Who Needs the UDID?

Demystifying Conversion Tracking in the Mobile World

Solving the Ad Tracking and Attribution Problem in the Mobile World

So lets start with the issues..

There are numerous problems that still need to be solved in the mobile advertising world. These include patchy cookie support (and understanding!), inconsistency of device identifiers (or as they're otherwise known – UDID's), and lack of ad tracking information pass through within the Apple App store, to name just a few.

The good news?

There are solutions on the market *right now* that allow advertisers and agencies to very precisely track and attribute downloads, conversions and even in-app events such as frequent use, purchases, game level completion and much more.

StrikeAd is one such solutions provider and the first truly and 100% Mobile focused Demand Side Platform (or DSP) to enter the market. A few others have emerged lately, but StrikeAd "was here first" and with the pace of technology change in the industry, a year's lead can mean a lot.

Not only is privacy safe app download tracking and attribution a reality right now, the information is also usable by media buyers to plug into automated programmatic buying solutions such as a DSP – this is what we've done at StrikeAd and it works very well.

Read on to learn more about the problems and how they are overcome.

The Goal and the Benefits of Tracking

By installing our SDK in the app, one can use the app download tracking data in a DSP to automatically adjust the buying strategy. So the DSP can buy more of the traffic that has the same profile as the traffic that delivered downloads, frequent users and those that purchase within the app on a regular basis.

The DSP effectively finds a "pattern" of the type of traffic that is most likely to result in a download. It does this by looking at many variables, such as device types, versions, location, time of day, day of week, the site/app the ads are shown on and much more.

When there are enough statistics for the events that the advertiser wants to buy the DSP will find the same recurring set of variable values and make a note of them. The event could be a download and install but could also be a subscription, game level completion, etc.

It will then make more bids for such traffic and buy more impressions with such a profile – all on an impression-by-impression basis. And we're talking 10's of thousands of impressions per second here. That's 10's of billions of mobile impressions a month and is what we process on our servers at StrikeAd.

A typical traffic "profile" may look something like this*: iPhone or Samsung Galaxy S, Sunday and Saturday lunch time, on sites from publisher XYZ and apps from game maker ABC* and so on. At StrikeAd we actually use about 30 different variables to determine this pattern.

Think of this technique as similar to using a number of criteria to distinguish between different people, such as personality, hobbies, height, weight, eye colour, hair colour and so on, as used on a dating site. The more parameters you use, the more precisely you can identify and differentiate people.

The result of this targeting is a more targeted ad buy. This has far better performance than that of a blind ad buy that may be made in bulk and not on an impression by impression basis.

So in summary, with privacy safe attribution and advanced targeting, the advertiser knows exactly who drove the more effective traffic and can adjust the spend in the right direction.

UDIDs – Set up to fail from the Start?

That's all well and good you say, but how does a DSP – or anybody for that matter – track and attribute impressions and clicks to a download or a subscription if there is "dead space" between the "entrance" and "exit" of the mobile app advertising workflow?

The answer is not a single solution or a magic bullet, but a combination of approaches and technical solutions, which together deliver the required result.

What's the situation with UDIDs today?

Currently, when a DSP or an ad network buys a mobile banner that runs on a mobile media exchange (a Supply Side Platform – aka 'SSP'), the latter's code in the app passes the UDID back, often one way encrypted (aka hashed).

So many ad networks quickly adopted a "match the ID" approach where they would record the UDID of the click and ask the advertiser to send the UDID of the device which installed the app to their server, where they would check to see if that UDID is in the click records – attributing the download if there was a match.

But there are several problems here right off the bat.

One is that typically, an advertiser will use many different banners to advertise an app and the above approach does not precisely pin point which banner was the one that drove the download. You also can't always argue that the last view or last click won, due to download and app start lag (more on this later on).

Secondly, the exchange may be sending the UDID hashed using one algorithm (e.g. SHA1) and the advertiser may be sending it hashed using another (e.g. MD5) – so there will never be any match. Often communication between all the parties in the chain is ineffective and the details of the hashing are miscommunicated – again, no matches and the media buyer is deemed as "ineffective". A bit like trying to match a phone number and a postcode. They're just never going to match, even if they're for exactly the same house.

Thirdly, the Android OS has several IDs available to app developers – AndroidID, IMEI, MEID or ESN. Exchanges and advertisers often will use and send different ones.

Combine that with different hashing algorithms and the matching nightmare gets worse.

Lastly, and importantly, the Apple iOS UDID, that's been the ID for iPhone, iPad and IPod Apple, is deprecating (retiring). So the UDID will no longer be available within apps to be pulled out by the SSP SDK – so this free ride will end soon. There will be other ID's one can use on the device, such as the Wifi MAC address (the WIFI chip inside the iPhone unique ID). However, the latter is not ideal as it may not get recycled and can become unavailable if WIFI is off.

So there is often a lack of clarity when setting up a campaign between the advertisers, agency, media buyer and the exchange and, again, the wrong IDs are often compared and never match. Many attributions are missed and don't go back into the mix – driving up cost of the download and reducing their number too.

If this is not enough to convince you, the last and worst of the problems with "match the UDID" approach is that only media that is running inside an app can be bought if UDIDs are to be used for download attribution i.e. the advertiser can't buy media seen on mobile sites in a mobile web browser.

This is because a UDID can only be obtained when you have access to the operating system API, i.e. you are some code running inside an app. If you're a simple meat and potatoes HTML page or even a fancy one with some JavaScript, you just cannot get the UDID of the device you're being shown on.

This is a problem because there is a huge amount of very popular mobile-optimized (i.e. formatted to read much better on a small screen) sites which could be driving customers to the app store to download that app. The same again as the app traffic in fact.

The net effect is that app inventory becomes more sought after and in turn price, yield and cost of download goes up. This keeps the sellers of app inventory happy – not necessarily a bad thing – but doesn't benefit anybody else.

So on the iPhone, the UDID will soon be a non-starter.

Ditching the UDID

"But if the app stores are the dead end where advertising tracking dies, how do you tie an impression to a download without a device ID?!", you ask.

Surprisingly it is back to the old veteran – the cookie!

Cookies actually work fine on most mobile devices, especially on all the main ones. Importantly it works on the ones which have app stores and apps, such as iPhones, iPads, Android powered ones, BlackBerries, new Nokias etc.

If they didn't, many sites where you have to log in, such as Gmail, eBay, PayPal, Facebook etc. would be pretty hard to use as from page to page the site would not remember who you were.

So a cookie is a bit like that ultraviolet stamp that a bouncer would put on your hand so they know who you are after you pop out for some fresh air (or a cigarette!) and decide to go back in. Or an electronic pass into a building – some will have a few of these for each building they

have access too – much like a cookie.

The only issue that exists with Cookies is on the Apple Operating system, iOS – but only with the setting – and not reading – of third party cookies.

If you're not sure what I mean by first and third party cookies, I simply refer to the domain of the server on which the page you look at is hosted and the domain of the server where the tracking cookie was set from.

For example, let's say you're looking at a page on *amazon.com* and there is a tracking pixel pointing to *strikead.com*, which tries to set a cookie. This strikead.com cookie will be classed as a third party one as it's not from the same domain as the page you're on.

Cookies from amazon.com, however, are first party since they're from the same domain as the page.

Setting a third party cookie in iOS Safari – the iPhone browser – won't work. The attempt to set a cookie will be blocked by Safari. However, reading both first and third party cookies is just fine on iOS Safari on the iPhone, iPad and iPod touch.

So the StrikeAd server sets a first party cookie – just after a click, when redirecting a user from the banner to the App Store or some landing page. During this redirect, a user's browser is going via the strikead.com server and is a "first party" and can therefore set a cookie.

OK so I can't set a third party cookie but first party cookies are OK and that's all well and good, you say, but how do I read that cookie from inside the app?

The short answer is…you can't, since Safari is just another app and apps on iOS cannot share data for security reasons.

Getting to Cookies from an iPhone App

The longer answer is….you can - but there are a few "workarounds" to be done.

The problem with reading cookies from inside an app is that they are set in the device's browser, and only the browser has access to them. You can, however, launch one app from another. For example, you could launch the device's web browser, such as Safari, from inside the downloaded app.

This is what the StrikeAd attribution SDK does – it asks Safari to go to the strikead.com server – the same page that the user was on when the cookie was set. Once the cookie and the necessary information from it is read, the server tells the browser to go back into the app where it came from. It also passes a variable back, which allows StrikeAd to sync the web view with the app download – thus attributing the conversion to a specific impression and click.

This all may seem very complicated but it happens quickly and is barely noticeable, and importantly, works very well. We believe our approach is the most effective solution available today.

The only slight negative impact of this process is a brief "flash" of the browser window whilst

you are in the app (the '**Browser Flash**'). For the reasons outlined earlier the StrikeAd SDK does this faster than anyone else and only for a very brief moment – a few hundred milliseconds – blink and you definitely will miss it.

Many advertisers are already using this method and have found that it does not affect user experience and only gives them much better insight into the app life.

Furthermore, the StrikeAd SDK only does the Browser Flash once – on first launch. This is because, after this first Flash, the unique ID from the cookie that tells us which click drove the download is passed to the server in the background and completely seamlessly to the user.

The Open Door in the Android Market

Good old Google – being deeply steeped in advertising they understand that certain things need to happen for the advertising machine to keep turning its wheels and provide free app developers with a source of income.

So, the Android Market actually has a mechanism, which allows variables to be passed to it from a click on a banner, and from there – to be passed to the app.

The app can then pass this information back to the media buyer for attribution and optimization.

So for those advertisers who really cannot have a pop up in their app, at least on Android, there is a way.

The only issue with this approach is that it makes "post view" conversion attribution impossible, as there is no cookie to be read during download, to attribute it to a click or impression that occurred earlier.

What Happens Inside an App after download?

As well as tracking downloads, advertisers also want to know a bit more about what happened after this event.

Just think – if a user goes to a website, do the developers of that website want to know which parts of the site the user uses most and how? Of course they do! This way they can work out which parts of the web site are most useful, make them better, add to them and so on.

Same with the app developers, except that often this information is not collected and those installed apps out there are a black box. App developers, however, want to know which parts of the app are most used and make them better.

The solution is to use the same code inside the app that is used for tracking the download, to also record various events such as registrations, game level play starts, completions, etc.

This ability is a reality and a solution now and the StrikeAd App Tracking SDK does all this and more. StrikeAd app reporting solution can even pass data into third party analytics packages such as Omniture so it can be examined and analysed there, together with other tracking data, allowing all reporting to be viewed in one place.

As well as collecting and reporting this data, the other challenge is collecting huge volumes of it and aggregating it into reports, which can be viewed in many different combinations. For example, you could pull out a report that would tell you "how many iPhone users in London during lunch time on Sunday completed a level" or "how many new issues of the magazine were purchased in the app in New York during a week" and so on. All in real time. Imagine the data that needs to be processed for this bearing in mind that Angry birds, for example, have around 500M users globally (downloads on all platforms).

This sort of data is pretty standard on the web, but in the app world it's actually quite new and missing in many cases.

The Download Lag

The other thing to bear in mind is that downloads are just like other conversion events and can happen sometimes days after the click.

Picture this:

> *You see a banner for some exciting new app.*
> *You click on it and this event is recorded.*
> *You end up in the app store, download the app and then turn your phone off as you get on a plane before you get a chance to launch the app.*
> *You then forget about the app.*
> *You get off the plane, go about your business, days pass.*
> *Finally – days later – you see the app and launch it.*

Since many current solutions on the market will not try and sift through days of UDID data – as there could be billions of records to process – the above situation will not be accounted for and whichever media buyer ran the ad will not be attributed to having driven the download.

On first look this is good for the agency and advertiser, since it means fewer downloads you have to pay for. However, if you look deeper, all it means is that the statistics are incomplete. Potentially productive inventory is not being re-targeted, there are fewer downloads than there could be in the long run, and CPD stays relatively high.

This is due to the long lag between the click on the banner and the launch of the app.

Don't forget, the app only "wakes up" for the first time and starts to do anything when it is first launched after the download to the device. Until you've started it at least once, it is asleep on your phone, like a new Sky Box you just bought that's still in its cardboard box that has not been unwrapped or plugged in.

Until you've launched it – nobody else knows you have it.

With a cookie approach it doesn't matter how many days the app lays dormant, as the information is kept in the cookie – on the users' device – and you can attribute downloads which may have happened days ago extremely easily.

There is no need to sift through data on the servers, trying to match up UDID's, since the cookie and the tracking data is stored on the device.

The "Post View" Download

Post view conversions are an extremely common approach in online advertising and it has been around for years. It's a process of attributing a conversion – purchase, form fill, etc – to a banner view, rather than a click. In mobile, however, this is not very common still, even in mobile web campaigns.

In fact, for the mobile app industry it is a quantum leap in the future...

With an app, the post view experience would be something like:

> *You see an ad for an app a few times (and a cookie is set to record this)*
> *You never click on the banner but you decide to have a look at the app later on*
> *You go to the app store, find the app, read the overview, download it*
> *The app starts, flashes up the browser*
> *The cookie from step one is read and the attribution occurs*

The trouble is, even though the above solution is technically very feasible right now, many advertisers and agencies will not consider such a download attribution process currently.

When downloads are sold and bought on a cost per download basis, it is in the interest of the person fitting the bill to pay for as few as possible whilst getting the supplier to generate as many as possible, arguing that they did not *truly* do so.

However, it is actually in the advertisers' direct interest to take those post view conversions into account, since the media buyer would then be able to optimize to them.

The result for the advertiser would be **lower** CPA/CPD and more downloads – all very worthwhile benefits! In turn it would allow for lower CPA/CPD to be commended in the market, perhaps $0.20 - $0.30 cents instead of the current $1.00-$2.00.

Adoption from all Parties

There are many solutions out there to make mobile app advertising more successful and cost effective, but it will take the triumvirate of the Advertiser, Agency and Media to adopt them.

Without all three parties understanding the options and deciding to utilise them, nothing will happen – or at least not easily and not quickly.

StrikeAd stands ready to work with all the partners to make this a reality now and will continue to try and innovate with the industry.

With the 2012 GSMA Mobile World Congress upon us, it's time we kick start mobile advertising into the next phase!