ENGINEERED BY
IPONWEB

BidSwitch Protocol for Suppliers

Release 1.0

Oct 15, 2019

VERSION INFORMATION

This guide covers the latest version of the BidSwitch Supplier Protocol. It is based on the latest OpenRTB Protocol Specification V2.5[1], and helps Suppliers to integrate with BidSwitch, and thus access all of its connected Buyers.

For Suppliers who wish to integrate with BidSwitch, if you are fully compatible with this spec, you can be integrated much more quickly than waiting for a bespoke integration to be developed.

[1] http://www.iab.com/wp-content/uploads/2016/03/OpenRTB-API-Specification-Version-2-5-FINAL.pdf

CHAPTER 1

PROTOCOL RELEASE NOTES

1.1 v1.0 - June 2017

Initial Release

CHAPTER 2

SENSITIVE CATEGORIES AND RICH MEDIA

For greater coverage of sensitive categories BidSwitch extends the standard IAB list with additional categories. These categories may be used in the following fields.

Table 1: Sensitive Categories Fields

Bid Requests	Bid Response
• `bcat` • `site.cat` • `app.cat`	`seatbid.bid.cat`

Table 2: BidSwitch Sensitive Categories

ID	Category
BSW1	Alcohol
BSW2	Gambling
BSW3	Tobacco and smoking
BSW4	Firearms and weapons
BSW5	Sexual & Reproductive Health
BSW6	Ringtones & Downloadable
BSW7	Drugs & Supplements
BSW8	Get Rich Quick
BSW9	Free Gifts, Quizzes, & Surveys
BSW10	Nudity
BSW11	Cosmetic Procedures & Body Modification

2.1 Supported Rich Media Frameworks

Table 3: Supported Rich Media Frameworks

Description	Value
Any framework is accepted	ALL
Adcentric	ac
Adinterax	ad
Adform	af
Atlas	at
Apivid	av
DoubleClick	dc
Eyeblaster	eb
EyeReturn	er
EyeWonder	ew
Flashtalking	ft
Klipmart	km
Kpsule	ks
MediaMind	mm
Mediaplex	mp
Piximedia	pm
PointRoll	pr
Pictela	pt
Rockabox	rb
Smart Adserver	sa
Silence Media	sm
Unicast	ui
Undertone	ut
Viewpoint	vp
Weborama	wo

CHAPTER 3

BACKWARD COMPATIBILITY

- BidSwitch Supplier Spec 1.0[2]

[2] https://docs.bidswitch.com/_downloads/BidSwitch_supplier_spec_1.0.pdf

CHAPTER 4

DATA COMPRESSION

BidSwitch supports JSON and Compressed JSON (gzip) as data formats for bid requests and bid responses. It is recommended to use Compressed JSON to minimize the amount of data exchanged between you and BidSwitch as this reduces latency times between servers, and traffic listening costs for all parties involved. Here is what BidSwitch has observed with gzip traffic:

- When enabled, gzip compression ratios tend to be in the range of x1.4 to x2.0 (depending on the size of the request/response)
- There is no tangible extra CPU load due to compressing/decompressing traffic, i.e. there is no extra cost

You can enable this on your integration at anytime as no changes are needed on the BidSwitch side.

In order to set up Compressed JSON bid requests and and accept compressed responses, use the following instructions:

1. Compress your HTTP request body with `gzip`
2. Add a `Content-Encoding: gzip` HTTP header to indicate that the request is compressed
3. Add an `Accept-Encoding: gzip` header to indicate that you can accept a compressed response
4. Please contact BidSwitch support at support@bidswitch.com to ensure all parties are aware of a change in the integration

Once set up, BidSwitch starts accepting bid requests in Compressed JSON and responds using compressed JSON. All such bid responses carry a `Content-Encoding: gzip` HTTP header indicating that the response is compressed. It is recommended that you check for this HTTP header to distinguish between JSON and Compressed JSON bid response formats, if it is present decompress the HTTP body before further processing.

Note: BidSwitch may still respond with non-compressed content when it detects that the size of a gzip compressed response would exceed the size of an uncompressed response. This may happen

for very short bid responses. In this case the `Content-Encoding: gzip` header is absent.

CHAPTER 5

MACROS

5.1 Win price macro

In order for the exchange to convey certain information to the winning bidder (e.g., the settlement price), some substitution macros can be inserted into the win notice URL. Prior to calling a win notice URL, BidSwitch will search the specified URL for any of the defined macros and replace them with the appropriate data.

Note: The substitution is simple in the sense that wherever a legal macro is found, it will be replaced without regard to syntax correctness.

Table 1: Win Price Macro Description

Value	Description
${AUCTION_PRICE}	Settlement price for the auction. The substituted value will be defined in CPM. This macro should be used in the `burl` field, see the *Response Bid Object* (page 66) section for usage details. As well as the `burl` field, its use is also supported in these fields: • For Native Responses: `seatbid.bid.burl`, `seatbid.bid.adm_native.jstracker` or `seatbid.bid.adm_native.imptracker` • For non-native Responses: `seatbid.bid.burl` or `seatbid.bid.adm`

5.1.1 Encrypting the Win Price

BidSwitch can process encrypted win price macros, and uses the same method as Google[3]. To enable this for your integration, contact BidSwitch support to receive the **integrity key** and **encryption key** necessary for doing this. Both of these keys will be sent to you in web-safe base64 strings and they should be web-safe decoded and then base64 decoded by your application.

```
skU7Ax_NL5pPAFyKdkfZjZz2-VhIN8bjj1rVFOaJ_5o=  // Example Encryption key (e_key)
arO23ykdNqUQ5LEoQ0FVmPkBd7xB5CO89PDZlSjpFxo=  // Example Integrity key (i_key)
```

Encryption scheme

The price is encrypted using a custom encryption scheme that is designed to minimize size overhead while ensuring adequate security. The encryption scheme uses a keyed HMAC algorithm to generate a secret pad based on the unique impression event ID. The encrypted price has a fixed length of 28 bytes, comprised of:

- `16-byte` initialization vector
- `8-bytes` of ciphertext
- `4-byte` integrity signature

The encrypted price must be web-safe base64-encoded, according to RFC 3548[4], with padding characters omitted. Thus, the 28-byte encrypted price is encoded as a 38 character web-safe base-64 string irrespective of the winning price paid.

```
# The encrypted format is:
{initialization_vector (16 bytes)}{encrypted_price (8 bytes)}{integrity (4 bytes)}

# Example encrypted prices:
WEp8wQAAAABnFd5EkB2k1wJeFcAj-Z_JVOeGzA  # 100 CPI micros
WEp8sQAAAACwF6CtLJrXSRFBM8UiTTIyngN-og  # 1900 CPI micros
WEp8nQAAAAADG-y45xxIC1tMWuTjzmDW6HtroQ  # 2700 CPI micros
```

The price needs to be encrypted as `<price xor HMAC(encryption_key, initialization_vector)>` so decryption calculates `HMAC(encryption_key, initialization_vector)` and xor's with the encrypted price to reverse the encryption. The integrity stage takes 4 bytes of `<HMAC (integrity_key, price||initialization_vector)>` where `||` is concatenation.

The following example code outlines this, with the following definitions:

- `iv` initialization vector (16 bytes - unique to the impression)
- `e_key` encryption key (32 bytes - provided by BidSwitch Support)
- `i_key` integrity key (32 bytes - provided by BidSwitch Support)
- `price` (8 bytes - in micros of account currency)

[3] https://developers.google.com/ad-exchange/rtb/response-guide/decrypt-price
[4] https://tools.ietf.org/html/rfc3548

- `hmac(k, d)` SHA-1 HMAC of data `d`, using key `k`
- `a || b` string `a` concatenated with string `b`

```
// example code
pad = hmac(e_key, iv)  // first 8 bytes
enc_price = pad <xor> price
signature = hmac(i_key, price || iv)  // first 4 bytes

final_message = WebSafeBase64Encode( iv || enc_price || signature )
```

Once enabled, you can return the encrypted price to BidSwitch who can then return it to the winning Buyer in the appropriate manner.

- You need to encrypt the value using SHA-1 HMAC. You can do this using a crypto library that supports SHA-1 HMAC, such as Openssl
- The encrypted value should be then encoded using web-safe BASE64

```
# <!-- Example billing URL (burl) which will have macro substituted -->
https://adserver.com/winnotice?impid=102&winprice=${AUCTION_PRICE}

# <!-- Example billing URL (e.g. burl) -->
https://adserver.com/winnotice?impid=102&winprice=1.34

# <!-- Example encrypted billing URL (e.g burl) -->
https://adserver.com/winnotice?impid=102&winprice=WEp8nQAAAAADG-y45xxIC1tMWuTjzmDW6HtroQ
```

5.2 Supplier Click Tracking URL Macro

If you require Buyers to include a click tracking macro in their creatives, Buyers will return this macro in the `adm` field. You should replace this macro with your Supplier click tracking URL. If you do not support click tracking macros and a Buyer replies with one in the `adm` field, BidSwitch will replace it with an empty string.

Note: In the case of video and native inventory, click tracking is handled on the Supplier side and the click macro is not used, nor is the `adm` field.

Table 2: Click Tracking Macro

Value	Type
${CLICK_URL:URLENCODE}	A placeholder for the Supplier click tracking URL in URL encoded form. Required for bids to Suppliers that support click tracking. No more than one click tracking macro can be used in the `bid.adm` field. Only single-encoded click tracking URLs are supported.

If requested, the Supplier click URL should be inserted before the landing page in the creative. The

landing page URL should be single-escaped. For example, if the Buyer click-URL contains:

```
http://dsp.com/click?bc=dnJD723&sspclick=${CLICK_URL:URLENCODE}
```

The macro is replaced by the Supplier and the user clicks the resulting URL

```
http://dsp.com/click?bc=dnJD723&sspclick=http%3A%2F%2Fssp.com%2Fclick%3Fic%3DbKk4%26lp%3D
```

The Buyer unescapes the `sspclick` parameter and redirects to the target URL while adding the landing page at the end

```
http://ssp.com/click?ic=bKks3k4&lp=http%3A%2F%2Fadvertiser.com%2Fhomepage
```

Note: Some Suppliers may keep the `/` and `:` characters unencoded, thus the resulting click URL may take a form such as the following

```
http://dsp.com/click?bc=dnJD723&sspclick=http://ssp.com%2Fclick%3Fic%3DbKk4%26lp%3D
```

CHAPTER 6

SUPPLIER BID REQUEST

This is the top level object that is sent by the Supplier to BidSwitch. Each bid request sent should contain the following fields.

Note:

- Fields marked with an asterisk (*) are optional.
- While individually none of the following fields are required, one of them is required to be in each bid request: `banner`, `video`, `audio`, or `native`.
- While individually neither of the following fields is required, one of them must be in the request: `site`, `app`.

Table 1: Bid Request Object Properties

Value	Type	Description
id	*string*	Unique ID of the bid request, for example, `"b5ba5ed2-547e-4e86-8a84-34a440dad6db"`
imp	*array of objects*	Array of objects representing the impressions offered, for more information, see the *Impression Object* (page 14) section.
device	*object*	Device object with details about the device to which the impression will be delivered, for more information, see the *Device Object Properties* (page 38) section.
*user**	*object*	User Object which describes the user, for more information, see the *User Object* (page 42) section.
*tmax**	*integer*	Maximum time in milliseconds the exchange allows for bids to be received to avoid timeout, including internet latency, for example, `120`.

Table 2: Bid Request Object Properties

Value	Type	Description
*cur**	*array of strings*	Array of allowed currencies for bids on this bid request using ISO-4217[5] alpha codes, for example, `["USD", "EUR"]`. The default is `["USD"]`.
*at**	*integer*	Auction type, the default value is `2`. • `1`: the first price auction. • `2`: the second price auction.
*source**	*object*	Indicates the entity responsible for the final impression sale decision.
*site**	*object*	The *Site Object* (page 50) describing the site. Either `Site` or `App` must be present.
*app**	*object*	The *App Object* (page 52) describing the mobile application. Either `Site` or `App` must be present.
*bcat**	*array of strings*	Blocked Advertiser Categories, using the IAB taxonomy, and extended with additional sensitive categories listed in the *Sensitive Categories and Rich Media* (page 3) section. Creatives belonging to at least one of the listed categories are not permitted for bidding in the current bid request, for example `["IAB10-1", "IAB25", "BSW3"]`
*badv**	*array of strings*	Array of strings of blocked top-level domains of advertisers, for example, `["mysite.com", "mysite2.com"]`
*wseat**	*array of strings*	An array of Buyer seats allowed to bid on this auction, for example `[58, 61, 99]`. If this field is present, the specified seat IDs may be supplied using BidSwitch or Supplier taxonomy. A bid request may contain multiple seat IDs using the Supplier taxonomy.
*allimps**	*integer*	A flag to indicate if the Supplier can verify that the impressions offered represent all of the impressions available in context (e.g., all on the web page, all video spots such as pre/mid/post roll) to support road-blocking. • `0` = no or unknown • `1` = yes, the impressions offered represent all that are available.
*regs**	*object*	A regulations object that specifies any industry, legal, or governmental regulations in force for this request, for more information, see the *Regulation Object* (page 53) section.
*ext**	*object*	Ext Object used for Supplier specific properties, for more information, see the *Ext Object* (page 44) section.

[5] https://www.iso.org/iso-4217-currency-codes.html

6.1 Impression Object

Note: Fields marked with asterisk (*) are optional.

Table 3: Impression Object Properties

Value	Type	Description
id	*string*	ID of the impression being shown, unique within the bid request, for example `"1"`
*banner**	*object*	The *Banner Object* (page 17) describes the ad properties. Required for banner impressions. One of these objects should be present in the request: `banner`, `video`, `audio`, or `native`.
*video**	*object*	The *Video Object* (page 20) describes the ad properties. Required for video impressions.
*audio**	*object*	The *Audio Object* (page 25) describes the ad properties. Required for audio impressions. One of these objects should be present in the request: `banner`, `video`, `audio`, or `native`.
*native**	*object*	The *Native Object* (page 28) describes the ad properties. Required for native impressions. One of these objects should be present in the request: `banner`, `video`, `audio`, or `native`.
*bidfloor**	*float*	Bid floor in CPM as set by the Supplier, for example, `0.01080`
*bidfloorcur**	*string*	Bid floor currency specified using ISO-4217 alpha codes, the default is, `"USD"`.
*instl**	*integer*	Specifies if the ad is an interstitial. • `0` = not interstitial, the default value. • `1` = the ad is interstitial or full screen
*tagid**	*string*	Identifier for specific ad placement or ad tag that was used to initiate the auction.
*secure**	*integer*	Specifies if the page is SSL compliant: • `0`: for insecure pages, the default value. • `1`: for secure pages. Creative assets for secure pages should be SSL-compliant.
*iframebuster**	*array of strings*	Array of names of supported iframe busters, for example, `["dc", "rb"]`, for more information, see the Supported Rich Media Frameworks[6] section.

[6] https://protocol.bidswitch.com/standards/bidswitch-categories.html#srmf

Table 4: Impression Object Properties

Value	Type	Description
*pmp**	*object*	The *Private Marketplace Object* (page 35), used for direct deals between Buyers and Suppliers.
*displaymanager**	*string*	Name of the ad mediation partner, SDK technology, or native player responsible for rendering the ad (typically video or mobile), for example, `"SOMA"`
*displaymanagerver**	*string*	Version of the ad mediation partner, SDK technology, or native player responsible for rendering the ad (typically video or mobile), for example, `"1.1"`
*metric**	*array of objects*	The object that is associated with an impression as an array of metrics, see the *Metric Object* (page 42) section.
*exp**	*integer*	Impression expiry timeout, in seconds, the default is `"300"`. An impression will be considered expired if it is registered later than `imp.exp` seconds after the auction.
*ext**	*object*	Impression extensiton object, see *Impression Ext* (page 15)

6.1.1 Impression Ext

Table 5: Impression Object Properties

Value	Type	Description
*ssai**	*int*	Indicates if server-side ad insertion (e.g., stitching an ad into an audio or video stream) is in use and the impact of this on asset and tracker retrieval. It can take the following values: • `0` = status unknown • `1` = all client-side (i.e., not server-side) • `2` = assets stitched server-side but tracking pixels fired client-side • `3` = all server-side.

6.1.2 Example Impression Object JSON

```
{
   "imp":[
      {
         "id":"1",
         "metric":[
            {
              "type":"viewability",
              "value":0.85
            }
         ],
         "bidfloor":0.426,
```

(continues on next page)

(continued from previous page)

```
         "banner":{
            "w":300,
            "h":250,
            "pos":1,
            "topframe":0,
            "expdir":[
               1,
               3
            ]
         }
      }
   ]
}
```

6.2 Banner Object

Note: Fields marked with an asterisk (*) are optional.

Table 6: Banner Object Properties

Value	Type	Description
*id**	*string*	Unique identifier for the banner object, for example, `3`. Can be used to tracking multiple banner objects in a companion banner array.
w	integer	Width of the impression in pixels, for example, `300`
h	*integer*	Height of the impression in pixels, for example `250`
*battr**	*array of integers*	Blocked creative attributes as defined in the OpenRTB protocol, for example, `[1, 23]`
*btype**	*array of integers*	Blocked banner ad types as defined in the OpenRTB protocol, for example, `[4, 21]`
*pos**	*integer*	Ad Position as defined in the OpenRTB protocol, for example, `1`
*topframe**	*integer*	Indicates if the banner is in the top frame as opposed to an iframe. • `0` = no • `1` = yes.
*mimes**	*array of strings*	Specifies the content MIME types supported, common MIME types include `"text/html"`, `"application/x-shockwave-flash"`, and `"image/gif"`. For example: `[ "video/mp4", "image/jpg"]`
*expdir**	*array of integers*	Possible expansion directions for an expandable ad, for example, `[2,5]`. This can take the following values: • `1`: Left • `2`: Right • `3`: Up • `4`: Down • `5`: Full screen If the field is not present, expandable creatives are not allowed.

Table 7: Banner Object Properties

Value	Type	Description
*format**	*array or objects*	An array of format objects, see *Format Object* (page 19), denoting the alternative sizes that may be used for bidding. If one of the alternative ad sizes is used in the bid response, then the `seatbid.bid.h` and `seatbid.bid.w` fields are required in the bid response.
*api**	*array of integers*	List of supported API frameworks for this impression as defined in the OpenRTB, for example `[3, 5]`. If an API is not explicitly listed, it is assumed not to be supported.

6.2.1 Banner JSON Example

```
{
   "banner":{
      "id":"abc123",
      "w":300,
      "h":250,
      "pos":1,
      "topframe":0,
      "btype":[
         2,
         3
      ],
      "mimes":[
         "text/html",
         "application/x-shockwave-flash"
      ],
      "format":[
         {
            "h":50,
            "w":300
         }
      ]
   }
}
```

6.3 Format Object

Table 8: Format Object Properties

Value	Type	Description
h	*integer*	Height of the impression in pixels, for example `500`
w	*integer*	Width of the impression in pixels, for example `340`

```
{
   "format":[{
      "w":300,
      "h":250
   }]
}
```

6.4 Video Object

Note: Fields marked with an asterisk (*) are optional.

Table 9: Video Object Properties

Value	Type	Description
mimes	*array of strings*	Content MIME types supported.
minduration	*integer*	Minimum video ad duration in seconds, for example, `2`
maxduration	*integer*	Maximum video ad duration in seconds, for example, `15`
*linearity**	*integer*	Indicates if the impression must be linear or nonlinear, for example, `1`. If none is specified, it is assumed all are allowed • `1`: Linear/In-stream • `2`: Non-Linear/Overlay
*placement**	*integer*	Placement type for the impression, for example `2`. **Note:** Though not required, this is an important field for some Buyers, not explicitly setting it will result in lower demand. This field can take the following values: • `1`: In-stream. Played before, during or after the streaming video content that the consumer has requested (e.g., Pre-roll, Mid-roll, Post-roll). • `2`: In-banner. Exists within a web banner that leverages the banner space to deliver a video experience as opposed to another static or rich media format. The format relies on the existence of display ad inventory on the page for its delivery. • `3`: In-article. Loads and plays dynamically between paragraphs of editorial content; existing as a standalone branded message. • `4`: In-feed. Found in content, social, or product feeds. • `5`: Interstitial/Slider/Floating. Covers the entire or a portion of screen area, but is always on screen while displayed (i.e. cannot be scrolled out of view). Note that a full-screen interstitial (e.g., in mobile) can be distinguished from a floating/slider unit by the `imp.instl` field.
*playbackend**	*integer*	The event that causes playback to end, for example `2`. This field can take the following values: • `1`: On Video Completion or when Terminated by User. • `2`: On Leaving Viewport or when Terminated by User. • `3`: On Leaving Viewport Continues as a Floating/Slider Unit until Video Completion or when Terminated by User.

Table 10: Video Object Properties

Value	Type	Description
protocols	*array of integers*	Accepted video bid response protocols as defined in OpenRTB, for example `[6,8]`. As BidSwitch only serves video using VAST wrappers, the valid response integers are `4`, `5`, `6`, or `8` for the request to be eligible for bidding.
*pos**	*integer*	Ad Position as defined in OpenRTB, for example `1`
*w**	*integer*	Width of the player in pixels, for example, `600`
*h**	*integer*	Height of the player in pixels, for example `400`
*startdelay**	*integer*	Indicates the start delay in seconds. If the start delay value is greater than 0, then the position is mid-roll and the value indicates the start delay. • `> 0`: Mid-Roll (value indicates start delay in second) • `0`: Pre-roll • `-1`: Generic mid-roll • `-2`: Generic post-roll
*battr**	*array of integers*	Blocked creative attributes as defined in OpenRTB, for example, `[6]`
*minbitrate**	*integer*	Minimum bit rate in Kbps, for example `680`
*maxbitrate**	*integer*	Maximum bit rate in Kbps, for example `990`
*api**	*array of integers*	List of supported API frameworks for this impression as defined in OpenRTB, for example, `[1,2]`. If an API is not explicitly listed, it is assumed not to be supported.
*maxextended**	*integer*	Maximum extended video ad duration if extension is allowed. • Blank or `0`, extension is not allowed. • `-1`, extension is allowed, and there is no time limit imposed. • Greater than `0`, then the value represents the number of seconds of extended play supported beyond the `maxduration` value.
*boxingallowed**	*integer*	Indicates if letter-boxing of 4:3 content into a 16:9 window is allowed: • 0 = no • 1 = yes.
*playbackmethod**	*array of integers*	Allowed playback methods as defined in the OpenRTB, for example `[1, 2]`. If none are specified, it is assumed all are allowed.
*delivery**	*array of integers*	Supported delivery methods (e.g., streaming, progressive) as defined in OpenRTB. If none specified, assume all are supported, for example, `[1, 2]`

Table 11: Video Object Properties

Value	Type	Description
*sequence**	*integer*	If multiple ad impressions are offered in the same bid request, the sequence number will allow for the coordinated delivery of multiple creatives, for example, 2.
*companionad**	*object array*	Array of Banner objects if companion ads are available. See the *Banner Object* (page 17) section for more information.
*companiontype**	*array of integers*	List of allowed companion ad types, for example [1, 2] Possible values: • 1: Static Resource • 2: HTML Resource • 3: iframe Resource
*skip**	*integer*	Indicates if the player will allow the video to be skipped, where 0 = no, 1 = yes.

6.4.1 Video Ext Object

Table 12: Video Ext Object Properties

Value	Type	Description
*rewarded**	*integer*	Indicates whether the ad is being rendered as part of a rewarded / incentivised user experience, where: • 0 = non-rewarded • 1 = rewarded • If omitted, non-rewarded can be assumed

6.4.2 Video Object Example

```
{
   "id":"1",
   "bidfloor":0.03,
   "video":{
      "w":640,
      "h":480,
      "pos":1,
      "startdelay":0,
      "minduration":5,
      "maxduration":30,
      "maxextended":30,
      "minbitrate":300,
      "maxbitrate":1500,
      "skip":1,
      "api":[
```

(continues on next page)

(continued from previous page)


```
            1,
            2
        ],
        "protocols":[
            2,
            3
        ],
        "mimes":[
            "video/x-flv",
            "video/mp4",
            "application/x-shockwave-flash",
            "application/javascript"
        ],
        "linearity":1,
        "boxingallowed":1,
        "playbackmethod":[
            1,
            3
        ],
        "delivery":[
            2
        ],
        "battr":[
            13,
            14
        ],
        "companionad":[
            {
                "id":"1234567893-1",
                "w":300,
                "h":250,
                "pos":1,
                "battr":[
                    13,
                    14
                ],
                "expdir":[
                    2,
                    4
                ]
            },
            {
                "id":"1234567893-2",
                "w":728,
                "h":90,
                "pos":1,
                "battr":[
                    13,
                    14
                ]
            }
```


(continues on next page)

(continued from previous page)

```
      ],
      "companiontype":[
         1,
         2
      ]
   }
}
```

6.5 Audio Object

Note: Fields marked with an asterisk (*) are optional.

Table 13: Audio Object Properties

Value	Type	Description
mimes	*array of strings*	Content MIME types supported, for example `["audio/mp4", "audio/mpeg"]`
*minduration**	*integer*	Minimum audio ad duration in seconds, for example, `2`
*maxduration**	*integer*	Maximum audio ad duration in seconds, for example, `15`
protocols	*array of integers*	Accepted audio bid response protocols as defined in OpenRTB, for example `[9,10]`
*startdelay**	*integer*	Indicates the start delay in seconds, or generic values below: • `0`: Pre-roll • `-1`: Generic mid-roll • `-2`: Generic post-roll
*battr**	*array of integers*	Blocked creative attributes as defined in OpenRTB, for example, `[6]`
*minbitrate**	*integer*	Minimum bit rate in Kbps, for example `32`
*maxbitrate**	*integer*	Maximum bit rate in Kbps, for example `320`
*api**	*array of integers*	List of supported API frameworks for this impression as defined in the OpenRTB guide, for example, `[1,2]`. If an API is not explicitly listed, it is assumed not to be supported.
*maxextended**	*integer*	Maximum extended audio ad duration if extension is allowed. • Blank or `0`, extension is not allowed. • `-1`, extension is allowed, and there is no time limit imposed. • Greater than `0`, then the value represents the number of seconds of extended play supported beyond the *maxduration* value.
*delivery**	*array of integers*	Supported delivery methods (e.g., streaming, progressive) as defined in OpenRTB. If none specified, assume all are supported, for example, `[1, 2]`
*maxseq**	*integer*	The maximum number of ads that can be played in an ad pod, for example, `1`
*feed**	*integer*	Type of audio feed, for example, `1`
*sequence**	*integer*	If multiple ad impressions are offered in the same bid request, the sequence number will allow for the coordinated delivery of multiple creatives, for example, `2`

Table 14: Audio Object Properties

Value	Type	Description
*stitched**	*integer*	Indicates if the ad is stitched with audio content or delivered independently, for example, `1`
*nvol**	*integer*	Volume normalization mode as defined in OpenRTB, for example, `1`
*companionad**	*array of objects*	Array of Banner objects if companion ads are available. See the *Banner Object* (page 17) section for more information.
*companiontype**	*array of integers*	Supported DAAST companion ad types, for example `[1, 2]` Possible values: • `1`: Static Resource • `2`: HTML Resource • `3`: iframe Resource

6.5.1 Audio Object Example

```
{
"id": "1",
"bidfloor": 0.03,
  "audio": {
    "startdelay": 0,
    "minduration": 5,
    "maxduration": 30,
    "maxextended": 30,
    "minbitrate": 300,
    "maxbitrate": 1500,
    "api": [
      1,
      2
    ],
    "protocols": [
      9,
      10
    ],
    "mimes": [
      "audio/aac",
      "audio/mp4",
      "audio/mpeg"
    ],
    "delivery": [
      2
    ],
    "battr": [
      13,
      14
    ],
    "companionad": [
```

(continues on next page)

(continued from previous page)

```
      {
      "id": "1234567893-1",
      "w": 300,
      "h": 250,
      "pos": 1,
        "battr": [
          13,
          14
        ],
      "expdir": [
                2,
                4
        ]
      },
      {
      "id": "1234567893-2",
      "w": 728,
      "h": 90,
      "pos": 1,
      "battr": [
        13,
        14
      ]
    }
    ],
    "companiontype": [
      1,
      2
    ]
  }
}
```

6.6 Native Object

Note: Fields marked with an asterisk (*) are optional.

Table 15: Native Object

Value	Type	Description
request_native	*object*	Contains the *Native Request Object* (page 29) object.
*battr**	*array of integers*	Blocked creative attributes as defined in OpenRTB., for example, `[1, 3]`
*api**	*array of integers*	List of supported API frameworks for this impression as defined in OpenRTB, for example `[2,3,5]`. If an API is not explicitly listed, it is assumed not to be supported.

6.6.1 Native Object Example

```
{
   "native":{
      "request_native":{
         "ver":"1.2",
         "layout":1,
         "adunit":4,
         "assets":[
            {
               "id":1,
               "required":1,
               "title":{
                  "len":25
               }
            }
         ]
      },
      "api":[
         3
      ],
      "battr":[
         13,
         14
      ]
   }
}
```

6.7 Native Request Object

Note: Fields marked with asterisk (*) are optional.

Table 16: Native Request Object

Value	Type	Description
ver	*string*	Version of the Native Markup in use, for example, `"1.2"`. **Note:** It must be `1.2`
*layout**	*integer*	The Layout ID of the native ad unit as described in OpenRTB Native specification, for example, `3`
*adunit**	*integer*	The Ad unit ID of the native ad unit as described in OpenRTB Native specification.
*plcmttype**	*integer*	The design/format/layout of the ad unit being offered. See the *Native Placement Type* (page 31) for a list of supported placement types
*plcmtcnt**	*integer*	The number of identical placements in this Layout, for example, `1`
*seq**	*integer*	0 for the first ad, 1 for the second ad, and so on. This is not the sequence number of the content in the stream.
*eventtrackers**	*array of objects*	Specifies what type of event tracking is supported, see *Event Tracker Request Object* (page 32). Required by some Buyers, see *Required Fields per Buyer* (page 54)
*privacy**	*integer*	Set to `1` when the native ad supports a buyer-specific privacy notice, set to `0` otherwise.
assets	*array of objects*	An array of Asset Objects. Any bid must comply with this array of elements. See the Native Asset Object section below for more details.

6.7.1 Native Asset Object

Table 17: Native Asset Object Properties

Value	Type	Description
id	*integer*	Unique asset id, for example `2`
*required**	*integer*	Set to 1 if asset is required (exchange will not accept a bid without it), default is 0.
title **	*object*	Native title object, see the Native Asset Title Object section below for more details.
img **	*object*	Native image object, see the Native Asset Image Object below for more details.
video **	*object*	Native video object, see the Native Asset Video Object below for more details.
data **	*object*	Native asset data object, see the Native Asset Data Object below section for more details.

Note: (**) There may only be exactly one of the fields marked with double asterisk in each asset object.

6.7.2 Native Asset Title Object

Table 18: Native Asset Title Object

Value	Type	Description
len	integer	Maximum length of the text in the title element, for example, `30`

6.7.3 Native Asset Image Object

The image asset object may contain the exact image size, the minimum image size, or both. If only the exact image size is specified then the image in the bid response should have the corresponding size. If the minimum size is specified then the image asset in the bid response should comply with the following restrictions.

- The size of the image should be equal to or larger than the minimum specified
- The image asset in the bid response should contain the `w` and `h` fields.

Note: It is recommended that the aspect ratio of the image should be close to the one specified by the exact size or by the minimum size; the acceptable aspect ratio deviation is from $0.8 * (w/h)$ to $1.25 * (w/h)$

Table 19: Native Asset Image Object

Value	Type	Description
*type**	*integer*	Image asset type, for example `3`. Takes the following values: • `1` Icon • `2` Logo (Logo image for the brand/app) • `3` Main (Large image preview for the ad)
*w**	*integer*	Width of the image in pixels, for example, `300`
*wmin**	*integer*	The minimum requested width of the image in pixels, for example, `100`
*h**	*integer*	Height of the image in pixels, for example, `250`
*hmin**	*integer*	The minimum requested height of the image in pixels, for example, `100`
*mimes**	*array of strings*	Whitelist of content MIME types supported, for example, `["image/gif"]` If blank, assume all types are allowed.

6.7.4 Native Asset Video Object

Table 20: Native Asset Video Object

Value	Type	Description
mimes	*array of strings*	Content MIME types supported, for example, `["video/mpeg", "video/mp4"]`
minduration	*integer*	Minimum video ad duration in seconds, for example, `2`
maxduration	*integer*	Maximum video ad duration in seconds, for example `15`
protocols	*array of integers*	Accepted video bid response protocols as defined in OpenRTB, for example, `[2,5]`

6.7.5 Native Asset Data Object

Table 21: Native Asset Data Object

Value	Type	Description
type	*integer*	Data asset type as described in OpenRTB Native specification, for example, `1`
*len**	*integer*	Maximum length of the text in the element's response, for example, `25`

6.7.6 Native Placement Type

Table 22: Native Placement Type Options

Value	Description
1	In the feed of content, for example as an item inside the organic feed/grid/listing/carousel.
2	In the atomic unit of the content, i.e. in the article page or single image page
3	Outside the core content, for example in the ads section on the right rail, as a banner-style placement near the content, etc.
4	Recommendation widget, most commonly presented below the article content.
500+	To be defined by the exchange

6.7.7 Event Tracker Request Object

Table 23: Event Tracker Request Object

Value	Type	Description
event	*integer*	Type of event available for tracking. See the *Event Tracking Types* (page 32)
method	*array of integers*	Array of the types of tracking available for the given event. See the *Event Tracking Methods* (page 32) table
*ext**	*object*	This object is a placeholder that may contain custom JSON agreed to by the parties to support flexibility beyond the standard defined in this specification

6.7.8 Event Tracking Types

Table 24: Event Tracking Types

Value	Type	Description
1	*Impression*	Impression
2	*viewable-mrc50*	Visible impression using MRC definition at 50% in view for 1 second.
3	*viewable-mrc100*	Visible impression using MRC definition at 100% in view for 1 second, i.e. GroupM standard
4	*viewable-video50*	Visible impression for video using MRC definition at 50% in view for 2 seconds.
500+	*exchange specific*	

6.7.9 Event Tracking Methods

Table 25: Event Tracking Methods

Value	Type	Description
1	*img*	Image-pixel tracking – The URL provided in the response will be inserted as a 1x1 pixel at the time of the event.
2	*js*	Javascript-based tracking – The URL provided in the response will be inserted as a js tag at the time of the event.
500+	*exchange specific*	Could include custom measurement companies such as Moat, DoubleVerify, IAS, etc – in this case additional elements will often be passed.

6.7.10 Example Native Request

```
{
   "native":{
      "request":{
         "plcmtcnt":1,
         "plcmttype":2,
         "privacy":1,
         "context":1,
         "contextsubtype":12,
         "eventtrackers":[
            {
               "event":1,
               "methods":[
                  1,
                  2
               ]
            },
            {
               "event":2,
               "methods":[
                  1
               ]
            }
         ],
         "assets":[
            {
               "id":1,
               "data":{
                  "type":12
               },
               "required":1
            },
            {
               "title":{
                  "len":50
               },
               "id":2,
               "required":1
            },
            {
               "id":3,
               "img":{
                  "w":80,
                  "h":80,
                  "type":1
               },
               "required":1
            },
            {
               "id":4,
               "img":{
```

(continues on next page)

(continued from previous page)


```
                "w":1200,
                "h":627,
                "type":3
            },
            "required":1
        },
        {
            "data":{
                "type":3
            },
            "id":5,
            "required":0
        },
        {
            "id":6,
            "data":{
                "len":100,
                "type":2
            },
            "required":1
        },
        {
            "id":7,
            "video":{
                "mimes":[
                    "video/mpeg",
                    "video/mp4"
                ],
                "minduration":2,
                "protocols":[
                    2,
                    5
                ],
                "maxduration":2,
                "ext":{
                    "playbackmethod":[
                        1,
                        2
                    ]
                }
            },
            "required":1
        }
    ],
    "ver":"1.2"
  }
 }
}
```

6.8 Private Marketplace Object

Note: Fields marked with an asterisk (*) are optional.

Table 26: Private Marketplace Object Properties

Value	Type	Description
*private_ auction**	*integer*	A value of 1 indicates that only bids submitted inside `pmp.deals` will take part in the auction. A value of 0 indicates that bids without deal information may also be considered for serving.
deals	*array of objects*	Array of Deal objects, for more information, see the *Deals Object* (page 36) section.

6.8.1 Private Marketplace Object Example

```
{
   "pmp":{
      "private_auction":1,
      "deals":[
         {
            "id":"deal-1",
            "wseat":[
               "58"
            ],
            "bidfloor":2.5,
            "at":1
         },
         {
            "id":"deal-2",
            "bidfloor":2,
            "at":2
         }
      ]
   }
}
```

6.9 Deals Object

Note: Fields marked with an asterisk (*) are optional.

Table 27: Deal Object Properties

Value	Type	Description
id	*string*	Deal id, for example, `"AA-1234"`
*wseat**	*array of strings*	Array of Buyer seats allowed to bid on this Direct Deal, for example, `[58, 99]`. If present, the allowed seat IDs may be supplied using the BidSwitch or Supplier taxonomy. • The BidSwitch taxonomy uses the Buyer ID as the single seat ID value. • The seat in the Supplier taxonomy may represent the whole Buyer or some entity on the Buyer side (e.g. agency) • A bid request may contain multiple seat IDs in the Supplier taxonomy. • The bid response should contain the appropriate seat value corresponding to one of values of the `wseat` field, see the *Seat Bid Object* (page 66) section.
*bidfloor**	*float*	Deal price in CPM. If it's a fixed price deal as set using `deals.at = 3` then this field sets the the exact price of the deal, otherwise this is the bid floor of the deal, for example, `1.3`
*bidfloorcur**	*string*	Bid floor currency specified using ISO-4217[7] alpha codes, for example, `"USD"`
*at**	*integer*	Auction type. • `1` for first price auction. • `2` for second price auction. • `3` for fixed price deal.

6.9.1 Deals Object Example

```
{
   "pmp":{
      "private_auction":1,
      "deals":[
         {
            "id":"deal-1",
            "wseat":[
               "58"
```

(continues on next page)

[7] https://www.iso.org/iso-4217-currency-codes.html

(continued from previous page)

```
            ],
            "bidfloor":2.5,
            "at":1
         },
         {
            "id":"deal-2",
            "bidfloor":2,
            "at":2
         }
      ]
   }
}
```

6.10 Device Object Properties

Note: Fields marked with an asterisk (*) are optional.

(**) Required for in-app requests.

Table 28: Device Object Properties

Value	Type	Description
geo	*object*	Geo Object as derived from the device's location services, or supplied by the Supplier if the device IP is missing. For more information, see the *Geo Object Properties* (page 41) section.
ip	*string*	Specifies the IPv4 address closest to the device. • Required for almost all requests, any containing invalid IP addresses will be discarded based on IP Validation e.g. `87.224.77.0` • **Note:** This field is only optional for Connected TV inventory
*ua**	string	Browser or application user agent string, for example, `"Mozilla/5.0 (Windows NT 6.3; WOW64; rv:35.0)Gecko/ 20100101Firefox/35.0"`
*language**	*string*	Alpha-2/ISO 639-1 code of browser language, for example, `en`
*carrier**	*string*	Carrier or ISP derived from the IP address, for example, `WIFI`
*connectiontype**	*integer*	Connection type as defined in OpenRTB, for example, `2`
*didsha1**	*string*	Hardware device ID (e.g., IMEI); hashed via SHA1, for example, `CCF6DC12B98AEB2346AFE1BEE7860DF01FDE158B`
didmd5*	string	Hardware device ID (e.g., IMEI); hashed via MD5. `93D05D4D69DEE2BC6645D9F0A0C1938C`
*dpidsha1**	*string*	Platform device ID (e.g., Android ID); hashed via SHA1, for example, `CCF6DC12B98AEB2346AFE1BEE7860DF01FDE158B`
*dpidmd5**	string	Platform device ID (e.g., Android ID); hashed via MD5, for example, `93D05D4D69DEE2BC6645D9F0A0C1938C`
*ifa***	*string*	ID sanctioned for advertiser use in clear text (i.e. not hashed), for example Apple's IDFA or Android's Advertising ID. **Note:** This field is required for in-app requests. The Apple IDFA is usually uppercase, and the Android Advertiser ID is usually lowercase. For example: • Android `"035911ea-467d-4056-903b-65cf44f5633b"` • iOS `"30255BCE-4CDA-4F62-91DC-4758FDFF8512"` **Note:** This fields uses UUIDv4 format. Version 4 UUIDs take the following format: `xxxxxxxx-xxxx-4xxx-yxxx-xxxxxxxxxxxx`, where `x` is any hexadecimal digit and `y` is one of 8, 9, A, or B. You can read more about UUIDv4 here: https://tools.ietf.org/html/rfc4122
*make**	*string*	Device make, for example, `Apple`
*mccmnc**	*string*	Mobile carrier as the concatenated MCC-MNC code (e.g., `"310-005"` identifies Verizon Wireless CDMA in the USA). Refer to https://en.wikipedia.org/wiki/Mobile_country_code for further examples. **Note:** that the dash between the MCC and MNC parts is required to remove parsing ambiguity

Table 29: Device Object Properties

Value	Type	Description
*model**	*string*	Device mode, for example, `iPhone`
*os**	*string*	Device operating system, for example, `iOS`
*osv**	*string*	Device operating system version, for example, `3.1.2`
*w**	integer	Physical height of the screen in pixels, for example, `750`
*h**	*integer*	Physical width of the screen in pixels, for example, `1334`
*pxratio**	*float*	The ratio of physical pixels to device independent pixels, for example, `1.0`
*dnt**	*integer*	Do not track. • `0`: do not track is set to false • `1`: do not track is set to true in the browser, for example, `0`
*lmt**	*integer*	Limit Ad Tracking. Signal commercially endorsed (e.g., iOS, recommended Android): • `0`: tracking is unrestricted, • `1`: tracking must be limited per commercial guidelines, for example, `0`
*devicetype**	*integer*	Device type as defined by OpenRTB, for example, `4`
*ipv6**	*string*	IP address in IPv6, for example, `fe80:0:0:0:200:f8ff:fe21:67cf`
*js**	*integer*	1 if the device supports JavaScript; otherwise 0.
*flashver**	*string*	Flash version detected, for example, `10.1`

6.11 Geo Object Properties

The information provided in the Geo Object is based on MaxMind database[8], except latitude and longitude values.

Note: Fields marked with an asterisk (*) are optional.

Table 30: Geo Object Properties

Value	Type	Description
*lat**	*float*	Latitude from -90 to 90. South is negative, for example, `52.35`
*lon**	*float*	Longitude from -180 to 180. West is negative, for example, `4.9167`
*type**	*integer*	Source of location data as defined by OpenRTB, for example, `1`
*country**	*string*	Country using ISO-3166-1[9] Alpha-2, for example `NL`
*region**	*string*	Region using ISO-3166-2 region codes, for example, `NY`
*city**	*string*	City name.
*zip**	*string*	Zip/postal code, for example, `"90210"`
*utcoffset**	*integer*	Local time as the number +/- of minutes from UTC, for example, `-240`

6.11.1 Geo Object Example

```
{
"geo": {
      "country": "US",
      "region": "NY",
      "city": "City Name",
      "zip": "10601",
      "utcoffset": -240
}}
```

[8] https://www.maxmind.com/en/geoip2-services-and-databases
[9] https://www.iso.org/iso-3166-country-codes.html

6.12 Metric Object

Note: Fields marked with an asterisk (*) are optional.

Table 31: Metric Object Properties

Value	Type	Description
type	*string*	The type of metric being presented. Currently BidSwitch only supports using `viewability` as the metric type
value	float	A decimal number representing the value of the metric being supplied • `viewability` probability is in the range *0.0 – 1.0*.
*vendor**	string	Source of the value declared by the Supplier

6.12.1 Metric JSON Example

```
{
  "metric":[
    {
      "type":"viewability",
      "value":0.85
    }
  ]
}
```

6.13 User Object

Note: Fields marked with asterisk (*) are optional.

Table 32: User Object Properties

Value	Type	Description
*id**	*string*	Unique BidSwitch ID of this user, for example, `252eb154-b3e5-473f-bad8-9b6d7f8646e5`. For in-app traffic the lowercase IDFA, or Android ID is used. For example, ‘“”38f72eaf-5d6f-4143-824f-deaf753d7239”‘
*buyeruid**	*string*	The Buyer user ID as mapped by BidSwitch for the DSP. ‘
*keywords**	*string*	Comma separated list of keywords, interests, or intent, for example, `Cars, sports, vacation`
*yob**	*integer*	Year of birth as a 4-digit integer, for example, `1977`
*gender**	*string*	Specifies the user gender, for example, `"F"` • `"M"` = Male • `"F"` = Female • `"O"` = Known to be other, or omitted if unknown
*data**	*array of objects*	Additional data. Each data object represents a different data source, for more information, see the *Data Object* (page 45) section.
*ext**	*object*	See *User Ext Object* (page 43)

6.13.1 User Ext Object

Table 33: User Ext Object Properties

Value	Type	Description
*consented_providers_settings**	*object*	Passes a set of IDs corresponding to providers for whom the publisher has provided user consent using Google vendor list. See the *Consented Provider Settings* (page 44)
*consent**	*string*	The binary encoding scheme that is passed in base64 URL/web safe format known as daisybit, e.g. `"Y29uc2VudCBkYXRh"` The data stored in the consent string is divided into 3 parts: metadata, the purposes for which the user has given consent, and to which vendors this consent was given. The Supplier should pass this information to Buyers to ensure they can bid appropriately in their responses. For more information see the following links: • http://advertisingconsent.eu/ • https://github.com/InteractiveAdvertisingBureau/GDPR-Transparency-and-Consent-Framework • https://gdpr-info.eu/ • http://gdpr-demo.labs.quantcast.com/user-examples/cookie-workshop.html

6.13.2 Consented Provider Settings

Table 34: Consented Providers

Value	Type	Description
consented_providers	*array of integers*	Set of IDs corresponding to providers for whom the publisher has provided user consent using Google vendor list. A mapping of provider ID to provider name is posted at https://storage.googleapis.com/adx-rtb-dictionaries/providers.csv

6.13.3 User Object Example

```
{
    "user":{
        "id":"45asdf987656789adfad4678rew656789",
        "buyeruid":"1234567890",
        "keywords":"sports, entertainment",
        "yob":1976,
        "gender":"F",
        "ext":{
            "consent":"Y29uc2VudCBkYXRh",
            "consented_providers_settings":{
                "consented_providers":[
                    1791
                ]
            }
        }
    }
}
```

6.14 Ext Object

Note: Fields marked with asterisk (*) are optional.

Table 35: Ext Object Properties

Value	Type	Description
*dsp_uuids**	*object*	key-value user ID pairs to support direct-like user syncs, see the *Supplier First-Party User-Matching* (page 110) section for more details.
*google_query_id**	*string*	This represents a unique ID for the overall query. In the event that there are multiple call-outs for a query, all call-out requests for that query will contain the same `google_query_id`, see more here: https://developers.google.com/authorized-buyers/rtb/openrtb-guide#bidrequestext It is highly recommended to include this ID if you are selling Exchange Bidding Dynamic Allocation (EDBA) sourced inventory through BidSwitch to Google DV360.

```
{
  "ext":{
    "dsp_uuids":{
      "77":"xyz"
    }
  }
}
```

6.15 Data Object

Note: Fields marked with asterisk (*) are optional.

Table 36: Data Object Properties

Value	Type	Description
id *	*string*	Exchange-specific ID for the data provider, for example `"BSW001"`
name	*string*	Exchange-specific name for the data provider, for example `"domain-origin"`
segment	*array of objects*	Array of Segment objects that contain the actual data values.

6.15.1 Segment Object

Table 37: Segment Object Properties

Value	Type	Description
*id**	string	ID of the data segment specific to the data provider, for example, `"Seg123"`
name	string	Name of the data segment specific to the data provider, for example, `"status"`
*value**	*string*	String representation of the data segment value, for example, `"verified"`

```
{
   "data":[
      {
         "name":"domain-origin",
         "segment":[
            {
               "name":"status",
               "value":"verified"
            },
            {
               "name":"domain",
               "value":"abcd.com"
            }
         ]
      }
   ]
}
```

6.16 Source Object

Note: Fields marked with an asterisk (*) are optional.

Table 38: Source Object Properties

Value	Type	Description
fd	*integer*	Indicates the entity responsible for the final impression sale decision, using the following values: • `0` = The exchange behind BidSwitch • `1` = An upstream source (usually header bidder) For example a bid request containing `ext.ssp='rubicon'` and `source.fd=0` implies that the auction is run at Rubicon SSP. If the same request has `source.fd=1` then the auction is run at a header bidder upstream from Rubicon SSP. **Note**: The BidSwitch platform never acts as the decision maker.
tid	string	(**Recommended**) Transaction ID that must be common across all participants in this bid request (e.g., potentially multiple exchanges).
*pchain**	*string*	Payment ID chain string containing embedded syntax described in the TAG Payment ID Protocol v1.0.
*ext**	*object*	Contains additional fields, see *Source Extension* (page 47)

6.16.1 Source Extension

Table 39: Source Extension Object Properties

Value	Type	Description
omidpn	*string*	Identifier of the OM SDK integration, the IAB Open Measurement specification on Github[10]
omidpv	*string*	Version of the OM SDK integration.
schain	*object*	Contains the supplychain object, as fully described here on the IAB Github Page[11]: The SupplyChain object is composed primarily of a set of nodes where each node represents a specific entity that participates in the selling of a bid request. The entire chain of nodes from beginning to end would represent all sellers who were paid for an individual bid request.

[10] https://github.com/InteractiveAdvertisingBureau/AdCOM/blob/master/OpenRTB%20support%20for%20OMSDK.md

[11] https://github.com/InteractiveAdvertisingBureau/openrtb/blob/master/supplychainobject.md

6.16.2 SupplyChain Object

Table 40: schain

Value	Type	Description
complete	*int*	(**Required**) Flag indicating whether the chain contains all nodes leading back to the source of the inventory, where `0` = no, `1` = yes.
nodes	*array of objects*	(**Required**) Array of objects in the order of placing in the chain. The original source of the request is first and the final seller of the request last, see *Supply Chain Nodes* (page 49)
ver	*str*	(**Required**) Version of the supply chain specification in use. Currently `"1.0"` is the only option.

6.16.3 Supply Chain Nodes

Table 41: supply chain node

Value	Type	Description
asi	*string*	(**Required**) The canonical domain name of the SSP, Exchange, Header Wrapper, etc system that bidders connect to. This may be the operational domain of the system, if that is different than the parent corporate domain, to facilitate WHOIS and reverse IP lookups to establish clear ownership of the delegate system. This should be the same value as used to identify sellers in an ads.txt file if one exists.
sid	*string*	(**Required**) The identifier associated with the seller or reseller account within the advertising system. This must contain the same value used in transactions (i.e. OpenRTB bid requests) in the field specified by the SSP/exchange. Typically, in OpenRTB, this is publisher.id. For OpenDirect it is typically the publisher's organization ID.Should be limited to 64 characters in length.
hp	*int*	(**Required**) Indicates whether this node will be involved in the flow of payment for the inventory. When set to `1`, the advertising system in the `asi` field pays the seller in the `sid` field, who is responsible for paying the previous node in the chain. When set to `0`, this node is not involved in the flow of payment for the inventory. For version 1.0 of SupplyChain, this property should always be `1`. It is explicitly required to be included as it is expected that future versions of the specification will introduce non-payment handling nodes. Implementers should ensure that they support this field and propagate it onwards when constructing SupplyChain objects in bid requests sent to a downstream advertising system.
rid	*string*	The OpenRTB RequestId of the request as issued by this seller.
*name**	*string*	The business name of the entity represented by this node. This value is optional and should NOT be included if it exists in the advertising system's sellers.txt file.
*domain**	*string*	The business domain name of the entity represented by this node. This value is optional and should NOT be included if it exists in the advertising system's sellers.txt file.

6.16.4 Example Source JSON

```
{
    "source":{
        "fd":1,
```

(continues on next page)

(continued from previous page)

```
        "ext":{
            "schain":{
                "complete":0,
                "ver":"1.0",
                "nodes":[
                    {
                        "asi":"reseller.com",
                        "sid":"aaaaa",
                        "rid":"BidRequest4",
                        "hp":1
                    }
                ]
            }
        }
    }
}
```

6.17 Site Object

Note: Fields marked with an asterisk (*) are optional.

Table 42: Site Object Properties

Value	Type	Description
publisher	*object*	Publisher object, for more information, see the *Publisher Object* (page 53) section.
*id**	*string*	An exchange specific identifier.
*name**	*string*	Site name (may be masked by publisher request), for example, `"Test Site"`
*domain**	*string*	Domain of the site, used for advertiser side blocking. `"testsite.com"`
*cat**	*array of strings*	Array of IAB content categories for the site, `["IAB1", "IAB2-3"]`. Based on the IAB taxonomy, and extended with additional sensitive categories listed in the *Sensitive Categories and Rich Media* (page 3) section. The content categories specified can be extended using BidSwitch *Sensitive Categories and Rich Media* (page 3).
*page**	*string*	URL of the page where the impression will be shown. `"http://testsite.com/main.asp"`
*ref**	*string*	Referrer URL that caused navigation to the current page, for example, `"http://testsite.com/main.asp"`
*privacypolicy**	*integer*	Indicates if the site has a privacy policy. • `0` = No • `1` = Yes.
*mobile**	*integer*	Mobile-optimized signal. • `0` = No • `1` = Yes.

6.17.1 SSP Site Object Example

```
{
   "site":{
      "id":"abc35123",
      "name":"Site ABCD",
      "domain":"siteabcd.com",
      "cat":[
         "IAB2-1",
         "IAB2-2"
      ],
      "page":"http://siteabcd.com/page.htm",
      "ref":"http://referringsite.com/referringpage.htm",
      "privacypolicy":1,
      "publisher":{
         "id":"abc2345",
         "name":"Publisher A"
      }
```

(continues on next page)

(continued from previous page)

```
    }
}
```

6.18 App Object

Note: Fields marked with an asterisk (*) are optional.

Table 43: App Object Properties

Value	Type	Description
publisher	*object*	Publisher object, for more information, see the *Publisher Object* (page 53) section.
*id**	*string*	The application ID.
*name**	*string*	Application name, for example, `"Test App"`
*domain**	*string*	The domain of the app, for example, `"mygame.example.com"`
*cat**	*array of strings*	Array of IAB content categories for the publisher site, for example, `["IAB1", "IAB2- 3"]`. Based on the IAB taxonomy, and extended with additional sensitive categories listed in the *Sensitive Categories and Rich Media* (page 3) section.
*bundle**	*string*	Application bundle or package name, for example, `"com.example.mygame"`
*paid**	*integer*	Specifies if the App is a free or paid version. • `0` = The app is free, • `1` = The app is a paid version.
*storeurl**	*string*	App store's URL for the mobile application, for example `"http://media-apps.cc/android"`
*ver**	*string*	Application version, for example `"1.1"`
*privacypolicy**	*integer*	Indicates if the app has a privacy policy. • `0` = No • `1` = Yes.

6.19 Publisher Object

Table 44: Publisher Object Properties

Value	Type	Description
id	*string*	Exchange-specific publisher ID.
*name**	string	Publisher name, for example `"AAP"`
*cat**	array of string	Array of IAB content categories for the publisher. `["IAB1", "IAB2-3"]`

Note: Fields marked with an asterisk (*) are optional and may not be sent in each request.

6.19.1 Publisher Object Example

```
{
   "publisher":{
      "id":"abc123",
      "name":"Publisher A"
   }
}
```

6.20 Regulation Object

Table 45: Regulation Object Properties

Value	Type	Description
coppa	*integer*	Flag indicating whether or not this request falls under the COPPA regulations established by the USA FTC: • `0` = No. • `1` = Yes.
*ext**	*object*	See the *Regs Ext Object* (page 54)

6.20.1 Regs Ext Object

Table 46: Regs Ext Object Properties

Value	Type	Description
*gdpr**	*integer*	Indicates whether the request falls under GDPR regulations: • `0` = No • `1` = Yes • Under OpenRTB conventions for optional attributes, omission indicates **Unknown** If consent is given, you should check if the `user.ext.consent` field is present to ascertain what form of consent was given, see the *User Ext Object* (page 43) section

```
{
   "regs":{
      "ext":{
         "gdpr":1
      }
   }
}
```

6.21 Required Fields per Buyer

Some Buyers request or require certain fields be sent to them that are above and beyond the normal IAB specification. This usually indicates fields that they deem important and thus including them will increase their interest in your inventory.

Table 47: Recommended Fields per Buyer

Buyer	Required or Recommended fields
DV360	• `native.request.eventtrackers.event` with value `1` • `native.request.eventtrackers.methods` with value `1` • `native.request.plcmttype` (recommended) • See the *Native Request Object* (page 29) section for more details.

Note: This table of fields does not include those fields that are required as part of this protocol.

6.22 Bid Request JSON Examples

6.22.1 Banner Ad Request Example


```
{
   "id":"c6987c2b-edb4-4b7b-b8cf-157af1d485e3",
   "regs":{
      "ext":{
         "gdpr":1
      }
   },
   "site":{
      "id":"ed2265d8",
      "ref":"http://ad32.answers.com/click.php?source=fb&param4=fb-us-de-red&param3=www.
↪answers.com%2Farticle%2F31029589%2Finsanely-useful-life-hacks-to-make-everything-
↪easier&param1=tattoo&param2=67660042&param5=10153631993521186&param6=6049542139960&
↪adt=4342",
      "publisher":{
         "name":"www.answers.com",
         "id":"946353442_12535"
      },
      "name":"www.answers.com",
      "cat":[
         "IAB24"
      ],
      "domain":"answers.com",
      "page":"http://www.answers.com/article/31029589/insanely-useful-life-hacks-to-make-
↪everything-easier?paramt=null&param4=fb-us-de-red&param1=tattoo&param2=67660042&s=8"
   },
   "wseat":[
      "165",
      "16"
   ],
   "source":{
      "fd":0
   },
   "ext":{
      "dsp_uuids":{
         "77":"xyz"
      }
   },
   "user":{
      "id":"5e29eb00-c30a-416e-9d2a-2e18901f0916",
      "buyeruid":"CAESEHL-9O4oJOAiC1YOO2EHTcE",
      "ext":{
```


(continues on next page)

(continued from previous page)


```
        "consent":"Y29uc2VudCBkYXRh"
     }
  },
  "device":{
     "pxratio":0,
     "language":"en",
     "mccmnc":"310-005",
     "w":1920,
     "geo":{
        "country":"US",
        "lon":-80.237,
        "city":"City Name",
        "lat":26.638,
        "zip":"33414",
        "region":"FL",
        "type":2
     },
     "os":"Windows",
     "devicetype":2,
     "h":1080,
     "ip":"73.139.39.18",
     "js":1,
     "ua":"Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0",
     "dnt":0
  },
  "tmax":75,
  "cur":[
     "USD"
  ],
  "imp":[
     {
        "bidfloor":3.213,
        "metric":[
           {
              "type":"viewability",
              "value":0.85
           }
        ],
        "id":"1",
        "banner":{
           "pos":1,
           "h":600,
           "battr":[
              1,
              3,
              5,
              6,
              8,
              9,
              10,
              14,
```


(continues on next page)

(continued from previous page)


```
                15,
                16
            ],
            "w":160,
            "format":[
                {
                    "h":300,
                    "w":300
                },
                {
                    "h":350,
                    "w":300
                }
            ],
            "btype":[
                1
            ]
        },
        "exp":300,
        "tagid":"25108",
        "bidfloorcur":"USD",
        "secure":0,
        "instl":0
    }
  ],
  "bcat":[
    "IAB25-3",
    "BSW1",
    "BSW2",
    "BSW10",
    "BSW4",
    "IAB26"
  ],
  "at":2
}
```


6.22.2 Native Ad Request Example


```
{
  "id":"129ca6dd-5403-4476-a4a6-555d6a538bc4",
  "app":{
    "id":"1009429",
    "publisher":{
      "name":"",
      "id":"trw1005292"
    },
    "storeurl":"https://play.google.com/store/apps/details?id=com.leo.appmaster",
    "bundle":"com.leo.appmaster",
    "cat":[
      "IAB3"
```


(continues on next page)

(continued from previous page)


```
      ],
      "name":"PG_lock_pic"
   },
   "wseat":[
      "167"
   ],
   "source":{
      "fd":0
    },
  "ext":{
    "dsp_uuids":{
      "77":"xyz"
      }
    },
   "user":{
      "id":"793ff4b0-d077-4002-aeb6-b8ea64dd4b2b"
   },
   "device":{
      "connectiontype":3,
      "model":"Micromax A096",
      "mccmnc":"310-005",
      "language":"en",
      "geo":{
         "country":"IN",
         "lon":85.1167,
         "city":"Patna",
         "lat":25.6,
         "zip":"800002",
         "region":"34",
         "type":2
      },
      "ifa":"793ff4b0-d077-4002-aeb6-b8ea64dd4b2b",
      "osv":"5.0.2",
      "os":"Android",
      "carrier":"Airtel",
      "devicetype":1,
      "ip":"223.176.12.242",
      "ua":"Dalvik/2.1.0 (Linux; U; Android 5.0.2; Micromax A096 Build/LRX21M)",
      "dnt":2
   },
   "tmax":80,
   "cur":[
      "USD"
   ],
   "imp":[
      {
         "bidfloor":0.324,
         "metric":[
            {
              "type":"viewability",
              "value":0.85
```


(continues on next page)

(continued from previous page)

```
            }
         ],
         "id":"1",
         "native":{
            "request_native":{
               "plcmtcnt":1,
               "assets":[
                  {
                     "id":1,
                     "data":{
                        "type":12
                     },
                     "required":1
                  },
                  {
                     "title":{
                        "len":50
                     },
                     "id":2,
                     "required":1
                  },
                  {
                     "id":3,
                     "img":{
                        "w":80,
                        "h":80,
                        "type":1
                     },
                     "required":1
                  },
                  {
                     "id":4,
                     "img":{
                        "w":1200,
                        "h":627,
                        "type":3
                     },
                     "required":1
                  },
                  {
                     "data":{
                        "type":3
                     },
                     "id":5,
                     "required":0
                  },
                  {
                     "id":6,
                     "data":{
                        "len":100,
                        "type":2
```

(continues on next page)

(continued from previous page)


```
                    },
                    "required":1
                }
            ],
            "ver":"1.2"
        }
      },
      "exp":1800,
      "bidfloorcur":"USD",
      "instl":0
   }
],
"bcat":[
   "IAB25-3",
   "BSW1",
   "BSW2",
   "BSW10",
   "BSW4",
   "IAB26"
],
"at":2
}
```


6.22.3 Video Ad Request Example


```
{
   "regs":{
      "ext":{
         "gdpr":1
      }
   },
   "id":"75c0238c-3b52-4b87-957a-817f83e853f1",
   "site":{
      "id":"adaptv_",
      "publisher":{
         "name":"",
         "id":"tv4182"
      },
      "cat":[
         "IAB1"
      ],
      "page":"http://kissasian.com"
   },
   "wseat":[
      "126"
   ],
   "source":{
      "fd":0
   },
   "user":{
```


(continues on next page)

(continued from previous page)


```
    "id":"b457c658-ffdc-415c-8d91-30d864f4a5f5",
    "buyeruid":"7bcb7e7c-eff0-43ad-8522-b5c9251f0d43",
    "ext":{
       "consent":"Y29uc2VudCBkYXRh"
    }
 },
 "device":{
    "language":"en",
    "mccmnc":"310-005",
    "geo":{
       "country":"US",
       "lon":-75.15,
       "city":"Philadelphia",
       "lat":39.94,
       "zip":"19147",
       "region":"PA",
       "type":2
    },
    "lmt":0,
    "os":"Other",
    "devicetype":6,
    "ip":"73.141.79.240",
    "ua":"Mozilla/5.0 (PlayStation 4 3.55) AppleWebKit/537.78 (KHTML, like Gecko)"
 },
 "tmax":120,
 "cur":[
    "USD"
 ],
 "imp":[
    {
       "bidfloor":0.02268,
       "metric":[
          {
             "type":"viewability",
             "value":0.85
          }
       ],
       "id":"1",
       "instl":0,
       "exp":300,
       "bidfloorcur":"USD",
       "secure":0,
       "video":{
          "protocols":[
             2,
             5
          ],
          "placement":2,
          "playbackend":1,
          "minduration":5,
          "skip":1,
```


(continues on next page)

(continued from previous page)

```
            "playbackmethod":[
               3
            ],
            "maxduration":60,
            "startdelay":0,
            "linearity":1,
            "mimes":[
               "video/mp4"
            ]
         }
      }
   ],
   "bcat":[
      "IAB25-3",
      "BSW1",
      "BSW2",
      "BSW10",
      "BSW4",
      "IAB26"
   ],
   "at":2
}
```

6.22.4 Audio Ad Request Example

```
{
  "id":"1234534625253",
  "wseat":[
    "58"
  ],
  "source":{
    "fd":0
  },
  "imp":[
    {
      "id":"1",
      "secure":1,
      "audio":{
        "startdelay":0,
        "minduration":5,
        "maxduration":30,
        "maxextended":30,
        "minbitrate":300,
        "maxbitrate":1500,
        "api":[
          1,
          2
        ],
        "protocols":[
          9,
```

(continues on next page)

(continued from previous page)


```
          10
        ],
        "mimes":[
          "audio/aac",
          "audio/mp4",
          "audio/mpeg"
        ],
        "delivery":[
          2
        ],
        "battr":[
          13,
          14
        ],
        "companionad":[
          {
            "id":"1234567893-1",
            "w":300,
            "h":250,
            "pos":1,
            "battr":[
              13,
              14
            ],
            "expdir":[
              2,
              4
            ]
          },
          {
            "id":"1234567893-2",
            "w":728,
            "h":90,
            "pos":1,
            "battr":[
              13,
              14
            ]
          }
        ],
        "companiontype":[
          1,
          2
        ]
      }
    }
  ],
  "site":{
    "id":"google_234563",
    "domain":"siteabcd.com",
    "page":"https://siteabcd.com/page.htm",
```


(continues on next page)

(continued from previous page)


```
    "ref":"http://google.com/?q=siteabcd",
    "publisher":{
      "id":"google_25"
    }
  },
  "device":{
    "ip":"64.124.253.1",
    "mccmnc":"310-005",
    "geo":{
      "country":"US",
      "region":"NY",
      "city":"White Plains",
      "zip":"10601"
    },
    "ua":"Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10.6; en-US; rv:1.9.2.16) Gecko/
↪20110319 Firefox/3.6.16",
    "language":"en"
  },
  "ext":{
    "dsp_uuids":{
      "77":"xyz"
      }
    },
  "user":{
    "id":"45asdf987656789adfad4678rew656789",
    "buyeruid":"1234567890",
    "cur":[
      "USD"
    ]
  }
}
```

CHAPTER 7

SUPPLIER BID RESPONSE

This is the top level object that is sent by BidSwitch to the Supplier. Each bid request sent should contain the following fields.

Table 1: Bid Response Object Properties

Value	Type	Description
id	*string*	Specifies the ID of the bid request to which this is a response to, for example, `"d7d1e107-fe7c-4a57-9592-d1d41fa702d9"`
seatbid	*array of objects*	An array of Seat Bid objects, see the *Seat Bid Object* (page 66) section. The length of the array can be either 1+ (for yes-bid) or 0 (for no-bid).
*cur**	*string*	Sets the bidding currency using ISO-4217[12] alphabetic codes. If not provided USD is assumed, `"USD"`
*nbr**	*integer*	Returns a reason why the impression was not forwarded to any Buyers. A No Bid Reason response if different to a No Bid Response. See the *Supplier No Bid Reason* (page 104) section for details.

[12] https://www.iso.org/iso-4217-currency-codes.html

7.1 Seat Bid Object

Table 2: seatbid Object Properties

Value	Type	Description
bid	*array of objects*	Array of Bid Objects, see *Response Bid Object* (page 66).
*seat**	string	ID of the bidder seat on whose behalf this bid is made. The value should match one of the values supplied in the `wseat` field of the bid request and it is REQUIRED if the `wseat` field is present in bid request. For example, `"34"`

Note: Fields marked with asterisk (*) are optional.

7.2 Response Bid Object

Note:

- (*) Fields marked with an asterisk are optional.
- While individually neither of the following fields is required, one of them must be in the response: `adm`, `adm_native`.

Table 3: Bid Object Properties

Value	Type	Description
id	*string*	A bidder generated ID for the bid object, used for tracking and debugging purposes, for example `3`.
impid	*string*	The ID of the impression object (`imp`) from the bid request to which this bid response applies, for example `"1"`
price	*float*	The bid price as a float value, expressed as CPM. All prices assumed to be in USD if the `cur` parameter is omitted, for example `1.23`
*protocol**	*integer*	The Video response protocol of the markup if applicable, see the *Video Response Protocols* (page 69) table for the valid values. **Note:** This field is required in video responses.
*adm**	*string*	Used to pass creative markup for display (banner), video, or audio ads. One of either `adm` or `adm_native` should be present in the response. • This field can contain the win price macro. • This field is not used for native bid responses. `<a href=\"http://adserver.com/click?adid=125\"> <img src=\"http://image1.cdn.com/impid=102\"/></a>`
*adm_native**	*object*	Used for native bid responses, see the *Native Response Object* (page 69) for the data it contains. One of either `adm` or `adm_native` should be present in the response.
burl	*string*	The Billing notice URL called by the exchange using a server-to-server call when a winning bid becomes billable based on exchange-specific business policy (e.g., typically delivered, viewed, etc.)." This field **should** contain the win price macro, see the *Macros* (page 8) section. `"burl":"https://adserver.com/winnotice?impid=102&winprice=${AUCTION_PRICE}"`
*iurl**	*string*	Sample image URL (without cache busting) for content checking. **REQUIRED** when bidding on on banner bid requests. `"http://adserver.com/preview?impid=102"`
*language**	*string*	The Alpha-2 ISO 639-1[13] code for the creative's language, for example, `ja`. The nonstandard code `"xx"` may also be used if the creative has no linguistic content (e.g., a banner with just a company logo).
adomain	*array of strings*	Advertiser's primary or top-level domain for advertiser checking. This can be a list of domains if there is a rotating creative. Note that some Suppliers allow only one domain. To those Suppliers BidSwitch only sends the first domain from the list, for example, `["advertiser.com"]`
*cat**	*array of strings*	Array of IAB content categories, for example, `["IAB1", "IAB2-3"]`. Based on the IAB taxonomy, and extended with additional sensitive categories listed in the *Sensitive Categories and Rich Media* (page 3) section.

Table 4: Bid Object Properties

Value	Type	Description
*cid**	*string*	Campaign ID or similar that is used by the Buyer to track and organize their campaigns, for example, `102`.
crid	*string*	Creative ID to assist with ad quality checking, for example ‘‘`3021`’’
*attr**	*array of integers*	Creative attributes as defined in the OpenRTB protocol, for example, `[1,3]`.
*dealid**	*string*	Reference to the `deal.id` from the bid request, if this bid pertains to a private marketplace direct deal, for example, `"AA-1234"`
*h**	*integer*	The height of the creative in pixels when an alternative ad size is used, relevant for banner ads only. `250`
*w**	*integer*	The width of the creative in pixels when an alternative ad size is used, relevant for banner ads only. `300`
*ext**	object	This field can be used to supply information about the creative agency for whom the Buyer is working, see the Bid Ext Object section below for details.

7.2.1 Bid Ext Object

Table 5: Bid Ext Object Properties

Value	Type	Description
*advertiser_name**	*string*	The name of the advertiser serving the creative, for example, `"Coca-Cola"`
*agency_name**	*string*	The name of the agency representing the advertiser, for example, `"CCA"`
*agency_id**	*string*	ID of the agency representing the advertiser, for example, ‘‘`123`’’

[13] https://www.iso.org/iso-639-language-codes.html

7.2.2 Video Response Protocols

Table 6: Video Response Protocols

Value	Type
1	VAST 1.0
2	VAST 2.0
3	VAST 3.0
4	VAST 1.0 Wrapper
5	VAST 2.0 Wrapper
6	VAST 3.0 Wrapper
7	VAST 4.0
8	VAST 4.0 Wrapper
9	DAAST 1.0
10	DAAST 1.0 Wrapper

7.3 Native Response Object

Note: Fields marked with an asterisk (*) are optional.

Table 7: Native Object Properties

Value	Type	Description
assets	*array of objects*	List of native ad assets.
link	*object*	The Native Link Object[14]. This is the default link object for the ad. Individual assets can also have a link object which applies if the asset is activated (clicked). If the asset has no link object, the parent link object applies.
imptrackers **	*array of strings*	An array of impression tracking URLs, expected to return a 1x1 image or 204 response, for example, `["http://adserver.com/native?impid=102"]` **Note:** This field can contain the win price macro, see the *Macros* (page 8) section for more details
*ver**	*string*	Version of the Native Markup version in use, for example, `"1.2"`.
*jstracker**	string	Optional JavaScript impression tracker. This should be valid HTML with JavaScript already wrapped in `<script>` tags. It will be executed at impression time where it can be supported. **Note:** This field can contain the win price macro, see the *Macros* (page 8) section for more details
*eventtrackers**	*array of objects*	Array of tracking objects to run with the ad, in response to the declared supported methods in the request. The link see *Event Tracker Request Object* (page 32) for details.
*privacy**	*string*	If support for this was indicated in the request, sets the URL of a page informing the user about the buyer's targeting activity, e.g `https://www.example.com/privacy-notice`

7.3.1 Native Assets Object

Note:

- (*) There may be exactly one of the fields marked with asterisk in one asset object.
- (**) The link object is optional and may not be present in each response.

[14] https://protocol.bidswitch.com/standards/native-object.html#native-link-obj

Table 8: Native Asset Object Properties

Value	Type	Description
id	*integer*	A unique asset ID, must match one of the asset IDs in the bid request, for example, `1`.
*required**	*integer*	Set to `1` if the asset is required (bidder requires it to be displayed), default is `0`, for example, `1`.
*title**	*object*	Title object for a title asset, see Native Assets Title Object below.
*img**	*object*	Image object for an image asset, see Native Assets Image Object below.
*video**	*object*	Video object for a video asset, see Native Asset Video Object below.
*data**	*object*	Data object for a data asset, see Native Asset Data Object below.
link **	*object*	Link object for a call to action. • The link object applies if the asset item is activated (clicked). • If there is no link object on the asset, the parent link object on the bid response applies. See Native Link Object below.

7.3.2 Native Assets Title Object

Table 9: Native Asset Title Object Properties

Value	Type	Description
*text**	string	The text associated with the title element. `"Our product is the best!"`

7.3.3 Native Assets Image Object

Table 10: Native Asset Image Object Properties

Value	Type	Description
url	*string*	URL of the image asset, for example, `"http://adserver.com/image?impid=102"`.
h	*integer*	Height of the image in pixels, for example, `250`.
w	*integer*	Width of the image in pixels, for example, `300`.

(*) The field is optional and may not be present in each response.

7.3.4 Native Asset Video Object

Table 11: Native Asset Video Object Properties

Value	Type	Description
vasttag	*string*	Vast XML, use the following example to format your VAST XML response.

```
<?xml version="1.0" encoding="UTF-8"?>
<VAST version="2.0">
   <Ad id="12345">
      <InLine>
         <AdSystem version="1.0">SpotXchange</AdSystem>
         <AdTitle><![CDATA[Sample VAST]]></AdTitle>
         <Impression>http://sample.com</Impression>
         <Description><![CDATA[A sample VAST feed]]></Description>
         <Creatives>
            <Creative sequence="1" id="1">
               <Linear>
                  <Duration>00:00:30</Duration>
                  <TrackingEvents />
                  <VideoClicks>
                     <ClickThrough><![CDATA[http://sample.com/openrt btest]]>
                     </ClickThrough>
                  </VideoClicks>
                  <MediaFiles>
                     <MediaFile delivery="progressive" bitrate="256"
                     width="640" height="480" type="video/mp4">
                     <![CDATA[http://sample.com/video.mp4]]>
                     </MediaFile>
                  </MediaFiles>
               </Linear>
            </Creative>
         </Creatives>
      </InLine>
   </Ad>
</VAST>
```

7.3.5 Native Asset Data Object

Table 12: Native Asset Data Object Properties

Value	Type	Description
value	*string*	The formatted string of data to be displayed. Can contain a formatted value such as "`5 stars`" or "`$10`" or "`3.4 stars out of 5`".

7.3.6 Native Link Object

Table 13: Native Link Object Properties

Value	Type	Description
url	*string*	Landing URL of the clickable link, for example, `"http://advertiser.com/"`
*clicktrackers**	*array of strings*	Click tracker URLs to be activated when the URL is clicked, for example, `["http://adserver.com/click?impid=102"]`

7.3.7 Native Response Example

```
{
  "seatbid":[
    {
      "bid":[
        {
          "adm_native":{
            "ver":"1.2",
            "jstracker":"<html></head><body></body><script src='./jquery.js'></script></
↪html>",
            "link":{
              "url":"http://adserver.com/click?impid=102"
            },
            "imptrackers":[
              "http://adserver.com/native?impid=102"
            ],
            "assets":[
              {
                "id":1,
                "required":1,
                "title":{
                  "text":"A test Native Ad"
                }
              }
            ]
          }
        }
      ]
    }
  ]
}
```

7.4 Bid Response JSON Examples

7.4.1 Banner Bid Response

```
{
  "id": "1234567890",
  "seatbid": [
    {
      "bid": [
        {
          "id": "1",
          "impid": "102",
          "price": 9.43,
          "crid": "314",
          "cid": "42",
          "language": "en",
          "burl":"https://adserver.com/imp?impid=102&winprice=${AUCTION_PRICE}",
          "adm": "<a href=\"http://adserver.com/click?adid=12345&tracker=${CLICK_
↪URL:URLENCODE}\"><img src=\"http://image1.cdn.com/impid=102\"/></a>",
          "iurl": "http://adserver.com/preview?crid=314",
          "adomain": [
            "advertiserdomain.com"
          ],
          "ext": {
            "advertiser_name": "Coca-Cola",
            "agency_name": "CC-advertising",
            "agency_id": "abcd1234"
          }
        }
      ],
      "seat": "4"
    }
  ]
}
```

7.4.2 Native Bid Response

```
{
  "id":"1234567890",
  "seatbid":[
    {
```

(continues on next page)

(continued from previous page)


```
      "bid":[
        {
          "id":"1",
          "impid":"102",
          "price":9.43,
          "crid":"314",
          "cid":"42",
          "language":"en",
          "burl":"https://adserver.com/imp?impid=102&winprice=${AUCTION_PRICE}",
          "adomain":[
            "advertiserdomain.com"
          ],
          "ext":{
            "advertiser_name":"Coca-Cola",
            "agency_name":"CC-advertising",
            "agency_id":"abcd1234"
          },
          "adm_native":{
            "ver":"1.2",
            "jstracker":"<html></head><body></body><script src='./jquery.js'></script></
↪html>",
            "privacy":"https://www.example.com/privacy-notice",
            "link":{
              "url":"http://adserver.com/click?impid=102"
            },
            "imptrackers":[
              "http://adserver.com/native?impid=102"
            ],
            "assets":[
              {
                "id":1,
                "required":1,
                "title":{
                  "text":"A test Native Ad"
                }
              }
            ]
          }
        }
      ],
      "seat":"58"
    }
  ]
}
```


7.4.3 Video Bid Response


```
{
  "cur":"USD",
  "id":"e9c3e120-ffcb-4300-9c98-644cb26f95df",
```


(continues on next page)

(continued from previous page)


```
  "seatbid":[
     {
        "bid":[
           {
              "crid":"3",
              "adm":"<?xml version=\"1.0\" encoding=\"UTF-8\"?><VAST version=\"2.0\"><Ad␣
↪id=\"e1081d52_a3d9353a3f5711e795201cdbeb920001\"><Wrapper><AdSystem>BidSwitch<\/
↪AdSystem><VASTAdTagURI><![CDATA[http:\/\/adsrv.com\/vast\/7drQU9ksr]]><\/VASTAdTagURI>
↪<Error><![CDATA[http:\/\/gce-sc.bidswitch.net\/vast_error\/gdmj4t2_3wJg\/]]><\/Error>
↪<Impression><![CDATA[http:\/\/gce-sc.bidswitch.net\/imp\/${AUCTION_PRICE}\/mj4t2_3wJg\/
↪]]><\/Impression><Creatives><\/Creatives><\/Wrapper><\/Ad><\/VAST>",
              "language": "en",
              "protocol":3,
              "burl":"https://adserver.com/imp?impid=102&winprice=${AUCTION_PRICE}",
              "adomain":[
                 "nokia.com"
              ],
              "cid":"11",
              "ext":{
                 "advertiser_name":"Nokia"
              },
              "id":"1c3ff810-3623-4b04-8396-9e7ca071cb72",
              "impid":"1",
              "price":4.079077199308326
           }
        ],
        "seat":"1"
     }
  ]
}
```


7.4.4 BidSwitch No Bid Reason


```
{
   "id":"1234567890",
   "seatbid":[

   ],
   "nbr":4
}
```


7.4.5 Audio Ad Response


```
{
  "cur":"USD",
  "id":"e9c3e120-ffcb-4300-9c98-644cb26f95df",
  "seatbid":[
```


(continues on next page)

(continued from previous page)


```
    {
       "bid":[
          {
             "crid":"3tre445",
             "adm":"<?xml version=\"1.0\" encoding=\"UTF-8\"?><VAST version=\"2.0\"><Ad␣
↪id=\"e1081d52_a3d9353a3f5711e795201cdbeb920001\"><Wrapper><AdSystem>BidSwitch<\/
↪AdSystem><VASTAdTagURI><![CDATA[http:\/\/adsrv.com\/vast\/7drQU9ksr]]><\/VASTAdTagURI>
↪<Error><![CDATA[http:\/\/gce-sc.bidswitch.net\/vast_error\/gdmj4t2_3wJg\/]]><\/Error>
↪<Impression><![CDATA[http:\/\/gce-sc.bidswitch.net\/imp\/${AUCTION_PRICE}\/mj4t2_3wJg\/
↪]]><\/Impression><Creatives><\/Creatives><\/Wrapper><\/Ad><\/VAST>",
             "burl":"https://adserver.com/imp?impid=102&winprice=${AUCTION_PRICE}",
             "adomain":[
                "example.com"
             ],
             "cid":"11",
             "language": "en",
             "ext":{
                "advertiser_name":"Nokia"
                "agency_name": "CC-advertising",
                "agency_id": "abcd1234"
             },
             "id":"1c3ff810-3623-4b04-8396-9e7ca071cb72",
             "impid":"1",
             "price":4.079077199308326
          }
       ],
       "seat":"1"
    }
  ]
}
```

CHAPTER 8

BIDSWITCH DATA CENTRES

Table 1: BidSwitch Data Centres

Geo	Data Centre	Hostname
US East	GCE Berkeley County, South Carolina	`us-east.bidswitch.net`
US West	Amazon California	`us-west.bidswitch.net`, for the IP range, search for `us-west-1` in the AWS list
APAC	GCE Northeastern Asia-Pacific	`apac-jp.bidswitch.net`
Europe	AWS Frankfurt	`eu.bidswitch.net`

8.1 IP Values

Google Cloud Engine (GCE) To find the actual range of IP Addresses, use the instructions on this page: Where can I find Compute Engine IP ranges?[15], and see also Google Cloud Platform zones[16].

Amazon Web Services (AWS) To find the actual IP range search for relevant region specified above within each DC, e.g `eu-central-1` in the following the list: https://ip-ranges.amazonaws.com/ip-ranges.json

[15] https://cloud.google.com/compute/docs/faq#where_can_i_find_product_name_short_ip_ranges
[16] https://cloud.google.com/compute/docs/regions-zones/

CHAPTER 9

ANOMALY DETECTION

BidSwitch processes massive volumes of data, and while doing so employs anomaly detection to spot deviations from the norm when applied to domains, publishers, user agent data, and other metrics. These deviations are then investigated to try and ascertain if any fraud is involved in generating these anomalies, and if so, learn to spot whatever new tricks are being used. The process of ensuring higher quality trading on the network, follows these three concepts.

Blocking Rules Basic rules that filter out the most obviously suspicious traffic based on factor such as the following:

- **External Services:** Sites with swinging traffic patterns are blocked, as this signals that traffic is being pumped through bots. This is based on reliable external traffic analysis providers such as similarweb.com[17]
- **Mismatching Data:** For example, if a user's data differs between the bid request and impression calls.
- **No Publisher Information:** All traffic with an empty `publisher_ID` field is blocked, see the Publisher Object[18] section for how to include this information. You can also check the BidSwitch Policies[19].
- **Sudden Spikes** If a surge in activity is seen in one particular parameter, such as certain UUIDs (universally unique identifier) displaying very high frequency usage.

Expert Blacklisting This is an ongoing process that involves people investigating and developing solutions to potential problems discovered on the network. Experts examine logs and data sets; then analyse these against expected results to identify bad actors, and if necessary blacklist them.

The Anomaly Detection Engine This is a sophisticated tool built on machine learning technology that processes the complete set of incoming bid logs. Every update to the Anomaly Detection Engine goes through an expert audit, and an extensive set of QA procedures.

[17] https://www.similarweb.com/
[18] https://protocol.bidswitch.com/standards/publisher-object.html#pub-obj
[19] http://www.bidswitch.com/policies/

Every bid request is filtered through the Anomaly Detection Engine, which is synced in real-time against a database of prior records. All bids are confirmed as acceptable before the bid request is sent to a Buyer. If a request fails these criteria a No Bid Reason is returned to the Supplier, see the *Supplier No Bid Reason* (page 104) section. This means that pre-bid fraud detection is in place.

Table 1: Common Ad Fraud Methods

Type	Description
Bots	Software that runs automated tasks repetitively, at a much higher rate than would be possible for a human alone. Usually designed to generate fake impressions or serve unseen ads in the background of a real users computer. This method can be difficult to detect and generally problematic because bots can be retargeted or even whitelisted as a real audience.
Ad Stacking	While a user only sees one ad, the publisher may be serving multiple ads, or 1x1 pixel ads simultaneously. These types of ads register as an impression but are never actually seen.
Spoof Sites	Sites built mainly for the purpose of serving ads. Spoof sites are often part of a larger network of sites created to avoid triggering suspicion around a single site collecting very large amounts of revenue. Spoof sites are becoming increasingly sophisticated and often include 1-2 layers of real content that could be considered worthy of real traffic.
Spoof Domain	Domains created to replicate premium, well known sites. Advertisers can be duped into thinking they are buying high quality inventory from a recognized site when they are not. This can also impact publishers who will appear to have more inventory than they do, decreasing their prices.

9.1 Benefits of Anomaly Detection

For BidSwitch

- A higher quality network is a more attractive place for Buyers and Suppliers to do business
- The size, scope, and historic nature of BidSwitch's data across many Buyers and Suppliers allows us to detect irregularities other fraud vendors cannot, placing BidSwitch as a market leader in fraud detection.

For Buyers

- Not buying low quality or fraudulent impressions
- Higher quality of bids received within QPS limits

For Suppliers

- Increased prices on real ads. With fraud in the system publishers will appear to have more inventory than they do, decreasing their prices on real inventory.
- Suppliers can take actions to clean up fraud coming from their inventory using the information sent in No Bid Responses, see the *Supplier No Bid Reason* (page 104) section.

CHAPTER 10

API AUTHORIZATION

To connect to BidSwitch using the API, use the following steps.

- Create an API user in your myBidSwitch UI account.
- Using this user's credentials, get your API token by making a `POST` request to the BidSwitch authorization endpoint.

10.1 Creating an API User

To create a user profile that can receive API Tokens, use the following steps:

1. From the BidSwitch UI, select *Users* → *Add User*
2. From the *User Role* dropdown menu, select *API Account*.

10.2 Getting an API Auth Token

To get your API token, make a `HTTP POST` request to the following URL. The response will contain your access token.

```
https://uauth.iponweb.com/oauth2/token/
```

The POST request must contain the following fields

```
grant_type=password
scope=service_id=api.bidswitch.com
username=<USERNAME> <!-- Your BidSwitch UI login -->
password=<PASSWORD> <!-- Your Bidswitch UI password -->
```

Authorization Response

```
{
  "token_type": "Bearer",
  "scope": "service_id=api.bidswitch.com",
  "access_token": "<your token>",
  "expires_in": 3600
}
```

Equivalent request via `curl` command:

```
$ curl --request POST --data "grant_type=password" --data "username=<username>" --data
↪"password=<password>" \
       --data "scope=service_id=api.bidswitch.com" https://uauth.iponweb.com/oauth2/
↪token/
```

CHAPTER 11

DISCREPANCY REPORTING

The Discrepancy Reporting API is used to upload a daily report about the previous day's activity so that differences between the numbers reported by BidSwitch and those of the reporting Buyer or Supplier can be monitored. Use the information in the following sections to set up your discrepancy monitoring with BidSwitch.

- Buyer Discrepancy Reporting API
- *Supplier Discrepancy Reporting API* (page 91)

11.1 Disputing a Discrepancy

The attached PDF outlines the Discrepancy Resolution[20] process.

[20] https://docs.bidswitch.com/_static/discrepancy-reporting.pdf

CHAPTER 12

GLOSSARY

Addressable TV While addressable TV is technically a form of *Programmatic TV*, its household-level and real-time targeting are major distinguishing factors. Advertisers can buy audiences instead of programs, similar to the programmatic display and video buying processes utilized on desktop and mobile devices.

Auto-Deal A bid request routed directly to a Buyer by a Supplier as part of a private deal relationship, see the Setting up Auto-Deals[21] section for more details.

BidStream This refers to all of the bid requests flowing through the system. Within BidSwitch, Buyers can configure their BidStream settings so that they focus on the types of bid requests they want to receive.

Broadcast Deal

Non-exclusive deal A bid request that is sent to all connected partners.

CPM (Cost-per-thousand) Media term describing the cost of 1,000 impressions.

Creative The media asset associated with an ad, such as an image or video file.

DAAST Digital Audio Ad Serving Template

Data Centre A data centre is a facility used to house computer systems and associated components, see the *BidSwitch Data Centres* (page 78) for a list of ours.

Day-part

Day-parting Day-parting means splitting the day into prime and non-prime time slots. The actual times vary depending on the Supplier and inventory type.

Deal ID The ID assigned to a programmatic ad transaction used by both the Buyer and Supplier to transact on prearranged terms, see the Private Marketplaces & Deal IDs[22] section.

[21] https://protocol.bidswitch.com/support/private-deals.html#auto-deals
[22] https://protocol.bidswitch.com/support/private-deals.html#pmp-deals

DOOH Digital out-of-home advertising or out-of-home media is advertising that reaches the consumers while they are outside their homes, for example, digital billboard displays, bus shelters, or telephone booth advertising.

Exclusive Deal

Deal Seats The seats (Buyers) allowed to bid on a bid request, as part of an Exclusive Deal.

Geotargeting

Geographic Targeting A method that enables advertisers to show an ad specifically to visitors based on zip code, area code, city, DMA, state, and/or country derived from user-declared registration information or inference-based mechanism.

Impression This is a single display of an online ad (creative) to a user's web-enabled device. Each time an advertisement loads into a user's screen, the ad server may count that as one impression. For more information on how impressions are tracked, see the Impression Delivery and Tracking[23] section.

Linear Television The traditional broadcast system in which a viewer watches a scheduled TV programme at the time it's broadcast, and on the channel it's originally presented on.

Nonlinear Television A non-traditional means of viewing TV content that enables "time-shifting" (not watching during the scheduled broadcast) using streaming, DVR, video-on-demand, over-the-top (OTT) or mobile TV technology.

Over-The-Top (OTT) The delivery of TV content via the Internet. Users are not required to subscribe to a traditional cable or satellite providers to watch TV programmes. Devices piggy back on an existing network.

Private Deal

PMP A Private Marketplace deal in which the Buyer and Supplier arrange a one-to-one deal. See the Private Marketplaces & Deal IDs[24] section for details.

Programmatic TV The use of Set-Top-Box data to inform the purchase process.

QPS Queries Per Second (QPS), see the *QPS Overview* (page 89) section for details.

SmartSwitch SmartSwitch optimises a Buyer's bidstream by analyzing bid request parameters, see the *SmartSwitch Overview* (page 87) section for details.

User Group Used to categorise users based on their perceived value to a Buyer, for more information see the SmartSwitch User Optimization[25] section.

User Syncing The process of matching Supplier cookie IDs with their matching Buyer cookie IDs, or vice versa. See the following sections for more details:

- *Supplier User Matching* (page 106) | *Supplier First-Party User-Matching* (page 110)
- Buyer User Matching[26]

VAST Video Ad Serving Template

[23] https://protocol.bidswitch.com/support/imp-tracking-markup-delivery.html#imp-track-deliver
[24] https://protocol.bidswitch.com/support/private-deals.html#pmp-deals
[25] https://protocol.bidswitch.com/support/smart-switching-user-groups.html#smart-switch
[26] https://protocol.bidswitch.com/features/user-matching.html#user-match

CHAPTER 13

SMARTSWITCH OVERVIEW

SmartSwitch optimises a Buyer's BidStream by analyzing bid request parameters such as *Country*, *Device*, *Native*, and others. Using these parameters and overlaying the data on impressions that were won/bid on, SmartSwitch is able to detect a Buyer's specific buying pattern and optimise the type of bidstream traffic it receives. For more information on the parameters used, see the BidSwitch Buyer Protocol v5.3[27] section.

SmartSwitch also uses u-Predict models to estimate the probability of a bid request sent to a Buyer receiving a winning bid. To learn more, see the u-Predict site[28].

By combining the traffic that matches both criteria: desired traffic with higher win probability, a Buyer receives a much higher quality of traffic as defined by their own buying preferences.

13.1 SmartSwitch Traffic

SmartSwitch sends the traffic that is most valuable to a Buyer, and that can be broken down into the following groups.

- **High Quality**: Traffic that the Buyer bids on and wins most often.

[27] https://protocol.bidswitch.com/standards/standards.html#bsw-proto
[28] http://www.iponweb.com/core-framework/u-predict/

- **Low Quality**: Traffic that the Buyer does not consider well priced, and for which it has a low win rate. A segment of traffic that is considered low quality is sent to verify SmartSwitch assumptions.

13.2 SmartSwitch and QPS

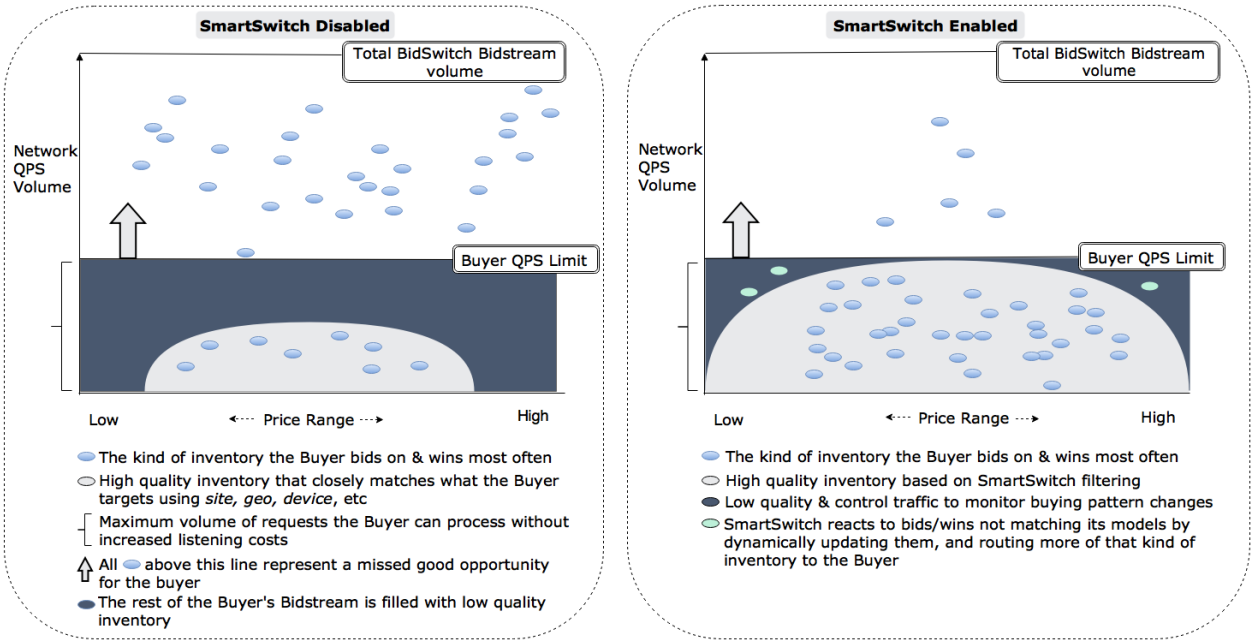
Listening to the the full BidSwitch bidstream would overwhelm most Buyers and result in overloading a Buyer's servers. As well as incurring increased costs just to be able to listen to more traffic, without SmartSwitch Buyers would also listen to a lot more random traffic thus reducing the quality.

The goal of SmartSwitch is to ensure that the best bids reach a Buyer within its QPS settings. For more information about QPS, see the *QPS Overview* (page 89) section. Use the following table and charts to understand the benefits of SmartSwitch.

13.2.1 SmartSwitch Buyer Benefits

Table 1: SmartSwitch Features and Benefits

SmartSwitch Disabled	SmartSwitch Enabled
Random inventory is sent to the Buyer based on a subset of the total bidstream	Optimised inventory is sent to the Buyer, which closely aligns with their buying pattern
Lower relative levels of desired inventory available to the Buyer versus the amount of traffic listened to	Increased levels of desired inventory sent to the Buyer within their listening capacity
Lower bid/win ratio due to inventory not matching the Buyer's bidding parameters as closely as possible	Higher bid/win ratio because traffic matches the Buyer's bidding strategy
Lower Buyer ROI on servers and other infrastructure costs	Better ROI due to increased efficiency of infrastructure usage
	Machine learning quickly responds to any changes in a Buyer's buying pattern, and realigns bids sent to match the Buyers wishes
	Any new inventory will be immediately routed to a Buyer if considered optimal



13.2.2 SmartSwitch Supplier Benefits

- Routes traffic to Buyers most interested in the inventory type you're selling.
- Increases the amount of interested Buyers potentially bidding on each impression, and with an increased number of bidders, theoretically a higher clearing price.
- Screens all Bid Requests and returns a No Bid Reason should some traffic be flagged as potentially fraudulent, enabling Suppliers to clean up their network of possible bad actors, see the *Supplier No Bid Reason* (page 104) and *Anomaly Detection* (page 79) sections for more details.

13.3 QPS Overview

QPS (Queries Per second) in BidSwitch refers to the maximum absolute limit placed on traffic received from a Supplier or sent to a Buyer per data centre.

Note: The QPS settings do not set the actual amount of traffic that BidSwitch sends to a Buyer. It is the maximum traffic that a Buyer could receive, and BidSwitch will not send more than the maximum QPS value.

Traffic Composition Buyers receive two types of traffic from BidSwitch: *biddable traffic* and *learning traffic*.

The *learning* traffic is sent to detect changes in the Buyer's buying preferences. If a Buyer starts buying more from a specific traffic segment, then BidSwitch sends more traffic from that segment. This also works vice-versa, if a Buyer stops buying from a particular traffic

segment, BidSwitch sends less traffic of that type. For more detailed information about traffic types, see the *SmartSwitch Overview* (page 87) section.

Influencing Traffic Volumes The real influence on traffic volumes sent to a Buyer is the amount and type of traffic that the Buyer buys. The more traffic that a Buyer buys in a particular segment, the more of that traffic will be sent to focus bidding on what is desired, see the *SmartSwitch Overview* (page 87) section for more information.

Important: It is highly recommended that you set the maximum QPS value in accordance with the real capacity of your servers. Otherwise, if you start buying in large volumes, BidSwitch will start sending the maximum amount of QPS which could overload your servers.

Further Reading You can find a number of case studies and FAQs about SmartSwitch in the SmartSwitch PDFs[29] section.

[29] https://protocol.bidswitch.com/support/pdf-downloads.html#ss-pdfs

CHAPTER 14

SUPPLIER DISCREPANCY API

The Discrepancy Reporting API is used to upload a daily report about the previous day's activity so that differences between the numbers reported by BidSwitch and the reporting Supplier can be monitored. Use the information in the following sections to set up your discrepancy monitoring with BidSwitch.

14.1 Process Overview

1. Get your API Access Token. This requires creating an API user in the myBidSwitch UI, see the *Getting API Credentials* (page 92) section for details.
2. Using this Token, make a `HTTP POST` request to the BidSwitch discrepancy endpoint. The post request should contain your platform type and partner ID in the URL, and be in JSON. See the *Uploading a Discrepancy Report* (page 93) section for details.

Note:

- If you upload several reports during one day, only the last one uploaded is going to be processed. So if you need to do this, use the Several Days report format.
- An *Uploading Script* (page 94) is available.

14.2 Getting API Credentials

To get access credentials for the Discrepancy Reporting API, you need to create an API User Account in the myBidSwitch UI, using the following information.

14.2.1 Creating an API User

To create a user profile that can receive API Tokens, use the following steps:

1. From the BidSwitch UI, select *Users* → *Add User*
2. From the *User Role* dropdown menu, select *API Account*.

14.2.2 Getting an API Auth Token

To get your API token, make a `HTTP POST` request to the following URL. The response will contain your access token.

```
https://uauth.iponweb.com/oauth2/token/
```

The POST request must contain the following fields

```
grant_type=password
scope=service_id=api.bidswitch.com
username=<USERNAME> <!-- Your BidSwitch UI login -->
password=<PASSWORD> <!-- Your Bidswitch UI password -->
```

Authorization Response

```
{
  "token_type": "Bearer",
  "scope": "service_id=api.bidswitch.com",
  "access_token": "<your token>",
  "expires_in": 3600
}
```

Equivalent request via `curl` command:

```
$ curl --request POST --data "grant_type=password" --data "username=<username>" --data
↪"password=<password>" \
       --data "scope=service_id=api.bidswitch.com" https://uauth.iponweb.com/oauth2/
↪token/
```

Note: API Authorization using `https://api.bidswitch.com/discrepancy-check/v1.0/login` is deprecated. Updated authorization steps are described in the *API Authorization* (page 82) section. The *Uploading Script* (page 94) is updated accordingly.

14.3 Uploading a Discrepancy Report

To upload your discrepancy report, use the following information.

- Make a `HTTP POST` request to the following appropriate link
- Specify the endpoint using only lowercase
- Ensure you include the HTTP headers with the required credentials and information

```
<!-- Supplier Endpoint All Lowercase -->
https://api.bidswitch.com/discrepancy-check/v1.0/ssp/<ssp-name>/upload-report/
```

Report Upload HTTP Headers

```
'Authorization':'Bearer <your-token>'
'Accept':'application/json'
'Content-Type':'application/json'
```

Example of uploading by curl:

```
# Syntax
$ curl -H "Accept:application/json" -H 'Authorization:Bearer <Your token here>' \
       -H 'Content-Type:application/json' -d @report.json \
       https://api.bidswitch.com/discrepancy-check/v1.0/ssp/<'ssp-name'>/upload-report/

# Supplier Example
$ curl -H "Accept:application/json" -H 'Authorization:Bearer CI6IkpXVCJ9.
↪eyJhdWQiOiJwdWJsaWNfY2xpZW50IiwiaXNzIjoidWF1dGgiLCJqdGkiOiJORlhvblRRSyIsImtsc2RmYXNkZmthZG1mbDthc2RuZm
↪' \
       -H 'Content-Type:application/json' -d @report.json \
       https://api.bidswitch.com/discrepancy-check/v1.0/ssp/abc/upload-report/
```

Example Response:

```
{
    "status": "success",
    "handled": [["2016-08-06", "2016-08-14", "2016-08-07"]]
}
```

14.4 Uploading Script

Using the Uploading Script

Pass the following arguments when calling the script from the command-line. You can download the `Uploading Script here`.

- `platform_type`: `ssp`
- `platform_name`: *SSP Name*
- `data.json`: the JSON report

```
# Syntax Example
$ upload.py -u <username>:<password> -p <platform_type>:<platform_name> -d <data.json>

# Supplier Upload Example
$ python upload.py -u ssp_dummy:123456 -p ssp:adrtbdummy -d ./yesterday.json
```

14.5 Upload Report Format

The post request requires the following data in JSON. Usually you need to upload the full report from the previous day, which contains the complete data.

14.5.1 Reports

Table 1: Reports Object

Field	Type	Validator	Description
reports	*Array of objects*	Required	Array of reports. The reports are created using the report object format, see the *Report Object* (page 95) for more details.

14.5.2 Report Object

Table 2: Report Object Parameters

Field	Type	Validator	Description
seat	*String*	Optional (Suppliers Only)	When a Supplier wants to check for discrepancies with a particular Buyer, they can specify the Buyer name using this field. **Note:** This field is only used when uploading a Supplier report, see the *Supplier Reports* (page 100) example.
timezone	*String*	Required	Sets the timezone of the report, the valid options being those in the `pytz.all_timezones` list.
currency	*String*	Required	Sets the currency used in the report, the valid options being one of the following: `[ 'USD', 'EUR', 'JPY']`.
data	*Array of objects*	Required	Specifies the date to which the report figures refer, and the report data values. See the *Report Data Values Object* (page 95) for more details. • Keys: date using the following format `"%Y-%m-%d"` • Values: list of the *Report Data Values Object* (page 95)

14.5.3 Report Data Values Object

Table 3: Report Data Values Object

Field	Type	Validator	Description
hour	*String*	Optional	Value is from range 00-23. If this field is absent, it is assumed that the report is for the full day.
imps	*Integer*	Required	The total number of impressions.
cost	*Float*	Required	The final cost for delivered impressions.
bids	*Integer*	Optional.	Field for Supplier reports. The total number of bid requests sent by the Supplier to BidSwitch. This includes all bid types such as requests ending in an error or timeout, no bids, yes bids, and incorrect responses.
timeouts	*Integer*	Optional.	Field for Supplier reports. The number of bid requests ending in a timeout.

14.6 Example Reports

Zero Reports (page 96) | *Hourly Report* (page 96) | *Supplier Reports* (page 100) | *Minimal report* (page 101) | *Several days* (page 101)

14.6.1 Zero Reports

It must contain `{"imps":0,"cost":0}`, which means there was no trading. If you don't have data for reporting just skip sending reports for that day.

```
{
   "reports":[
      {
         "data":{
            "2016-08-05":[
               {
                  "imps":0,
                  "cost":0
               }
            ]
         },
         "currency":"USD",
         "timezone":"UTC"
      }
   ]
}
```

14.6.2 Hourly Report

This is considered the best practice reporting format, which splits reports by hours and contains bids, timeouts, imps, and cost.

Note: `"hour"` is represented as a string and must contain 2 digits.

```
{
   "reports" : [
      {
         "timezone" : "UTC",
         "currency" : "USD",
         "data" : {
            "2016-08-05" : [
               {
                    "timeouts": 26887,
                    "imps": 895,
                    "bids": 419746,
                    "cost": 16.6,
                    "hour": "00"
               },
               {
                    "timeouts": 292712,
                    "imps": 1442,
                    "bids": 496269,
                    "cost": 32.59,
                    "hour": "01"
```

(continues on next page)

(continued from previous page)


```
        },
        {
            "timeouts": 523600,
            "imps": 736,
            "bids": 264686,
            "cost": 8.74,
            "hour": "02"
        },
        {
            "timeouts": 431437,
            "imps": 299,
            "bids": 346682,
            "cost": 16.93,
            "hour": "03"
        },
        {
            "timeouts": 207841,
            "imps": 778,
            "bids": 104816,
            "cost": 10.71,
            "hour": "04"
        },
        {
            "timeouts": 510346,
            "imps": 398,
            "bids": 538113,
            "cost": 41.47,
            "hour": "05"
        },
        {
            "timeouts": 475965,
            "imps": 2320,
            "bids": 259689,
            "cost": 40.1,
            "hour": "06"
        },
        {
            "timeouts": 226502,
            "imps": 2242,
            "bids": 522407,
            "cost": 8.9,
            "hour": "07"
        },
        {
            "timeouts": 117024,
            "imps": 464,
            "bids": 91400,
            "cost": 28.71,
            "hour": "08"
        },
        {
```


(continues on next page)

(continued from previous page)


```
                "timeouts": 434858,
                "imps": 1142,
                "bids": 179587,
                "cost": 32.67,
                "hour": "09"
            },
            {
                "timeouts": 43211,
                "imps": 2223,
                "bids": 276447,
                "cost": 16.61,
                "hour": "10"
            },
            {
                "timeouts": 310415,
                "imps": 1649,
                "bids": 51621,
                "cost": 12.66,
                "hour": "11"
            },
            {
                "timeouts": 430663,
                "imps": 249,
                "bids": 474692,
                "cost": 36.81,
                "hour": "12"
            },
            {
                "timeouts": 310470,
                "imps": 774,
                "bids": 222399,
                "cost": 1.24,
                "hour": "13"
            },
            {
                "timeouts": 427396,
                "imps": 1706,
                "bids": 144780,
                "cost": 4.2,
                "hour": "14"
            },
            {
                "timeouts": 290821,
                "imps": 1492,
                "bids": 121281,
                "cost": 37.24,
                "hour": "15"
            },
            {
                "timeouts": 45520,
                "imps": 1209,
```


(continues on next page)

(continued from previous page)


```
            "bids": 3909,
            "cost": 16.51,
            "hour": "16"
        },
        {
            "timeouts": 232909,
            "imps": 2184,
            "bids": 55060,
            "cost": 43.22,
            "hour": "17"
        },
        {
            "timeouts": 504859,
            "imps": 1145,
            "bids": 221379,
            "cost": 41.09,
            "hour": "18"
        },
        {
            "timeouts": 327505,
            "imps": 11,
            "bids": 138869,
            "cost": 37.56,
            "hour": "19"
        },
        {
            "timeouts": 512344,
            "imps": 1360,
            "bids": 492131,
            "cost": 30.65,
            "hour": "20"
        },
        {
            "timeouts": 315876,
            "imps": 552,
            "bids": 461330,
            "cost": 32.76,
            "hour": "21"
        },
        {
            "timeouts": 258491,
            "imps": 722,
            "bids": 341752,
            "cost": 8.0,
            "hour": "22"
        },
        {
            "timeouts": 399477,
            "imps": 2263,
            "bids": 515451,
            "cost": 32.97,
```


(continues on next page)

(continued from previous page)

```
                    "hour": "23"
                }
            ]
        }
    }
  ]
}
```

14.6.3 Supplier Reports

You may use the `"seat"` field to split your report by Buyer.

```
{
   "reports":[
      {
         "data":{
            "2015-11-27":[
               {
                  "timeouts":24450,
                  "bids":1943830,
                  "hour":"00",
                  "imps":83922,
                  "cost":3234
               },
               {
                  "hour":"23",
                  "imps":83922,
                  "cost":3234,
                  "timeouts":24450,
                  "bids":1943830
               }
            ]
         },
         "currency":"JPY",
         "timezone":"US/Pacific",
         "seat":"123"
      },
      {
         "currency":"JPY",
         "data":{
            "2015-11-27":[
               {
                  "cost":3234,
                  "imps":83922,
                  "hour":"00",
                  "bids":1943830,
                  "timeouts":24450
               },
               {
                  "timeouts":24450,
```

(continues on next page)

(continued from previous page)

```
                "bids":1943830,
                "hour":"23",
                "imps":83922,
                "cost":3234
            }
         ]
      },
      "timezone":"EST",
      "seat":"456"
   }
]
}
```

14.6.4 Minimal report

```
{
   "reports":[
      {
         "timezone":"UTC",
         "currency":"USD",
         "data":{
            "2015-11-26":[
               {
                  "imps":83922,
                  "cost":3234
               }
            ]
         }
      }
   ]
}
```

14.6.5 Several days

If you missed uploading reports for several days, you can upload several days worth of reports at once.

```
{
   "reports":[
      {
         "timezone":"UTC",
         "currency":"USD",
         "data":{
            "2015-11-26":[
               {
                  "imps":83922,
                  "cost":3234.22
               }
```

(continues on next page)

(continued from previous page)

```
            ],
            "2015-11-27":[
               {
                  "imps":93922,
                  "cost":823.42
               }
            ],
            "2015-11-28":[
               {
                  "imps":92,
                  "cost":6.223
               }
            ]
         }
      }
   ]
}
```

14.7 Upload Response

If you upload successfully, you will receive a 200 OK response with a dated results list as shown in the following example. If your upload is in the wrong format, you will get an error status.

Successful Response

```
{
   "handled":[
      [
         "123 2015-11-27",
         "123 2015-11-26"
      ],
      [
         "456 2015-11-26"
      ]
   ],
   "status":"success"
}
```

Error Response

```
{
   "errors":[
      [
         "<index of report>",
         "<error text>"
```

(continues on next page)

(continued from previous page)

```
      ]
   ],
   "status":"error"
}
```

Other Possible Responses

You may also receive one of the following HTTP status codes:

- `401 Unauthorized`: If you don't have access to the Buyer or Supplier details, or you don't have the permission to upload, ensure that you have configured your account to use an API Token. You need an API account to upload reports.
- `404 Not Found`: If the `<ssp-name>` is not found in our system.
- `500 Internal Server Error`: If another unexpected error occurred.

CHAPTER 15

SUPPLIER NO BID REASON

Suppliers can receive two kinds of No Bid Responses (NBRs) from BidSwitch.

- A Buyer No Bid Response, which is usually an empty `204` response indicates that only one particular Buyer did not bid on the offer. See the Buyer No Bid Response[30] section for an example of this kind of response.
- A BidSwitch No Bid Reason, which contains the `nbr` field indicates that the particular Bid Request was not forwarded to any Buyers due to it failing impression anomaly checks.
- The No Bid Reasons sent by BidSwitch conform with those outlined in the Open RTB Specification[31], sections 5.22 and 7.1.

A BidSwitch No Bid Reason can occur for a number of different reasons, including suspected fraudulent activity like bot traffic or traffic coming from a known bad actor IP, site/publisher blocking, or invalid request elements. BidSwitch will continue to add more nuanced NBRs as time goes on.

By returning this information to Suppliers, it allows them to address potential problems in their system. If a No Bid Reason is triggered, BidSwitch responds with a `HTTP 200` response and one of the following codes in the `nbr` field depending on what has been identified as the problem. For more information you can also download the `No Bid Reason Fact Sheet`.

[30] https://protocol.bidswitch.com/standards/bid-response-examples.html#no-bid-resp
[31] https://www.iab.com/wp-content/uploads/2016/03/OpenRTB-API-Specification-Version-2-4-FINAL.pdf

Table 1: BidSwitch No Bid Reasons

Code	OpenRTB Description	BidSwitch Usage
0	Unknown Error	Not used
1	Technical Error	Not used
2	Invalid request	Bid requests that are treated as bad by BidSwitch, except for in-app requests with no IFA, which use #8
3	Known Web Spider	Not used
4	Suspected non-human traffic	The bid request was identified to be from a bot or other non-human activity
5	Cloud, Data centre, or Proxy IP	The bid request was blocked because of its IP
6	Unsupported Device	Not used
7	Blocked Publisher or Site	The Publisher ID, Site ID, Site Domain, App Domain, App Bundle, or App Name have been blocked in BidSwitch
8	Unmatched User	In-app bid requests without an IFA. This does not apply to cookie-less website bid requests.

15.1 Example Response

```
{
   "id":"1234567890",
   "seatbid":[

   ],
   "nbr":4
}
```

15.2 Enabling No Bid Reason

- To receive a No Bid Reason, you need to support the `nbr` field in bid responses.
- To enable No Bid Reasons on your BidSwitch account, contact support@bidswitch.com.

CHAPTER 16

SUPPLIER USER MATCHING

As the Supplier and BidSwitch sit between the end-user and the Buyer, user matching is important as it allows a Buyer to know the potential value of a user, and thus make calculated decisions to serve the most appropriate advertisements. Additionally, it can increase CPM and media spend to Suppliers . It is therefore in both parties best interests to sync their cookies appropriately. Use the information int he following sections to get the most out of user matching.

16.1 Bid Request Fields

User matching, or cookie syncing, is the process of matching an Supplier cookie ID to a Buyer cookie ID. BidSwitch has an integrated user matching functionality to facilitate this between Buyers and Suppliers .

When receiving a bid request from a Supplier, BidSwitch expects both the BidSwitch User ID and Supplier User ID to be sent in the `buyeruid` and `id` fields of the User Object[32].

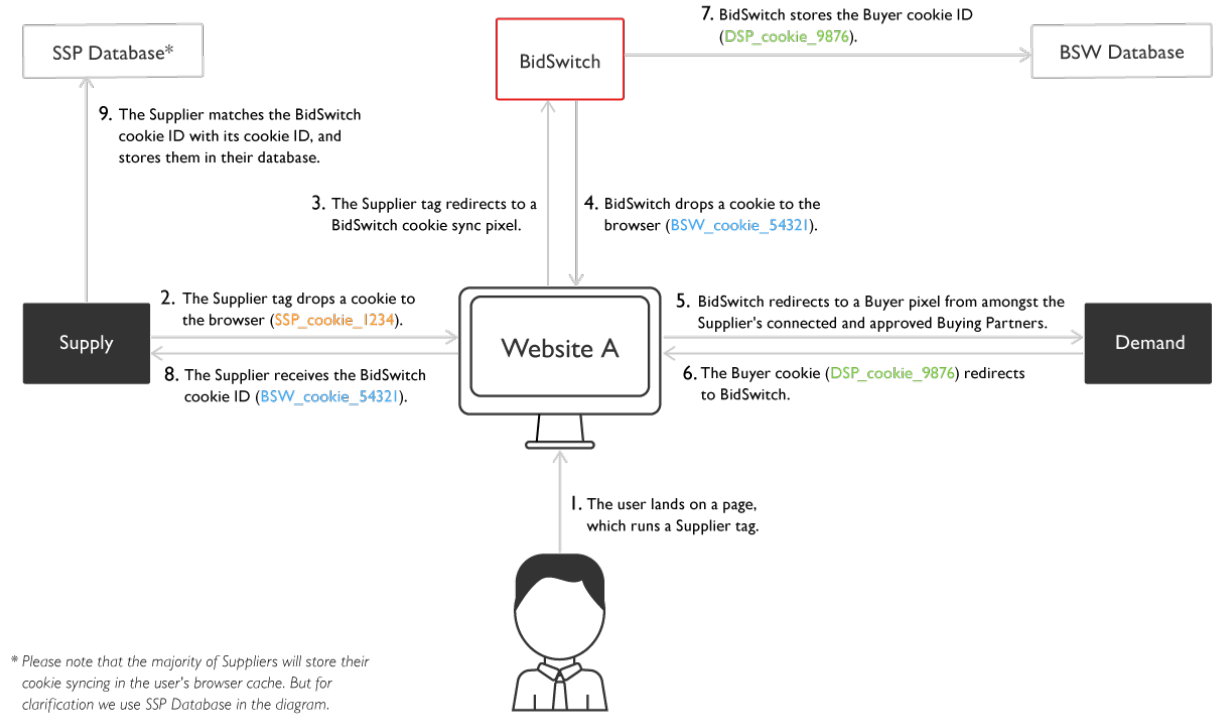
[32] https://protocol.bidswitch.com/standards/user-object.html#user-obj

Table 1: Supplier Buyer User Matching

Supplier to BidSwitch Bid Request	BidSwitch to Buyer Bid Request

<pre>{
"user":{
"id":"ssp-cookie-1234",
"buyeruid":"bsw-cookie-54321",
}
}</pre>
	<pre>{
	"user":{
	"id":"bsw-cookie-54321",
	"buyeruid":"DSP-cookie-5678",
	}
	}</pre>

16.2 Supplier User Matching Flow



Process

1. The user lands on a page, which runs a Supplier tag
2. The Supplier tag runs on the page and drops a cookie to the user browser (e.g. `SSP_cookie_1234`)
3. The Supplier tag redirects to a BidSwitch cookie sync pixel
4. BidSwitch drops a cookie to the users browser (e.g. `BSW_cookie_54321`)
5. BidSwitch redirects to a randomly selected Buyer pixel from amongst the Buyers listening to the Supplier's traffic

6. The Buyer drops its cookie (e.g. `DSP_cookie_9876`) and redirects to BidSwitch with its cookie
7. BidSwitch stores the Buyer cookie ID (`DSP_cookie_9876`)
8. The Supplier receives the BidSwitch cookie ID (`BSW_cookie_54321`)
9. The Supplier matches the BidSwitch cookie ID with its cookie ID, and stores them in their database.

Note: Please note that the majority of Suppliers will store the cookie syncing in the user's browser cache. But for clarification we use the database in the diagram.

When all parties have synced their cookies, bid requests from the Supplier to BidSwitch will contain the BidSwitch cookie ID. BidSwitch will then match and retrieve the Buyer cookie ID from the BidSwitch database and send it to the correct Buyer. This enables the Buyer to use their cookie ID and bid intelligently.

16.3 Supplier Initiated User Sync

Suppliers should sync users using the BidSwitch user sync URL. Once synced, BidSwitch redirects the user to the Supplier's sync URL, this endpoint is completely up to the Supplier and you can check what link BidSwitch is syncing with on the UI *Settings* page.

Supplier Initiated Cookie Sync Endpoint

```
## Syntax
http://x.bidswitch.net/sync?ssp=$SSP_NAME&user_id=$SSP_UID

## Example
http://x.bidswitch.net/sync?ssp=sspname&user_id=user12345
```

Note: Fields marked with asterisk (*) are optional.

Table 2: HTTP Request Parameters

Value	Type	Description
ssp_id	*string*	The ID assigned to the Supplier by BidSwitch, for example, `sspname`
*user_id**	*string*	The Supplier user ID. This is the value normally passed in the `user.id` field in OpenRTB bid requests. **Note:** This parameter is highly recommended for browsers that do not support 3rd party cookies, e.g. Safari and IOS_App

16.4 Cookie Syncing Best Practices

While it is also the responsibility of Buyers to play an active part in the cookie syncing process, BidSwitch strongly recommends the following Supplier practices to maximise ROI.

- If there is any cookie logic, ensure that BidSwitch is set with a high priority.
- Aim to cookie sync user data with BidSwitch once per day.
- Set the BidSwitch cookie expiration date to more than 30 days. Ideally closer to 365 days. This will prevent BidSwitch from storing a lot of expired user IDs and increase the sync rate with the Supplier.

16.5 BidSwitch Initiated User Sync

When BidSwitch initiates a user sync with a Supplier, the process flows in the following manner:

1. The user lands on a page, which runs a Buyer tag
2. Running this Buyer tag drops a cookie to the user browser e.g. `DSP_cookie_9876`, which redirects to BidSwitch
3. BidSwitch stores the Buyer cookie ID, e.g. `DSP_cookie_9876`
4. BidSwitch then redirects to a sync pixel belonging to a Supplier connected to the Buyer
5. The Supplier receives the BidSwitch cookie ID for this user, e.g. `BSW_cookie_54321`
6. The Supplier matches the BidSwitch cookie ID with its cookie ID, and stores them in their database

CHAPTER 17

SUPPLIER FIRST-PARTY USER-MATCHING

In the standard BidSwitch supplier user matching scenario, when sending bid requests to Buyers, BidSwitch uses the cookie ID stored in the browser cookie to map the Buyer user IDs for this user.

One of the limitations of this process is that certain platforms, such as Safari, do not support 3rd party user syncing, which prevents BidSwitch from dropping a cookie to a user's browser. To improve user syncing on such platforms, BidSwitch offers two solutions.

- *First-Party Cookie Syncing* (page 110)
- *Direct-like User Syncing* (page 111)

Note: Suppliers only need to implement one of these options, not both.

17.1 First-Party Cookie Syncing

First-party cookie syncing means that suppliers pass their cookie syncing events to BidSwitch while they have access to the browser as a first party, thereby matching users through events such as clicks for those receiving the supplier's cookie for the first time.

To do this, direct the user's browser to the following BidSwitch endpoint, where `${landing_page}` is the URL-encoded URL of the page the user will be redirected to.

```
# http endpoint
http://x.bidswitch.net/clk?lp=${landing_page}

# https endpoint
https://x.bidswitch.net/clk?lp=${landing_page}

# Example encoded URL
https://x.bidswitch.net/clk?lp=https%3A%2F%2Fmy.site.com%3Fmypage.html
```

17.2 Direct-like User Syncing

Direct-like user syncing allows Buyers and suppliers to match their `UUIDs` directly, bypassing the BidSwitch User ID in the middle.

Once suppliers know the `UUIDs` used by their buying partners on BidSwitch, then this information can be included in the bid request to enable Buyers make better bid pricing decisions.

- When BidSwitch receives a supplier initiated sync request with a Buyer `UUID`, it analyses it and tries to drop a cookie to the user's browser.
- If this fails, BidSwitch redirects the sync request to the Buyer, enabling the direct sync process, and returns the Buyer's matching `UUID` to the Supplier.
- Once the supplier knows the matching Buyer `UUIDs` for their user IDs, they should include this information in the bid request. BidSwitch then sends it to Buyers so they can make more informed bid pricing decisions.

For Suppliers to enable this feature, a number of changes need to be made to their integration with BidSwitch.

Note: If a supplier does not have a large pool of Buyer `UUIDs` to sync directly, in a limited 3rd party cookie environment like Safari, then supporting this functionality will bring limited value.

17.2.1 URL Parameters

For suppliers to support this new functionality, they need to ensure that they support these additional parameters in their sync endpoint URL.

- `bsw_uuid` : Sets the BidSwitch User ID. When using the direct user syncing method this field will always be empty.
- `dsp_uuid` : Sets the Buyer's User ID of the user being synced
- `dsp_id` : Sets the Buyer ID with whom the supplier is syncing

```
some.ssp.com/sync?bsw_uuid=${UUID}&dsp_uuid=${DSP_UUID}&dsp_id=${DSP_ID}
```

Once these parameters are supported, suppliers should contact support@bidswitch.com to have this activated on their connection, and to have their user sync URL updated in their myBidSwitch UI.

17.2.2 Supplier Initiated Direct Syncing

Once these changes are complete, suppliers must include the user ID they wish to sync with BidSwitch in the user sync URL, as in the following example.

```
x.bidswitch.net/sync?ssp=somessp&user_id=XYZ
```

BidSwitch will then sync with one of its connected Buyers and respond to the sync request with the `${DSP_ID}` and `${DSP_UUID}` information. Sending this to the supplier's user sync URL.

```
some.ssp.com/sync?bsw_uuid=&dsp_uuid=ABC&dsp_id=77
```

Suppliers should store this Buyer-related data so that they can forward it to BidSwitch during bid requests.

Note: BidSwitch initiated user syncs work in the same manner and sync data to the Suppliers User Sync URL.

17.2.3 Bid Request JSON Changes

BidSwitch expects the `bid_request.ext.dsp_uuids` field in the bid request JSON.

```
{
   "ext":{
      "dsp_uuids":{
         "dsp1_id":"dsp1_uuid",
         "dsp1_id":"dsp2_uuid",
      }
   }
}
```

17.2.4 Example Bid Request With UUID Sync

If the Supplier does not have a match with BidSwitch for a certain user, then the `user.buyeruid` field is left empty, and the direct syncing approach shall be used to sync the Buyer `UUIDs`. In this scenario, the Supplier should enrich the bid request with the Buyer UUIDs for that user, using the following examples.

Step 1

During syncing, BidSwitch will have returned the Buyer `UUIDs` to the user sync URL, as in the following example.

```
some.ssp.com/sync?bsw_uuid=&dsp_uuid=XYZ&dsp_id=77

some.ssp.com/sync?bsw_uuid=&dsp_uuid=ABC&dsp_id=41

some.ssp.com/sync?bsw_uuid=&dsp_uuid=IKL&dsp_id=42
```

Step 2

When sending the bid request to BidSwitch, the Supplier should include the UUIDs that match their user in `ext` section. See lines `12-13` that specify the Supplier's `UUID`, and lines `36-40` that match this user directly to the given Buyers.

```
{
 "cur": [
   "USD"
 ],
 "device": {
   "devicetype": 2,
   "language": "en",
   "ua": "Mozilla\/5.0 (Windows NT 6.3; WOW64) AppleWebKit\/537.36 (KHTML, like Gecko)␣
↪Chrome\/52.0.2743.116 Safari\/537.36",
   "ip": "24.68.173.116"
 },
 "at": 2,
 "user": {
   "id": "53C0C0D66E6521B101749B59F00222A8"
 },
 "site": {
   "publisher": {
     "id": "123"
   },
   "page": "http:\/\/www.somepage.com\/",
   "id": "123"
 },
 "imp": [
   {
     "banner": {
       "api": [
         3,
         4,
         5
       ],
       "h": 90,
       "w": 728
     },
     "id": "1"
   }
 ],
 "ext" : {
   "dsp_uuids" : {
      "77" : "XYZ",
      "42" : "IKL",
      "41" : "ABC"
   }
 },
 "id": "reqid"
}
```

INDEX