Attribute based encryption algorithm.

The selective scheme for attribute-based encryption is as follows.

If at least one attribute in the set{t\_i}\_U is equal to the attribute in the set{t\_i}\_M, the corresponding user U can decrypt the text M.

As soon as user and share have one attribute in common – the user can get access to the share.

1. **Master-key (MK)** which is kept safely on server and accessible only for the domain administrator

*The values ti are randomly selected from the huge group Zp . They are the private keys corresponding to the group attributes. Note, that this is different from the usual PK encryption: the private keys are controlled by the admin and not by the users.*

1. **Public key (PK)** depends on the master key values and is kept in the clear allowing users to access the information:

Here e(g,g) is the bilinear pairing.

1. Secret user **KEY\_SET** depends on his attribute set. Here each Di (**GROUP\_KEY**) serves for decryption of the data of a single group of users, for example, related to some project:
2. **Encryption** of the text M (in our case the text is the **FILE\_KEY**, or the permanent AES symmetric key, the key is permanent in order to reduce the necessity to re-download the files on the device) is multiplication. The set of the public keys Ei (**PUBLIC\_SHARE\_KEY**) corresponding to the set of groups able to access the text is kept along with the encrypted text:
3. **Decryption** is division:

In order to perform this operation the user needs Di corresponding to the secret attribute ti and :

*The result of decryption is the FILE\_KEY - the symmetric AES key to decrypt the contents of file.*

The general workflow of the key generation is as follows:

The generation of the encryption parameters:

1. {ti} are generated according to the user groups. They can be generated in the beginning (some limited number) or can be added to the MK value gradually.
2. The generated MK should be backed up.
3. The parameter w allows to change the user GROUP\_KEY and PUBLIC\_SHARE\_KEY without affecting the result of encryption. In other words, Ei and Di are modified and the value E(KF) = KF Yys stays the same. This allows to avoid re-downloading the files on the device each time the user keys are refreshed. This parameter is temporary and IS NOT KEPT. This allows to protect the GROUP\_KEY data even if the MK is stolen.
4. The parameters sij allows to control the access to specific files within a group. In a situation when some file is deleted (not accessible) for the group members their parameter sij is changed so that they do not access this file anymore. In this situation all the PUBLIC\_SHARE\_KEYs Ei are modified so that the unauthorized access to file KF is forbidden.
5. The main complexity is the necessity to exponentiate each time the public/private keys are generated which is a heavy operation.
6. The secrecy is based on DDH. There are scenarios when malicious user keeps using his old key in order to access the file. WE use some offline protection mechanisms in order to restrict it.

Below we describe the workflow of periodical key renewal, key renewal when the user is added or deleted from group and key renewal when the file is added or deleted from group.

Periodical key renewal

**NOTE.** The description of the server-client communication is a subject of the other technical paper. We propose to use an effective EKE protocol for this purpose. In our case this could be J-PAKE or SIS-EKE due to the time constraint and the necessity to re-generate the public keys within the specific time interval.

Key renewal when user is deleted from the group

NOTE. The vulnerability in this situation is that the user could still keep his PUBLIC\_SHARE\_KEY and use it. In this case we can propose:

* Forbid to copy the PUBLIC\_SHARE\_KEY
* Keep track of user activities (i.e. the device was not connected for a long time)

Key renewal when user is added to the group

When the user is added to the group there is no need to renew the keys. Server needs to send the GROUP\_KEY (protected) to the client and the PUBLIC\_SHARE\_KEY (along with encrypted file, in the clear) to the user.

Key renewal when file is deleted from the group

NOTE.

1. The vulnerability in this situation is that the user could still keep his PUBLIC\_SHARE\_KEY and use it. In this case we can propose:

* Forbid to copy the PUBLIC\_SHARE\_KEY
* Keep track of user activities (i.e. the device was not connected for a long time)

1. This function is based on the observation that:

*And there is no more PUBLIC\_SHARE\_KEY to decrypt this value.*

Key renewal when file is added to the group

*Note that this is based on the following equalities:*