Default Signature Type

Signing a Request

By default, apstrata database will use a secure signature which depends on the data sent. It involves most of the data in the request, and it is calculated using the secret authentication key.

To Create a signature:

1. Create a standardized string of the query string to be hashed.
    a. URL encode both the parameters and their values.
        i. In case of an attachment, the value would be the URL encoded hexadecimal representation of the MD5 hash of the file bytes (all capital letters).

Example

```
MessageDigest messageDigest = MessageDigest.getInstance("MD5");
FileInputStream fis = new FileInputStream(filePath);
byte[] bytes = new byte[4096];
int readPos = fis.read(bytes);
while (readPos != -1) {
      md5.update(bytes, 0, readPos);
      readPos = fis.read(bytes);
}
String hashedValue = hexToString(messageDigest.digest()).toUpperCase();
```

    b. Separate the encoded parameters and their values by the equals sign.
    c. Sort the parameters and their values by natural byte ordering (as described in the below example).
    d. Append the name value pairs with an ampersand.
2. Create the string to be hashed:

Hashed String

```
String to hash = HTTPVerb (Upper case) + "\n" + encoded Http URL + "\n" + standardized string
```

   The HTTP URL will include the scheme, host, port (if present), and the path. It will not include any of the query strings.
3. Calculate the HMAC-SHA1 hash string using the secret authentication key.

⚠ Spaces and asterisk should be encoded as %20 and %2A respectively.

Example

```
POST http://sandbox.apstrata.com/apsdb/rest/[authentication_key]/CreateStore
Host: host
.........
apsdb.store=myStore&additionalParam1=value1&apsws.time=1234567890
```

Sorted Parameters

```
additionalParam1=value1
apsdb.store=myStore
apsws.time=1234567890
```

String to Hash

```
"POST\n
http%3A%2F%2Fsandbox.apstrata.com%2Fsandbox%2Dapsdb%2Frest%2Fauthenticationkey%2FCreateStore\n
additionalParam1=value1&apsdb.store=myStore&apsws.time=1234567890"
```

Assuming that the secret authentication key is “secret”

```
HMAC-SHA1("secret", "POST\n
http%3A%2F%2Fsandbox.apstrata.com%2Fsandbox%2Dapsdb%2Frest%2FmyKey%
2FCreateStore\nadditionalParam1=value1&apsdb.store=myStore&apsws.time=1234567890")
```

Example (PHP)

```
POST http://sandbox.apstrata.com/apsdb/rest/[authenticationkey]/CreateStore
Host: host
.........
apsdb.store=myStore&additionalParam1=value1&apsws.time=1234567890


$paramArr = array();
array_push(rawurlencode("apsws.time") . "=" . rawurlencode("1234567890"));
array_push(rawurlencode("apsdb.store") . "=" . rawurlencode("myStore"));
array_push(rawurlencode("additionalParam1") . "=" . rawurlencode("value1"));


sort($paramArr);


$stringToSign = "";
for($i = 0; $i <count($paramArr); $i++)
{
            $stringToSign .= $paramArr[$i];
            If($i <count($paramArr) - 1)
            {
                        $stringToSign .= "&";
            }
}
$stringToSign = "POST" . "\n" . rawurlencode (http://sandbox.apstrata.com/apsdb/rest/[authenticationkey]
/CreateStore)
. "\n" . $stringToSign;
// assuming that the secret authentication key has the value "secret"
$signature = hash_hmac("sha1",  $stringToSign, "secret");
```