Single Sign On for Mimeo - Developers Guide

What is it?

Single sign-on (SSO) is a method of access control that enables a user to authenticate once and gain access to the resources of multiple software systems. That means your users can access Mimeo directly from your corporate intranet, LMS system or any other secure system deemed to be a trusted authority without requiring separate Mimeo login information. SSO is designed for use with a private Mimeo Marketplace. Existing users are seamlessly logged into Mimeo without having to enter any login information. For users who are new to Mimeo, an account is created on the fly and the appropriate corporate pricing and other account attributes are applied automatically. If your system also provides Mimeo with an internal unique account ID, the SSO will even support email address changes.

Why do I want it?

Benefits of using the SSO include:

- Reduce or eliminate administrative costs around Mimeo account management
- Added convenience leads to increased user adoption, satisfaction and productivity
- Immediate and automatic set up of new Mimeo users 24/7

How does it work?

Mimeo's SSO uses a simple HTTP Post method and .Net RSA signing and encryption to receive and process authentication and authorization requests. .Net RSA involves a public and private key pair. The public key can be known to everyone and is used for encrypting messages. The encrypted messages can only be decrypted using the corresponding private key. Mimeo will provide you with a Public Key (for encryption), unique CompanyID and RedirectURLs. When a user clicks a link from your system to log into Mimeo your system will send an HTTP Post with encrypted data (FirstName, LastName, Email, internal unique accountID, RedirectURL Destination, etc.), along with your CompanyID, which Mimeo will use to identify you and corresponding private key (for decryption) decrypt the data. For users new to Mimeo, an account will automatically be created and permissioned to the appropriate entitlements before being logged in. Existing Mimeo users will be logged in immediately and their Mimeo account data will be updated.

How do I start?

Mimeo will provide you with a datasheet containing all the data values and URLs needed to begin coding.

What data needs to be sent?

The SSO request Url which accepts Http post method to submit encrypted data is:
https://my.sandbox.mimeo.com/sso/authenticate.ashx

Parameter	Description	Encrypted	Provided By
redirectUrl	Url that user needs to be directed to after user account has been authenticated successfully. This will be the Marketplace URL.	No	Mimeo
errorUrl	Url that user needs to be redirected to if any error occurs during authentication process. This page is hosted by client.	No	Client
firstName	User's first name	Yes	Client
lastName	User's last name	Yes	Client
email	User's email address.	Yes	Client
companyName	User's company name.	Yes	Client
customId	Internal unique account ID. (Optional)	Yes	Client
companyId	Mimeo assigned company ID.	No	Mimeo
initialCredit	One time assigned to a newly created user account. (Optional)	Yes	Client
ssoVersion	Mimeo assigned SSO Version #.	No	Mimeo
organizationId	Mimeo assigned Organization ID.	Yes	Mimeo
authorizedMarketPlaceUrl	The Marketplace url which user has permission to access.	No	Mimeo

Encryption algorithm and code sample

The technique enables you to transmit the user's information in encrypted form from the browser to the server, using the asymmetric RSA algorithm. The form submission content is encrypted using the public key and only the server knows the corresponding private key in order to decrypt the data.

The following is code snippet in .Net to demonstrate how to encrypt user's email value.

```
string SamplePublicKey =

"<RSAKeyValue><Modulus>4JO9VHkG3lHBwex/6HGA/j9w9eO0F1R65uoaagpGl3spbWM55
115N2n3Jn+i+XRfLHkxdkMBy1Sr23IAOQg0rf+2ZbzK/nZDBnk59bYy5BI8dBvVHwb2a9Kg
atC8cf0XJaIKgZE7es1wID/bRYQz60XUQu40XeEOZWbjVh5e0DM=</Modulus><Exponent
>AQAB</Exponent></RSAKeyValue>";

string email = "jchen@mimeo.com";

using (RSACryptoServiceProvider rsa = new RSACryptoServiceProvider())
{
  // read public key into RSA class.
 rsa.FromXmlString(SamplePublicKey);

 // encrypt email value.
 byte[] encryptedEmailAccount=rsa.Encrypt(new ASCIIEncoding().GetBytes(email), false);

 // convert the encrypted email value to base64String
 this.encryptedUserData=Convert.ToBase64String(encryptedEmailAccount, 0,
 encryptedEmailAccount.Length);

}
```

Implementation language other than Microsoft .Net framework

The core RSA algorithm is a simple modular exponentiation, or rather, is a simple equation with very big numbers, therefore big numbers are crucial to ensure the strength of the algorithm. Raw RSA encryption without any padding is not secure because the number used could turn out to be small. For this reason Microsoft's API asks for mandatory padding. Incompatible padding scheme from the other language such as openSSL, javascript would produce the "bad data" exception at the server side.

There are two padding schemes used in .NET RSA implementation. The first is PKCS#1 v1.5 padding and another is OAEP(PKCS#1 v2) padding.

Quoted from the Microsoft's documentation:

Direct Encryption (PKCS#1 v1.5)

Microsoft Windows 2000 or later with the high encryption pack installed. Padding: Modulus size - 11. (11 bytes is the minimum padding possible.)

Mimeo Single Sign on uses Direct Encryption(PKCS#1 v1.5) padding scheme.

More research on the simpler PKCS#1 v1.5 (see here , for 8.1 Encryption-block formatting) shows how padding should be carried out. During encryption, pseudorandom nonzero bytes are generated and the final padded message (before modular exponentiation) should look like this:

```
                0x00 || 0x02 || PseudoRandomNonZeroBytes || 0x00 ||
Message
```

Error Codes & Messages

Note that optional fields not being used should be represented by an empty string.

Code	Message
101	The Organization ID does not exist.
102	The user's email address already exists on other user's record.
103	The user is disabled.
104	The user's organization doesn't match.
106	EncryptionKey format is not correct.
107	Error decrypting submitted key.
108	The authorizedMarketPlaceUrl not found in database.
109	The user's email already exists in database record.
110	Unable to create a new user account.
111	Initial credit is not in correct format. If there is a decimal point it requires 2 numeric characters after the decimal point. For example, 3.10 is valid but 3.1 is not.
120	GUID format error – Company ID= { xxxx }.
122	The company doesn't exist. Company ID = {xxxxx}.
124	The length of first name, last name, custom id, email and organization ID can't be zero.
125	Company RSA CryptoKey does not exist.
127	Email format is incorrect.

For all questions and comments, please contact:

Raul Moncada
Director, Enterprise Development
1-512-578-9450
Email: rmoncada@mimeo.com