the entire SimplyE system, not 10k per library

2. A single Adobe ID can be used to open books distributed by any vendor
3. Neither Adobe nor NYPL ever sees a library patron's credentials

So, that's the most common case. to summarize:

- Your library Circulation Manager turns your credentials into a Short Client Token
- SimplyE passes your Short Client Token to Adobe, who passes it to NYPL, who issues/looks up your Adobe Account ID ("Adobe ID") and passes it back to Abobe, who passes it back to SimplyE
- You get an ACSM file by taking out a loan through your library's Circulation Manager
- SimplyE combines the ACSM file with the Adobe Account ID to download and decrypt an encrypted ebook

Back to Paul's Question

So, now we can talk about Paul's topics:

- activation limits
- not being able to read a title in simplye that is checked out on the vendor platform
- contract renewal schedule
- contract limitations

Activation Limits

Every time SimplyE passes a Short Client Token to Adobe and gets an Adobe Account ID back, Adobe treats this as an "activation". Adobe enforces a limit of six (6) activations per Adobe Account ID, so we want to minimize the number of times this happens.

We can "deactivate" to get one of those activations back, and that's what we do when you log out of a library -- we deactivate your Adobe ID. But there have been bugs in SimplyE that led to duplicate activations. All the ones we know of have been fixed, but there might be more. Not only bugs are the problem, though. More harmfully, there are actions the patron can take, such as uninstalling SimplyE without logging out. These burn an activation, which we can't prevent them. (You can't run cleanup code when an app is uninstalled, presumably to stop apps from saying "if you uninstall me your phone will explode!")

Once all six activation "slots" have been used, the next time SimplyE passes a Short Client Token for the user to Adobe, Adobe *will not* check it with NYPL. Instead, Adobe will say "you used up all your slots, too bad." Now the patron is logged in to their library, but they don't have an Adobe Account ID. They can get padlocks, but they don't have a key to use.

Our last-ditch solution to this is the "Adobe ID reset" functionality in the Circulation Manager's admin interface. Here's how that works. For every patron there are 3 different identifiers:

1. the patron's actual barcode, e.g. `2333312345`
2. the alias used by the library's Circulation Manager to identify this patron to NYPL's library registry, e.g. `a77d4156-0434-11e9-8c35-0a8b31d0b954`, as seen in the Short Client Token example above
3. the Adobe Account ID associated with that alias, e.g. `urn:uuid:039a0c7a-a51e-11e6-93ab-ebd28876c900` as seen above

The Circulation Manager keeps track of the link between 1 and 2, and the Library Registry keeps track of the link between 2 and 3. The second item, the alias, is the connection between the Circulation Manager and the Library Registry. The "Adobe ID reset" script breaks the link between 1 and 2. The Circulation Manager then makes up a new alias for this patron, let's say `07bb4156-0544-17e9-8a35-0a8118d0b933`.

The next time the patron logs in:

1. SimplyE finds a Short Client Token based on the new alias
2. It sends this SCT to Adobe, who sends it to NYPL's Library Registry
3. The Library Registry says: `07bb4156-0544-17e9-8a35-0a8118d0b933`? never heard of them; must be a new patron over at `KLBRA`
4. The Registry issues a brand new Adobe Account ID

The old Adobe Account ID is still in the library registry's database, but it will never be used again. Now the patron has a working key, and can open padlocks again--but with one *BIG* caveat, which leads us to the next topic.

Not Being Able to Read Titles in SimplyE Checked Out from Vendor App

Let's go back to the padlock. Unlike a physical padlock, an ACSM file can be opened with any key whatsoever--any valid Adobe Account ID. However, Adobe keeps track of which key is used to open which padlock. Once you open a padlock with key #1, you can only ever use key #1 to open that padlock. This causes two types of problems for patrons.

Vendor-Specific Adobe Account IDs

Let's use OverDrive as an example. OverDrive has their own Adobe Vendor ID and produces their own Adobe Account IDs for use with their app. I don't know what their Vendor ID is; let's say it's `OVERD`. When you log in to OverDrive's Libby, it does the same thing SimplyE does--it sends your credentials to Adobe, who sends them right back to an Overdrive server (rather than the NYPL Library Registry). OverDrive knows nothing of SimplyE users, though, and that server creates or finds an Adobe Account ID for you. That Adobe Account ID is sent back to Libby. It will be distinct from your Adobe Account ID in SimplyE. I don't know the details, and they certainly don't use Short Client Tokens, but that's how Adobe Account ID negotiation works *in general*.

So, you have a key that was manufactured by `OVERD`. Let's say you use Libby to borrow a book and start reading it. That padlock--the ACSM file corresponding to your particular loan of this book--is now tied to the `OVERD`-based key. But nobody knows this except Adobe. Overdrive doesn't know it, NYPL doesn't know it, SimplyE doesn't know it. Because only Adobe knows that a specific key was used to open a specific ACSM file.

So let's say you open up SimplyE and check your loans. That book you borrowed through Libby will show up in your loans feed, just like an Overdrive book you borrowed through SimplyE. (It shows up because SimplyE is requesting all the items on loan based on the user's library barcode, which is the basis of the loan transaction.) And you'll be able to download the 'padlock'--the ACSM file--through SimplyE. But opening that padlock requires the `OVERD` key, and SimplyE doesn't have it. SimplyE has a different `NYPL` key. The keys were made by different manufacturers and nobody--not even Adobe--knows that the two keys identify the same library patron.

The Brand New Account ID Created by Reset

The other problem that frequently bites patrons isa different angle on the same problem. If your Adobe Account ID is reset, then you have a brand new `NYPL` key. But the loans you've already requested have padlocks that can only be opened with the old `NYPL` key. And you lost access to
that key when the account was reset. And, unless you are Adobe, the only way to know whether a key fits a padlock is to try it.

Contract Schedule & Limitations

The last two items--contract schedule and limitations--i'm going to defer to @Julie Wolf. i will say that one big limitation is that currently we are paying for two different Adobe Vendor IDs, one for SimplyE and one for Open Ebooks
because they're distinct mobile apps
we could save a chunk of money if we only had to pay for one

Questions

Ask me any follow-up quesitons. I know this is a huge chunk.

Deactivating from an Alias

Jonathan Green [12:45 PM]
I have a question about the activations and Adobe IDs. Does the API allow deactivations? So instead of breaking the link and issuing a new alias, and associating that with new Adobe IDs. Is it possible to deactivate a certain Adobe ID from an alias?

Leonard Richardson [12:45 PM]
Sure. To answer that let's introduce a new concept, the "Device ID". When SimplyE sends your Short Client Token to Adobe, it also includes an ID associated with the device hardware (like the Adobe Account ID this a UUID-formatted string: `urn:uuid:3119af54-bd36-b508-b59a-ab16fe0b175b`). We believe the Device ID is generated within the Adobe "black box" running on the mobile device.

So, we now have three pieces of information:

- The "username" and "password" (that's how Adobe sees it, but it's actually a single piece of information, the Short Client Token)
- The Adobe Vendor ID (`NYPL`, so that Adobe knows who to ask about the "username" and "password")
- The Device ID

On Adobe's side, we believe an activation is stored as a 2-tuple (Device ID, Adobe Account ID). So, back to the point, we have a Device ID generated within SimplyE for the current device according to an Adobe algorithm.

Now I can answer your question: we *can* deactivate an Adobe ID, but we need to know both the Device ID and the Adobe Account ID. When you log out, it's easy--we can generate the Device ID for the current device, and we know the Adobe Account ID we were using. So deactivation is easy.

But let's say you try to activate, and Adobe says "you used up all your slots". Theoretically, we could clear out all your old activations to free up slots, but we would have to know the Device IDs to use. But if a device is uninstalled without logging out, we don't know that. That's where we stand at the moment. But, there are two improvements we could make.

1. We're pretty sure that if you register (DeviceID1, AccountID1) and then register (DeviceID1, AccountID1) _again_, Adobe counts it as two activations.
   Is this a bug? I'd say so, but maybe Adobe feels differently. Since most people just use one device, if we ran out of slots we could theoretically try unregistering the current device ID and the current Adobe Account ID over and over again, until the unregistering combination failed.
2. We actually keep track of the Adobe Device IDs, on the Circulation Manager. You can see this list by going to the library's Circulation Manager URL with a path of `/AdobeAuth/devices` (e.g., for NYPL: https://circulation.librarysimplified.org/NYNYPL/AdobeAuth/devices).
   Whenever SimplyE registers a Device ID with Adobe, it *also* registers that Device ID with the Circulation Manager.

So, again, theoretically, if we run into the 'you used up all the slots' problem, we can grab that list and deactivate *all* the known Device IDs activated for the user, increasing the chance that we'll free up a slot.

At this point, the problem is practical. we just don't have the mobile developer resources necessary to do this work. Something else is always more important. Relatively speaking, we're heavy on server-side developer resources, which is why we addressed this problem with a server interface for resetting Adobe Account ID rather than a client-side fallback mechanism that would hopefully make it unnecessary.

Last, we have noticed that the 'no more slots' problem now primarily happens on Android, so we think there are some unknown bugs causing duplicate registrations. The work @Mark Raynsford is doing will hopefully get rid of those bugs as a side effect, and then this should become a lot less of a problem.