Data Router

File Publishing and Delivery API

Background

The Data Router (DR) exposes an HTTPS interface to allow a publisher to post and retract files. The DR delivers files and retractions to subscribers using a similar HTTPS interface, acting as an HTTPS client.

The fundamental operation of the data router is simple: for each valid incoming HTTPS request from a publisher, the DR attempts to send a similar request to each of the endpoints subscribed to the feed specified in the incoming request. From the DR's point of view, an incoming request from a publisher means nothing more than that the DR should transmit the request—modified according to well-defined rules—to each of the subscribing endpoints. This document describes—sometimes in tedious detail—the constraints that the DR places on incoming requests and the characteristics of the requests that the DR will send to subscribing endpoints.

Note on terminology: In some high-level descriptions of the DR, there is a notion that a feed can have multiple versions. A new version typically indicates that the payload or metadata for the feed has changed in some way. In this document, the term "feed" refers to a single version of a feed.

File Publishing

A publisher posts a file to a feed by making an HTTP PUT request to a URL constructed by appending a path segment containing a publisher-generated file ID to the feed's publishing URL. The body of the request contains the data being published. A publisher retracts a file by making an HTTP DELETE request to the URL previously used to publish the file.

A valid PUT or DELETE request for a file publishing operation has the following characteristics:

- The Request-URI is an absolute path consisting of the path component of the feed publishing URL followed by a "/" followed by the file ID for the file being published or retracted. The file ID must conform to the syntax for a non-zero-length path segment as defined in RFC 3986, section

[1] Feed publishing URLs are assigned during the feed creation process, described in *Data Router Provisioning API*,

[2] A PUT request can be sent with no body. This will result in the DR making delivery PUT requests to the feed's subscribers, also with no body.

[3] The DR does not require that a DELETE be preceded by a PUT to the same URL.

3.3. The Request -URI may contain a query string and/or a fragment. The DR does not interpret query strings or fragments but will include them in delivery requests, allowing a feed to exploit these HTTP features if they are useful for the feed's application.

- The Host header contains the host name and port address (if one is present) from the feed publishing URL.
- The Authorization header specifies basic HTTP authentication and includes the publisher's user ID and password encoded as specified in RFC 2617, section 2. Note that the publisher should include this in the initial request.
- The X-ATT-DR-META header contains the metadata for the file, expressed as a JSON object. The header value must be a well-formed JSON string that conforms to the restrictions in Appendix 1 and that is no longer than 4096 bytes.

The DELETE request has no body. The PUT request carries a body along with file metadata. A valid PUT request has the following additional characteristics:

- The Content-Type header indicates the type of the body. The DR places no restriction on this, and will use the same Content-Type when delivering the file to a subscriber.
- The request must contain either a Content-Length header indicating the size of the body or must use the chunked transfer encoding, signaled with a Transfer-Encoding header with a value of "chunked".
- The request must not apply a content coding to the body.
- The request may use the Expect header with a value of "100-continue" to signal that the client will wait for a response with a status code of 100 before sending the request body[7].

The publisher transmits a PUT or DELETE request via HTTPS to the host and port address specified in the authority component of the feed publishing URL. If no port is explicitly specified in the URL, the request is directed to the default HTTPS port (443).

The DR will use a server certificate signed by VeriSign and issued under AT&T Services, Inc.[8]. A publisher making a request to the DR needs to be configured to recognize this certificate authority.

If the request included an Expect header with a value of "100-continue", the DR will send a response with a status code of 100 after the DR has inspected the request headers and determined that the client should proceed with sending the request body. If there is a problem with the request, or if the DR

determines that the request should be redirected to a different URL, the DR will send a final response with an appropriate status code rather than sending a 100 response.

The DR will respond to a successful PUT or DELETE request with an HTTP response carrying a status code of 204 ("No Content"). This response indicates that the DR has received the request and found it acceptable. It does not indicate that the request has been propagated successfully to all subscribers. The response will include a header named X-ATT-DR-PUBLISH-ID whose value is a unique identifier assigned by the DR for the request.

To make sure that the client sends requests to the nearest DR node, the DR may redirect a PUT or DELETE request to another URL by responding with a status code of 301 ("Moved Permanently") . A client that receives a 301 response must repeat the request, sending it to the URL in the Location header of the response (the *redirect URL*) . The redirect operation establishes a new effective publishing URL for the feed. The new effective publishing URL is just the redirect URL, minus the file identifier component . The client must use this new effective publishing URL for all future PUT or DELETE requests for the feed, until another redirect is received. If at any time a request fails because the client cannot connect to the DR node using the new effective publishing URL (for instance, because the connection times out or the connection is refused), the client should attempt to send the request to the original file publishing URL established when the feed was created. Thus the client must store both the original file publishing URL and the last effective publishing URL.

In the event of a problem with the request, the DR will return a response with an appropriate error status code. Some possible status codes include:

- 400: Indicates a malformed request.
- 401: Indicates that the request was missing the Authorization header or, if the header was presented, the credentials were not acceptable.
- 403: Indicates that the user could not be authenticated or that the user was not authorized to make the request.
- 404: Indicates that the path in the Request-URI did not point to a valid feed publishing URL.
- 500: Indicates that the DR experienced an internal problem.
- 503: Indicates that the DR is not currently available. The response may include a Retry-After header.

As is standard practice in HTTP, clients should not re-submit requests that elicit responses with status codes in the range 400-499 without correcting the problem, as the requests will continue to fail. Status codes in the range 500-599 indicate a problem with the DR; requests that elicit these status codes can

be resubmitted after the DR problem has been cleared. A 503 response may include a Retry-After header that the client can use to determine when to resubmit the request.

Any publishing client that does not implement the "100-continue" capability should watch for error responses from the DR as the client is transmitting the body of a request. If the client receives an error response, the client should immediately stop transmitting the body. (See RFC 2616, section 8.2.2.)

File Delivery

The DR uses essentially the same interface to deliver files and retractions to subscribers as it uses to receive files and retractions from publishers. In the case of delivery, the DR acts as an HTTPS client. File deliveries are made with PUT requests, and file retractions are made with DELETE requests. Subscriber endpoints must run software capable of handling these HTTP requests.

A PUT or DELETE request originated by the DR has the following characteristics:

- The Request-URI is an absolute path consisting of the path component of the delivery URL specified at the time the subscription was created, followed by a "/" , followed by the file ID passed in the original file publishing request, followed by any query string or fragment that appeared in the Request-URI in the original publishing request.
- The Host header contains the host name and port address (if one is present) from the delivery URL.
- The Authorization header specifies basic HTTP authentication and includes the user ID and password specified at the time the subscription was created, encoded as specified in RFC 2617, section 2. The DR always includes this header in the request.
- The X-ATT-DR-META header contains the metadata for the file, expressed as a JSON object, as provided by the publisher in the original file publishing PUT request.
- The X-ATT-DR-PUBLISH-ID header carries the unique identifier assigned to the file publishing request that was responsible for initiating this delivery request. This allows each delivery request originated by the DR to be correlated with a file publishing request.
- The X-ATT-DR-RECEIVED header carries information about the path the request has taken through the data router network. This information can be used to determine the address from which the request was originally published and the DR nodes that have handled the request. The format of the X-ATT-DR-RECEIVED header is described in Appendix 2.
- Any headers whose names begin with "X-" but not "X-ATT-DR" provided by the publisher in the original file publishing PUT or DELETE request will be copied into the file delivery request.

The DELETE request has no body. The PUT request carries a body unless the subscription has been configured as a metadata-only subscription. A PUT request has the following additional characteristics:

- The Content-Type header is copied from the Content-Type header provided by the publisher in the original file publishing PUT request.

- Any of the following headers provided by the publisher in the original file publishing PUT request will be copied into the file delivery request, if the subscription is not a metadata-only subscription:
    - Content-Language
    - Content-MD5
    - Content-Range

  (Note that the DR does not validate the content of these headers, nor does it assign them any meaning.)
- If the subscription has been configured to enable the use of the 100-continue feature, an Expect header with the value "100-continue" will be present.
- The DR may provide a Content-Length header or may use a chunked transfer encoding.
- The body of the request will be an exact copy of the body in the original file publishing PUT request.

The PUT or DELETE request is transmitted via HTTPS to the host and port address specified in the authority component of the delivery URL. If no port is explicitly specified in the URL, the request is directed to the default HTTPS port (443).

The DR expects the subscriber software receiving the request to present a server certificate signed by VeriSign and issued under AT&T Services, Inc . The subscriber should arrange to obtain such a certificate from the appropriate source.

If the subscription has been configured to enable the use of the 100-continue feature, the subscriber software must honor the 100-continue expectation if the request contains the Expect header. The subscriber must send a response with a status code of 100 or a non-success final response after it has received the request headers.

The subscriber software receiving a request should return a response with an appropriate status code.

- The DR will treat any code in the range 200-299 as a success. (We recommend using 204, just as the DR does in responding to successful file publishing request.)
- If the DR receives a response with a status code of 300-399 (indicating a redirection), it behaves in the same way that a publisher is expected to behave. The DR will attempt to deliver to the redirect URL in the `Location` header of the response. The DR stores the redirect URL, minus the file identifier component, and uses it as the delivery URL for the subscription for subsequent deliveries, until a delivery results in another redirection. If, on a subsequent delivery attempt, the DR cannot connect to the effective delivery URL established by a redirect, the DR will attempt to deliver to the original provisioned delivery URL for the subscription.
- If the DR receives a response with a status code of 500-599, the DR will attempt the request at a later time.

- The DR will log any other non-success response and will not attempt the request again.

. If a subscription is configured to use the 100-continue feature, the subscriber software can inspect the HTTP headers on the incoming request (including, for example, the per-file metadata carried in the X-ATT-DR-META header) and decide to reject the incoming request, using a 4xx response, before receiving the data in the body.

Appendix 1: Format of the X-ATT-DR-META Header

The X-ATT-DR-META header carries the per-file metadata, expressed as a JSON object. The DR accepts a subset of the full JSON language[15]:

- The DR accepts only JSON objects, not JSON arrays.
- The DR does not accept objects that contain nested objects or arrays. To put it somewhat differently, the DR accepts only those objects whose properties have only the following types of values: The DR accepts objects that contain properties storing only the following types of values:
    - Numbers
    - Strings
    - `true` or `false`
    - `null`

Appendix 2: Format of the X-ATT-DR-RECEIVED Header

The X-ATT-DR-RECEIVED header provides information on the path that the published file (or retraction) has taken through the data router system. For each "hop", the header has an entry with a timestamp (in ISO-8601 format, stated in UTC, with millisecond resolution), the address of the node receiving the request, and the address of the host that made the request. For example:

```
X-ATT-DR-RECEIVED: 2013-01-24T21:25:00.495Z;from=192.168.1.50;by=192.168.1.175
```

This indicates a request received on January 24, 2013 at 21:25:00.495 UTC, from a host at IP address 192.168.1.50 by a host at 192.168.1.175.

The first entry in an X-ATT-DR-RECEIVED header indicates when and how the request entered the DR network. The "from=" attribute gives the address of the endpoint that published the request. The "by=" attribute gives the address of the DR node that accepted the request.

In many cases, the DR node that accepts the request will deliver it directly to the subscriber. In those cases, there will be only one entry in the header. In other cases, the DR node that receives the request will transfer the request to another DR node that's closer to the subscriber endpoint. In those cases, there will be two entries in the header. The first entry still indicates the initial entry of the request into the DR network. The second entry shows when the request was transferred to another DR node, and what DR node received the request. For example:

```
X-ATT-DR-RECEIVED: 2013-01-24T21:25:00.495Z;from=192.168.1.50;by=192.168.1.175,2012-01-
24T21:25:01.095;from=192.168.1.175;by=192.168.1.188
```

This first entry of this header shows the entry into the network, as before. The second entry shows that the DR node that accepted the original request (192.168.1.175) passed the request to the DR node at 192.168.1.188, 600 milliseconds after accepting the request from the publisher.

In rare cases, there may be more than two entries in the header, each showing another DR node through which the request has passed.

Appendix 2: Examples

Here are examples of a file publishing (PUT) request from a publisher to the DR, a file delivery (PUT) request from the DR to a subscriber, and a file retraction (DELETE) request, from the publisher to the DR and from the DR to a subscriber. The responses to the requests are also shown. In the example, we assume:

- The publisher has user ID `jack` and password `password123`
- The feed publishing URL is `https://dr.example.com:4949/feedx849`
- The publisher has chosen the file ID `access-log-2012-10-17-0004` for this file
- The file metadata consists of a server name ("server") and a date ("date")
- There is a subscription to the feed with the following characteristics:
    - The delivery URL is `https://deliver.example.com:8443/myfeed`
    - The delivery user name is `datarouter` and the delivery password is `password123`
    - The subscription is configured so that the use of the 100-continue feature on delivery is disabled.
- The file is delivered to the subscriber endpoint from the DR node that accepts the original file publishing request.

Here are some things to note about the examples:

- The `Authorization` header follows the RFC 2617 specification, signaling basic authentication and using a base-64 encoding for the user ID/password pair.
- The `User-Agent` header isn't strictly required, but the DR will not complain about it. The values used in the examples are just that—examples. The value used by the DR has not been established, and no implementation should depend on seeing a specific value.
- The `Accept` header also isn't strictly required. Since the DR does not return any content in its responses to these requests, it ignores the header.
- The `Server` and `Date` headers in the responses are not strictly required, but they are not forbidden. HTTP application frameworks often insert them automatically. Like the `User-Agent` values, the `Server` values shown in these examples are just examples. The value used by the DR has not been established, and no implementation should depend on seeing a specific value.

File Publishing

The publisher publishes a file to the DR. The request includes a Content-Range header and an X-Simple-Publisher-Sample-Header, both of which will ultimately be delivered to subscribers. The DR does not attempt to interpret these headers.

Request: publisher to DR node

```
PUT /feedx849/access-log-2012-10-17-0004 HTTP/1.1
Host: dr.example.com:4949
Authorization: Basic amFjazpwYXNzd29yZDEyMw==
User-Agent: SimplePublisher/0.23a
Accept: */*
Content-Type: text/plain
X-ATT-DR-META: {"server" : "preston", "date" : "2012-10-17"}
X-SimplePublisher-Sample-Header: this is a sample
Expect: 100-continue
Content-Range: bytes 4000-5265/*
Content-Length: 1266
```

At this point, after sending the request headers, the publisher pauses, waiting for response from the DR node. In this example, the DR node finds the request acceptable and sends a response with status code 100 to indicate that the publisher should process with sending the body of the data.

Provisional response: DR to publisher

```
HTTP/1.1 100 Continue
```

Now the publisher sends the body, which happens to be a portion of a Web server log file.

```
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /coll/symposium2012/ HTTP/1.1" 200
19496 "-" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5) Gecko/20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/blank.gif HTTP/1.1" 200 148
"http://dr.com/coll/symposium2012/" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5) Gecko/
20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/back.gif HTTP/1.1" 200 216
"http://dr.com/coll/symposium2012/" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5) Gecko/
20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/movie.gif HTTP/1.1" 200 243
"http://dr.com/coll/symposium2012/" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5) Gecko/
20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/image2.gif HTTP/1.1" 200
309 "http://dr.com/coll/symposium2012/" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5)
Gecko/20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/unknown.gif HTTP/1.1" 200
245 "http://dr.com/coll/symposium2012/" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5)
Gecko/20100101 Firefox/10.0.5"
```

After receiving the body, the DR node responds. The response includes a transaction ID.

Success response: DR to publisher

```
HTTP/1.1 204 No Content
Server: Apache-Coyote/1.1
Date:  Thu, Jan 24 2013 21:25:00 GMT
X-ATT-DR-PUBLISH-ID: ef9481ea-6a43-11e2-bf58-001b243dc997
```

File Delivery

The DR delivers the file that it accepted from the publisher in the previous example. Note that the request includes the X-ATT-DR-PUBLISH-ID and X-ATT-DR-RECEIVED headers. Note also that the DR includes the Content-Range and X-Simple-Publisher-Sample-Header headers that the publisher included in the original file publishing request.

Request: DR to subscriber

```
PUT /myfeed/access-log-2012-10-17-0004 HTTP/1.1
Host: deliver.example.com:8443
Authorization: Basic ZGF0YXJvdXRlcjpwYXNzd29yZDEyMwo=
User-Agent: ATT DataRouter/1.0
Content-Type: text/plain
X-ATT-DR-META: {"server" : "preston", "date" : "2012-10-17"}
X-ATT-DR-PUBLISH-ID: ef9481ea-6a43-11e2-bf58-001b243dc997
X-ATT-DR-RECEIVED: 2013-01-24T21:25:00.495Z;from=192.168.1.50;by=192.168.1.175
X-SimplePublisher-Sample-Header: this is a sample
Content-Range: bytes 4000-5265/*
Content-Length: 1266

135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /coll/symposium2012/ HTTP/1.1" 200
19496 "-" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5) Gecko/20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/blank.gif HTTP/1.1" 200 148
"http://dr.com/coll/symposium2012/" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5) Gecko/
20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/back.gif HTTP/1.1" 200 216
"http://dr.com/coll/symposium2012/" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5) Gecko/
20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/movie.gif HTTP/1.1" 200 243
"http://dr.com/coll/symposium2012/" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5) Gecko/
20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/image2.gif HTTP/1.1" 200
309 "http://dr.com/coll/symposium2012/" "Mozilla/5.0 (Windows NT 6.1; rv:10.0.5)
Gecko/20100101 Firefox/10.0.5"
135.44.24.219 - - [17/Oct/2012:13:37:39 +0000] "GET /icons/unknown.gif HTTP/1.1" 200
245 "http://preston.proto.research.att.com/coll/symposium2012/" "Mozilla/5.0 (Windows
NT 6.1; rv:10.0.5) Gecko/20100101 Firefox/10.0.5"
```

The DR sends the body immediately after the headers because the subscription is configured not to use the 100-continue feature. In this example, the delivery is successful and the subscriber sends a success response.

Response: subscriber to DR

```
HTTP/1.1 204 No Content
Server: SimpleSubscriber/1.0.1
Date:  Thu, Jan 24 2013 21:25:04 GMT
```

File Retraction

The publisher sends a retraction to the DR.

Request: publisher to DR node

```
DELETE /feedx849/access-log-2012-10-17-0004 HTTP/1.1
Host: dr.example.com:4949
Authorization: Basic amFjazpwYXNzd29yZDEyMw==
User-Agent: SimplePublisher/0.23a
X-ATT-DR-META: {"server" : "preston", "date" : "2012-10-17"}
Accept: */*
```

The DR responds. The response includes a transaction ID.

Success response: DR to publisher

```
HTTP/1.1 204 No Content
Server: Apache-Coyote/1.1
Date:  Fri, Jan 25 2013 01:55:00 GMT
X-ATT-DR-PUBLISH-ID: 58ee9676-6c7e-11e2-81b0-001b243dc997
```

File Retraction Delivery to Subscriber

The DR delivers the retraction that it accepted from the publisher in the previous example. The request includes the X-ATT-DR-PUBLISH-ID and X-ATT-DR-RECEIVED headers.

Request: DR to subscriber

```
DELETE /myfeed/access-log-2012-10-17-0004 HTTP/1.1
Host: deliver.example.com:8443
Authorization: Basic ZGF0YXJvdXRlcjpwYXNzd29yZDEyMwo=
User-Agent: ATT DataRouter/1.0
Content-Type: text/plain
X-ATT-DR-META: {"server" : "preston", "date" : "2012-10-17"}
X-ATT-DR-PUBLISH-ID: 58ee9676-6c7e-11e2-81b0-001b243dc997
X-ATT-DR-RECEIVED: 2013-01-25T01:55:00.372Z;from=192.168.1.50;by=192.168.1.175
```

The subscriber receives the retraction successfully.

Response: subscriber to DR

```
HTTP/1.1 204 No Content
Server: SimpleSubscriber/1.0.1
Date:  Fri, Jan 25 2013 01:55:04 GMT
```

Change History

Version 1.5

New behavior for DR when it receives a 300-399 (redirect) response.

Version 1.4

Clarify how a client should handle a 301. (Bug #15). Correct some typographical errors, update reference to *Provisioning API* document.

Version 1.3

Add information about certificate signing .

Version 1.2

Change redirect status code to 301 and clarify how a client should respond to a redirect.

Version 1.1

Add references to the Provisioning API document, which was not available at the time Version 1.0 was written.

Version 1.0

Initial version; reviewed and baselined without substantive changes on 2013-02-11.