Today I want to tell about my little experience with crypto providers. In a few words it is widely used for protecting your personal data in commercial area, internet banking and allow or deny access to smart house, for example.

There are a lot of different formats and methods to store your private key: .p12 file, security tokens, smart cards and it would be great to have a convenient way for getting the data from them in your program.

So, many security companies provide their own SDK with a specific API but also implement popular interfaces such as Microsoft Crypto API, JCE for accessing their crypto functions. In other words, they have their own cryptography service provider (CSP).

As a rule every country has its own encryption, digital signature standards (such as BelT in Belarus) and one or more CSP, that implements this algorithm. Examples of CSP:

1. CryptoPro CSP (<https://www.cryptopro.ru/products/csp>)
2. TumarCSP (<http://www.gamma.kz/products/tumar_csp>)
3. AvestCSP (<http://avest.by/crypto/csp.htm>)

You just need to download this program and install it in your system. So, how we can use it in your program?

Of course, CryptoAPI is the most flexible and popular solution, but it is complex and sometimes difficult way for an ordinary software developer. And what about Java? You have to access it via JNI or something else and waste a lot of time for mapping all the parameters.

But almost every CSP has an access via COM interface. And it’s a lot easier in do it on Java! By the way, Microsoft created a special library for this many years ago called CAPICOM. It is deprecated now, but it is still working ☺

To get it working on Java I’ve created a project capicom-wrapper. To use it you have to:

1. Download CAPICOM from official site
2. Register it in your system: regsvr32.exe capicom.dll
3. Add capicom-wrapper in your project:

Add the repository:

<repository>

<id>capicom-wrapper-mvn-repo</id>

<url>https://raw.github.com/creepid/capicom-wrapper/mvn-repo/</url>

<snapshots>

<enabled>true</enabled>

<updatePolicy>always</updatePolicy>

</snapshots>

</repository>

And dependency:

<dependency>

<groupId>by.creepid</groupId>

<artifactId>capicom-wrapper</artifactId>

<version>0.1</version>

</dependency>

Let’s try to sign something like string “test” using AvestCSP:

//2 - storeLocation

//"My" - storeName

//2 - openMode

CapicomStore store = new CapicomStore(2, "My", 2);

//1 – subject certificate number in windows store

CapicomCertificate cert = store.getCertificates().getAll()[1];

//create the signer object

CapicomSigner signer = new CapicomSigner();

//set the certificate to it

signer.setCertificate(cert);

//define the data to sign

CapicomSignedData signedData = new CapicomSignedData();

//"test" - signing string

signedData.setContent("test");

//signing time definition

CapicomAttribute signingTime = new CapicomAttribute();

//Add signing time attribute, see CapicomAttributeEnum

signingTime.setName(CapicomAttributeEnum.CAPICOM\_AUTHENTICATED\_ATTRIBUTE\_SIGNING\_TIME);

//set the value

signingTime.setValue(new Date());

//add the signing time attrubute

signer.getAuthenticatedAttributes().add(signingTime);

//return signature in Base64, throws InvalidCertificate otherwice

String signature = signedData.sign(signer, true);

More examples it available here.