Symmetric Searchable Encryption component

API specification version 0.3.0

Component name: SSE – Symmetric Searchable Encryption
Component deployment name: sse

Changelog

Version	Date	Pages	Author	Modification
0.1	26/2/2020	2	Hai-Van Dang	Initial release with Javascript APIs for upload and search data
0.2	15/4/2020	4	Hai-Van Dang	Update response of uploadData function Add specification for updateData function
0.3	5/5/2020	6	Hai-Van Dang	Add specification for deleteData function, uploadKeyG function, change parameter names of previous functions

Table of Contents

Terminology

Terminology/ Abbreviation	Explanation
End-user	User who uploads/ searches data
SSE server	Server which stores encrypted data
Trusted Authority	Server which stores metadata necessary for upload/ search encrypted data

Introduction

This API specification covers APIs relevant to uploading, search, and update data, which are implemented in Javascript. Data upload is the process that a user chooses to send data, i.e. Json object, to SSE server in cloud. Data search is the process when a user wishes to search for the stored encrypted data in SSE server by providing a Json object by template. Data update is the process when a user wishes to update values of the whole or part of a stored Json object.

The following section describes specification of the Javascript library functionalities which supports the above processes.

API description

Upload data: function uploadData(data,file_id,pwd1,pwd2)

This API allows a user to encrypt a Json object, then send its ciphertext to SSE server. It currently supports the following Json object format:

- JSON objects are surrounded by curly braces {}.
- JSON objects are written in key/value pairs.
- Keys must be strings, and values can be string, number. It does not support array and object type for values.
- Keys and values do not contain the vertical slash symbol, i.e. "|" (Because the vertical slash is used as string denominator in the implementation).
- Keys and values are case-sensitive
- Keys and values are separated by a colon.
- Each key/value pair is separated by a comma.

Parameters:

Name	Type	Description
data	Json object	Data to be uploaded, which is a JSON object
file_id	String	File identifier, which must be unique string
pwd1	String	Passphrase which is used to compute a shared symmetric key of users and TA
pwd2	string	Passphrase which is used to compute symmetric encryption key for users. This passphrase is needed to upload/ search/ update/ delete data

Hash computation of pwd1 and pwd2 are different. Therefore, client can use the same value for pwd1 and pwd2 while TA still does not learn anything about their values and symmetric encryption key.

Example:

```
uploadData({firstname: "David", lastname: "White", age:
25},"id1","pwd1","pwd2")
```

Response

Returned type	Description
Boolean value	True if uploaded successfully False if failed to upload data

Search data: function search(data,pwd1,pwd2)

Search for encrypted data in SSE server by providing a search content. SSE server will return encrypted files which contain the searched keyword.

The search content is a Json object, which follows the following format:

- JSON objects are surrounded by curly braces {}.
- JSON objects are written in one key/value pair.
- Key is "keyword", and value is a combined string of an attribute and its value separated by the vertical slash symbol, i.e. "|". For instance, if a user wishes to search for "firstname=David", the value will be "firstname|David".
- Key and value are case-sensitive

Example:

```
{
    "keyword": "firstname|David"
}
```

Parameters:

Name	Type	Description
data	Json	Searched data, which is a Json object
pwd1	String	Passphrase which is used to compute a shared symmetric key of users and TA
pwd2	String	Passphrase which is used to compute symmetric encryption key for users. This passphrase is needed to upload/ search/ update/ delete data

Example:

```
search({"keyword": "firstname|David"}, "pwd1", "pwd2")
```

Response

Returned type	Description

Json object	Json object contains the number of found objects, and their content. {count: <number of found objects>, objects: <Array of Json objects, which contain decrypted data>}

Update data: function updateData(data,file_id,pwd1,pwd2)

This API allows a user to update the whole or part of a Json object which is identified by its file_id. It currently supports the following Json object format:

- JSON objects are surrounded by curly braces {}.
- JSON objects are written in key/value pairs.
- Keys must be strings, and values are arrays of size two. The first item of the array is the current value of the corresponding key, and the second item is the update value. The two items are separated by comma.
- Keys and values do not contain the vertical slash symbol, i.e. "|" (Because the vertical slash is used as string denominator in the implementation).
- Keys and values are case-sensitive
- Keys and values are separated by a colon.
- Each key/value pair is separated by a comma.

Example:

```
{
    "firstname":["David","Peter"],
    "lastname":["White","Yellow"]
}
```

Parameters:

Name	Type	Description
data	Json	Update data, which is a Json object
file_id	String	File identifier, which must be unique string
pwd1	String	Passphrase which is used to compute a shared symmetric key of users and TA
pwd2	String	Passphrase which is used to compute symmetric encryption key for users. This passphrase is needed to upload/ search/ update/ delete data

Example:

```
updateData("firstname":["David","Peter"],"id1", "pwd1", "pwd2")
```

requests to update firstname from "David" into "Peter" of the Json object with "id1".

Response

Returned type	Description

Boolean value	True if updated successfully False if failed to update

Delete data: function deleteData(file_id,pwd1,pwd2)

This API allows a user to delete a Json object which is identified by its file_id.

Parameters:

Name	Type	Description
file_id	String	File identifier, which must be unique string
pwd1	String	Passphrase which is used to compute a shared symmetric key of users and TA
pwd2	String	Passphrase which is used to compute symmetric encryption key for users. This passphrase is needed to upload/ search/ update/ delete data

Example:

```
updateData("id1", "pwd1", "pwd2")
```

requests to delete the Json object with "id1".

Response

Returned type	Description
Boolean value	True if deleted successfully False if the provided file_id does not exist

Upload shared key: function uploadKeyG(pwd1)

This API allows a user to upload a shared key to Trusted Authority.

Parameters:

Name	Type	Description
pwd1	String	Passphrase which is used to compute a shared symmetric key of users and TA

Response

Returned type	Description

Boolean value	True