Contents

[Creating Basic Authentication Header 1](#_Toc486405437)

[Checking for Basic Authentication Header 2](#_Toc486405438)

[Encryption Keys 3](#_Toc486405439)

Each service call will have a Basic Authentication Header which is essentially, a base64string consisting of encrypted unix timestamp string, concatenated with MD5 Hashed information, separated by colon (:) as described below:

E.g.

//Basic Authentication Header Base64 String  
Authorization: Basic TnZsZDZsR0ZqL1pXVnRkOXpMdE1ldz09Ok9UUXdOMEl3TlRRNE5Ua3hOamRGUWtGQlEwVXpOalpHUlRVMU5VRkZOa1k9

# Creating Basic Authentication Header

The Basic Authentication Header consists of two parts:

1. **The encrypted unix timestamp.**

The unix timestamp is the elapsed seconds since Jan 1, 1970. Timestamp for 27/06/2017 12:00 would be 1498564800. The timestamp is then encrypted using blowfish encryption (keys for the encryption is shared below).

Encrypted string for the same would be dh7ofs+rmR2V7YROzzm5bw==

***In production server, each service call will be valid only for 5 minutes i.e. if the difference between this timestamp & the UTC time at the time of service call is more than 5 minutes, the call will be terminated with a timeout exception. However, on development environment the timestamp validity will be 15 minutes for the ease of debugging.***

1. **MD5 Hash information.**

Following information is tightly concatenated without any character separation & an MD5 hash is created:

1. Secret key e.g. “123456”
2. The METHOD of the request e.g. “post” or “get”
3. The Content Type of the request e.g. “x-www-form-urlencoded”
4. The Action name = “feedback”

*FORMAT: <SECRET\_KEY>+<METHOD>+<CONTENTTYPE>+<ACTIONNAME>*

E.g. Concatenated String: 123456postapplication/x-www-form-urlencodedfeedback

MD5 Hash of the concatenated string is created and converted into a base64 string.

E.g. MD5 hash: 2BD0AE3D99192C873E10A9DBBFE7750F

Base64string: MkJEMEFFM0Q5OTE5MkM4NzNFMTBBOURCQkZFNzc1MEY=

The resultant timestamp & MD5 hash are then concatenated with a colon (:) as shown below:

*FORMAT: <encrypted\_timestamp>:<base64 MD5 hash>*

e.g. dh7ofs+rmR2V7YROzzm5bw==:MkJEMEFFM0Q5OTE5MkM4NzNFMTBBOURCQkZFNzc1MEY=

This string is further converted into a base64 string and passed as the basic header.

e.g. Base64string : ZGg3b2ZzK3JtUjJWN1lST3p6bTVidz09Ok1rSkVNRUZGTTBRNU9URTVNa000TnpORk1UQkJPVVJDUWtaRk56YzFNRVk9

Authentication Header: Basic ZGg3b2ZzK3JtUjJWN1lST3p6bTVidz09Ok1rSkVNRUZGTTBRNU9URTVNa000TnpORk1UQkJPVVJDUWtaRk56YzFNRVk9

# Checking for Basic Authentication Header

Within the service, before decrypting the data sent, the first & foremost step would be to check the Basic Authentication. The timestamp & MD5 hash must be validated to authenticate the sender.

1. Check for the presence of Basic Authentication Header.
2. Remove the word ‘Basic ’ from the Authentication string & decode the base64string.
3. Split the resultant string by colon (:) separator.
4. The split should give two different strings
   1. The timestamp
   2. MD5 Hash
5. Decrypt the timestamp string using blowfish encryption.
6. Check the difference between the timestamp sent and the current UTC time to be within 5 minutes, otherwise throw ‘timeout’ exception.
7. Decode the second part of Authentication Header which is the base64 encoded MD5 hash.
8. Recreate the MD5 hash using the below information from the web request.
   1. Secret key e.g. “123456”
   2. The METHOD of the request e.g. “post”
   3. The Content Type of the request e.g. “x-www-form-urlencoded”
   4. The Action name = “feedback”

*FORMAT: <SECRET\_KEY>+<METHOD>+<CONTENTTYPE>+<ACTIONNAME>*E.g. Concatenated String: 123456postapplication/x-www-form-urlencodedfeedback

MD5 Hash of the concatenated string is created   
E.g. MD5 hash: 2BD0AE3D99192C873E10A9DBBFE7750F

1. Compare this MD5 hash with the one extracted from the Authentication header to be same, otherwise throw incorrect authentication exception.
2. Once the timestamp is validated & MD5 hash authenticated, decrypt the data & proceed.

# Encryption in Input data:

The input parameters are shared as Encrypted JSON String.

1. Prepare the JSON Object as per the API documentation.
2. Serialize the JSON Object
3. Encrypt using the blowfish encryption.
4. Now, pass the encrypted data in JSON format with “encryptedtext” as the key and actual data as value. E.g. {"encryptedtext":"eENHfd3g1OecdqLQNnBvsWNXv/to+qH9x3guKlLH1wXkEvILf+4N0PhvhREO7OZUsAlxx46YmiqRk+XhXCoSpzxd816IXh1V71pSrSThNpDWPKfOO7Cti1nZHXfJ3gyuHE6qfNlz8l0="}
5. If the Programming is in
   1. C#/ .Net, Following DLLs can be used to implement Blowfish encryption: https://www.bouncycastle.org/

# Encryption Keys

The Data passed should be encrypted with blowfish encryption.

Following keys should be used to encrypt the data as well as Unix Timestamp in Basic Authentication Header.

blowfishkey = TO BE Shared Separately  
blowfishIV = TO BE Shared Separately  
secretkey = TO BE Shared Separately  
Web service method = "post"  
Content type = " application/json"