SmarterServices

Helping you make smarter decisions

Request Authentication

Creating the Request Signature

HMAC-SHA1

Introduction

All SmarterServices web services and Single Sign On (SSO) requests must contain authentication information to establish the identity of the principal making the request. In a SOAP request this information is put in the elements of the SOAP request. For SSO requests, this information should be contained in the HTTP request header.

The elements that are to be required in each request are as follows:

AccessKey	The access key that is assigned to your account by the service being accessed.
TimeStamp	This must be a dateTime in the Coordinated Universal Time (Greenwich Mean Time) time zone - ex. 2009-01-01T12:00:00Z. Authorization will fail if this timestamp is more than 5 minutes from the eLearningToolBox servers. Our servers are synced with the United States Naval Observatory (USNO) Atomic Clock.
Resource	The resource that is being requested. This will be included in the documentation for the specific resource being consumed.
RequestSignature	The RFC 2104 HMAC-SHA1 digest (http://www.ietf.org/rfc/rfc2104.txt) of the concatenation of TimeStamp + SharedSecret, using your Shared Secret Key as the key.

The following sections will guide you through the process of creating your Request Signature.

IMPORTANT: All requests to external SmarterServices APIs are required to connect with SSL. All non-SSL requests will fail.

Signature Creation Walk-Through

1) Generate the timestamp

The current date/time should be generated in the ISO8601 format with the precision of seconds. The timestamp will be used to ensure the request is completed within 5 minutes. Timestamps outside of the 5 minute window will fail authentication. Depending on the resource that is being accessed, this timeout value may differ. Please refer to the documentation on the resource being accessed for the exact timeout value.

Example (PHP)

```
# Set Time stamp variable as ISO 8601 Format
$timestamp = time();
$iso8601timestamp = gmdate("Y-m-d",$timestamp) . "T" . gmdate("H:i:s",$timestamp) . "Z";
```

2) Create the algorithm key

The algorithm key is a concatenation of the timestamp and the shared secret that you were provided.

Example (PHP)

```
$sharedSecret = "MySharedSecretKey"; // Shared secret key provided by SmarterServices
# Create the algorithmKey variable by combining the timestamp and shared secret.
$algorithmKey =  $iso8601timestamp . $sharedSecret;
```

3) Generate the Message Authentication Code (MAC)

This step will require the use of the HMAC-SHA1 algorithm. The "algorithmKey" generated in the last step will be used and the resource is the test that will be signed. For this example using PHP, the hmacsha1() function is a custom written function and can be found in the appendix.

Example (PHP)

```
#Create Message Authentication Code (MAC) and encode as base 64.
#This example calls the reporting service
$resource = "/external/services/v1/reporting.cfc?wsdl"; //URL string after the domain
$requestSignature = base64_encode(hmacsha1($algorithmKey, $resource));
```

4) Sending the request

After the request signature has been created you are now ready to send the request. Consult the documentation for the specific request you are making in order to properly add the request signature and other required specific parameters to your request.

Appendix

PHP HMAC-SHA1 Function

```
# Define HMAC-SHA1 Creation Function
function hmacsha1($key,$data) {
   $blocksize=64;
   $hashfunc='sha1';
   if (strlen($key)>$blocksize)
      $key=pack('H*', $hashfunc($key));
   $key=str_pad($key,$blocksize,chr(0x00));
   $ipad=str_repeat(chr(0x36),$blocksize);
   $opad=str_repeat(chr(0x5c),$blocksize);
   $hmac = pack(
            'H*',$hashfunc(
               ($key^$opad).pack(
                  'H*',$hashfunc(
                     ($key^$ipad).$data
                  )
               )
            )
         );
   return $hmac;
}
```