Web приложения
Описание инструментов цифровой подписи
(ТОВ «Базис»)

Общая информация

Клиент

На стороне клиента используется (в зависимости от ОС):

Windows: Криптопровайдер CESARIS CSP.

Поддерживает национальные стандарты ГОСТ 34.311, ДСТУ ГОСТ 28147, ДСТУ 4145 и RSA, SHA-1, SHA-2; поддерживает е-токен Avest (ДСТУ 4145, RSA) и др. (RSA).

Хранилище ключей:
- файловый токен на жестком диске (Windows);
- файл *.pfx (стандарт PKCS#12) на любом носителе;
- смарт-карта и е-токен.

Соответствует спецификации Microsoft, соответствует стандарту PKCS#11, интегрирован с операционной системой Windows XP/2003/2008/7/8.

При программировании на C++, C# можно использовать MS CryptoAPI.

Дополнительно можем предоставить (если требуется) библиотеку функций, которая построена на вызовах MS CryptoAPI, и содержит основные функции: подписать, проверить подпись и т.д.

Mac OS X, Linux...: Криптопровайдер Java “AMBProvider”.

Поддерживает национальные стандарты и все основные международные криптографические стандарты.

Соответствует спецификации Java JCE/JCA.

Хранилище ключей:
- хранилище “PKCS12” (*.pfx, стандарт PKCS#12);
- хранилище Java “JKS”.

Сервер

На стороне сервера могут использоваться те же криптопровайдеры, что и на клиенте.

Хранилище ключей (рекомендуется):
- хранилище “PKCS12” (*.pfx, стандарт PKCS#12);
- аппаратные модули HSM (при наличии).

Дополнительные инструменты Java

AMB ASN (ambasn.jar) – обработка ASN.1;

AMB API (ambapi.jar) – доп.инструменты и desktop утилиты (например, управление хранилищем PKCS12, проверка установленных смарт-карт и е-токен устройств и др.);

PDF API (pdfapi.jar) – операции с PDF файлами: подписать, проверить подпись и т.д.

Web клиент

Мульти-браузерность (IE, Opera, Firefox, Chrome, Yandex) обеспечивается применением Java Applet (ambclient.jar).

Основные функции Java Applet (ambclient.jar) клиента

Пример подписать данные см. **http://itsway.kiev.ua/signPKCS7.html** и подписать PDF файл см. **http://itsway.kiev.ua/signPdfPfx.html**

Установлена Java?

До загрузки апплета требуется проверить, установлена ли Java, и если нет – предложить установить.

Для этого в теле javascript (в <head>) поместить код:

```
var javaEnabled = navigator.javaEnabled();//установлена Java?
if (!javaEnabled) {//не установлена Java
	userInput = confirm("Вам необходимо загрузить Java(TM) Runtime Environment.\nВы
	согласны?");
	if (userInput) {
		window.location = http://java.com/ru/download/testjava.jsp
	}
}
```

Загрузка апплета

Это одна из критических операций. Так как возможны такие ситуации: одновременно открыто несколько окон/закладок одного приложения в одном браузере (разных браузерах); библиотека(и) апплета обновилась и т.д, - требуется, чтобы каждый экземпляр апплета загружался в отдельной JVM и требуется, чтобы WEB приложение должно быть написано так, чтобы апплет не перегружался в течение сессии.

Ниже приведен javascript загрузки апплета (поместить в <head>), удовлетворяющий этим требованиям.

Примечание. В случае, когда апплет перегружается в течение сессии, можно предложить другую процедуру загрузки – работа в одной JVM всех экземпляров апплета. Однако в этом случае будут конфликты при обновлении библиотек апплета.

```
var _ie = navigator.appName.indexOf("Microsoft") > 0;//IE 7,8,9;
if (!_ie)
	_ie = navigator.userAgent.indexOf("Trident") > 0;//IE 10, 11 appName = "Netscape"

function startApplet() {
	  var appletHtml;
	  if (_ie) {// BEGIN: for IE 7-11
	     appletHtml = '<object id="signerApplet" name="Bazis Signer Client"
classid="clsid:CAFEEFAC-0016-0000-FFFF-ABCDEFFEDCBA" width="1" height="1"
declare="declare"> '
+ '<param name="type" value="application/x-java-applet"/> '
+ '<param name="code" value="com.amb.applet.view.SignerWin"/> '
+ '<param name="codebase_lookup" value="false"/> '
+ '<param name="codebase" value="./plugins/"/> '
+ '<param name="cache_option" value="Plugin"/> '
+ '<param name="cache_archive_ex"
	value="ambprovider.jar;preload,ambapi.jar;preload,pdfapi.jar;preload,ambasn.jar;preload "> '
+ '<param name="cache_archive" value="ambclient.jar"/> '
+ '<param name="autoplay" value="true"/> '
+ '<param name="autostart" value="true"/> '
+ '<param name="mayscript" value="true"/> '
```

```
+ '<param name="scriptable" value="true"/> '
+ '<param name="language" value="ru"/> '
+ '<param name="java_arguments" value=" -Djnlp.packEnabled=true -Xms512m -Xmx1024m -
Dsun.java2d.d3d=false -Dsun.java2d.noddraw=true"/> '
 + '<\/object>';
          }
          else {
               // BEGIN: NETSCAPE, Chrome, Opera, Forefox.
appletHtml = '<embed id="signerApplet" name="Bazis Signer Client" classid="clsid:CAFEEFAC-
0016-0000-FFFF-ABCDEFFEDCBA" width="1" height="1" '
+ 'type="application/x-java-applet" pluginspage="http://www.java.com/en/download/" '
+ 'code="com.amb.applet.view.SignerWin" codebase_lookup="false" codebase="./plugins/" '
+ 'cache_option="Plugin" '
+
'cache_archive_ex="ambprovider.jar;preload,ambapi.jar;preload,pdfapi.jar;preload,ambasn.jar;preloa
d" '
+ 'cache_archive="ambclient.jar" autoplay="true" autostart="true" '
+ 'mayscript="true" scriptable="true" language="ru" '
+ 'java_arguments=" -Djnlp.packEnabled=true -Xms512m -Xmx1024m -Dsun.java2d.d3d=false -
Dsun.java2d.noddraw=true"/>';
           }

           document.getElementById("signer").innerHTML = appletHtml;
        }
```

Примечания.

1) Здесь language="ru" – язык сообщений апплета; поддерживаются "ru", "uk", "en".

2) Библиотеки ambprovider.jar,ambapi.jar,pdfapi.jar должны находиться в папке "./plugins/".

3) На странице должен быть создан («пустой») элемент апплета:

```
<div id="signer"/>
```

4) В аргументах java_arguments указано

```
-Djnlp.packEnabled=true
```

Это означает, что все *.jar файлы должны быть в упакованном формате *.jar.gz

5). Функцию загрузки надо запустить так

```
<body onload="startApplet();">
```

Проверка загрузки апплета

Большинство браузеров не разрешают (первую) загрузку апплета без подтверждения юзера (политика безопасности). Запрос «разрешить…» не всегда видят юзеры (или не обращают внимания).

Во-вторых, иногда Java устанавливается в системе, но Java плагин под браузер нет. Это часто происходит, если при установке Java браузер открыт.

В обоих случаях апплет не загружается и браузер не выдает никаких сообщений, - молча молчит.

Чтобы помочь клиенту понять причину, надо вставить скрипт проверки загрузки апплета:

```
function appletLoaded() { //загружен апплет?
    setTimeout(function go() {
        if (document.body) {
            var applet;
```

```
                    if (_ie)
                      applet = document.applets["signerApplet"];
                    else
                      applet = document.embeds[0];

                    var s = new String(applet);
                    var text = "Нет разрешения для Java в браузере - необходимо разрешить.\nОбратите внимание на верхнюю/нижнюю строку сообщений браузера.\nПроверьте, что подключаемые модули Java не заблокирована в браузере";
                    if (_ie && s.indexOf("HTMLObjectElement") > 0 && navigator.appName.indexOf("Microsoft") < 0) {
                        //для IE 10, 11 эта проверка не работает
                        var selects = document.getElementsByTagName("select");
                        if (selects[0].options.length == 0) {
                          alert(text);
                        }
                    }
                    else if (s.indexOf("java") < 0) {
                      alert(text);
                    }
                    return;
                  }
                },
        10000);
            }
```

Примечания.

1) Функцию надо запустить после загрузки апплета:

```
<body onload="startApplet();appletLoaded();">
```

2) В этом коде функции таймаут (10000) = 10 сек – ожидание загрузки апплета. При плохих каналах связи можно увеличить.

Вспомогательная функция

Используется другими функциями скрипта:

```
function getApplet() {
        var app;
        if (_ie)
            app = document.applets["signerApplet"];
        else
            app = document.embeds[0];
        return app;
    }
```

Информация о бразере

Эта функция вызывается из апплета для получения информации о браузере. Имя функции (getUserAgent) не должно изменяться.

```
function getUserAgent() {
        return navigator.userAgent;
    }
```

Загрузить список сертификатов при старте апплета

Эта функция вызывается из апплета при его старте.

При старте апплет считывает сертификаты, установленные у пользователя и формируется их список для отображения на странице. Имя функции (onLoadPage) не должно изменяться.

```
function onLoadPage() {
        try {
           getCertList();
        }
        catch (exc) {
           ;
        }
}
```

Получить из апплета список сертификатов пользователя

Получить из апплета список сертификатов для элемента select (здесь id="certlist") страницы.

В список включаются только сертификаты, предназначенные для подписи (KeyUsage: signature); исключаются сертификаты, срок действия которых истек, и сертификаты, у которых не установлена полная цепочка доверия (сертификатов).

```
function getCertList() {
        var jsArray = [];
        var jsArrayDate = [];
        var select = document.getElementById("certlist");

        var applet = getApplet();
        jsArray = applet.getCertCN();
        jsArrayDate = applet.getDateTo();

        select.options.length = 0;
        for (var i = 0;i < jsArray.length;i++) {
           if (jsArrayDate[i] != "")
              jsArray[i] = jsArray[i] + " ; Действителен до: " + jsArrayDate[i];
           select.options[select.options.length] = new Option(jsArray[i], i);
        }

        var data = applet.getCertCount();
        var obj = document.getElementById("certCount");
        if (data != null && obj != null) {
           obj.value = data;
        }

    }
```

Элемент (не обязательный) с id="certCount" и функция апплета getCertCount() предназначены для отображения общего числа загруженных сертификатов.

Данные о загруженных сертификатах

Для получения данных о загруженных сертификатах используются списки данных (по индексу списка) о сертификате, это такие функции апплета:

applet.getCertCN() – список общего имени (Common Name) сертификата (для физ.лица – ФИО, для печати – название организации и т.д.);

applet.getDateTo() – список окончания срока действия сертификата (дата, время);

applet.getDateFrom() – список начала срока действия сертификата (дата, время);

applet.getSN() – список серийного номера сертификата в Hex-формате (макс. 64 символа);

applet.getIssuers() – список общего имени (Common Name) эмитента сертификата (кем выпущен)

Примечание. Список applet.getCertCN() должен вызываться первым, т.к. при его вызове фактически обновляется список сертификатов.

Показать (открыть) выбранный сертификат

Запустить в апплете функцию просмотра (View) выбранного сертификата (по индексу в списке):

```
function viewCert() {
        var index = document.getElementById("certlist").selectedIndex;
        var applet = getApplet();
        applet.viewCert(index);
}
```

У Windows вызывается его стандартное приложение просмотра сертификата и его атрибутов, которое выполняет полную проверку валидности сертификата. На других платформах вызывается приложение из библиотеки ambapi.jar, где есть команда (кнопка) проверки валидности сертификата.

Загрузить файл с заданного URL

Адрес загрузки (с сервера) должен быть задан в элементе, например, с id="documentURL"; апплет возвращает путь+имя файла, куда был загружен файл на клиенте:

```
function download() {
        var applet = getApplet();
        var url = document.getElementById("documentURL").value;
        var infile = applet.download(url);
        if (infile != null)
           document.getElementById("inputFile").value = infile;
}
```

Выбрать (на клиенте) файл для подписывания

В апплете выполняется выбор файла (из файловой системы клиента); возвращается путь+имя выбранного файла:

```
function getChoisedFile() {
        var applet = getApplet();
        document.getElementById("inputFile").value = applet.fileChoiser();
}
```

Подписать PDF файл

Вызывается для подписи файла PDF. Апплет возвращает имя выходного/ подписанного файла:

```
function signFile() {
        document.getElementById("signFileName").value = "";//очистить

        var indx = document.getElementById("certlist").selectedIndex;//выбранный сертификат
        var tsp = document.getElementById("TimeStampCheckBox").checked;//добавить метку времени
        var crl = document.getElementById("CRLCheckBox").checked;//добавить списки отзыва

        var tsaUrl = document.getElementById("hiddenTsaUrl").value;//URL службы метки времени
       if (tsp == false)
          tsaUrl = "";

        var applet = getApplet();
        var data = applet.getSign(indx, tsaUrl, crl);

        if (data != null) {
          //имя подписанного файла можно вывести
          //если ошибка при подписывании, то возвращает пустое имя ""
          document.getElementById("signFileName").value = data;
        }
      }
```

Примечания.

1) добавить к подписи метку времени (boolean элемент с id=”TimeStampCheckBox”) и списки отзыва (boolean элемент с id=”CRLCheckBox”) – опциональны (не обязательны). Если не используются надо задать null.

2) Настоятельно рекомендуется добавлять к подписи:

- метку времени – фиксирует время подписания; это требуется для придания законодательного статуса подписи;
- списки отзыва – фиксирует валидность сертификата подписи на момент подписания; это требуется для придания законодательного статуса подписи.

3) Если метка времени добавляется, то в скрытом элементе с id=”hiddenTsaUrl” надо указать URL службы метки времени. Это должно быть:

<input type="hidden" id="hiddenTsaUrl" value="http://cesaris.itsway.kiev.ua/tsa/srv/"/>

4) **Ограничение**: Можно сформировать максимум 4 подписи на одном PDF документе, которые размещаются на каждой странице (внизу).

Проверить подписи PDF-файла

Проверить подписи PDF-файла. Возвращает 0 (true) или 1 (false); сообщения и краткие детали результата проверки выводит сам апплет.

Выполняется экспресс (минимальная) проверка:

- проверяет все существующие подписи PDF,
- проверяет срок действия сертификатов подписи

Не проверяет:

- списки отзыва CRL,
- подписи сертификатов и все их цепочки.

Имя проверяемого файла в элементе с id=”signFileName”

```
function signatureVerify() {
```

```
    var applet = getApplet();
    var url = document.getElementById("signFileName").value;
    result = applet.verify(url);
}
```

Показать (открыть) исходный PDF-файл

Для открытия файла используется приложение системы, зарегистрированное для файлов PDF:

```
function viewSourcePdf() {
    var applet = getApplet();
    var file = document.getElementById("inputFile").value;
    applet.viewPdf(file);
}
```

В этом вызове открывается исходный файл с id="inputFile".

Для открытия используется стандартное приложение, зарегистрированное в системе для файлов PDF.

Показать (открыть) подписанный PDF-файл

Открыть подписанный файл (с id="signFileName "):

```
function viewSignedPdf() {
    var applet = getApplet();
    var file = document.getElementById("signFileName").value;
    applet.viewPdf(file);
}
```

Для открытия используется стандартное приложение, зарегистрированное в системе для файлов PDF.

Формирование/проверка подписи PDF на сервере приложений Java

На серверах приложений под Java требуется установить Java JDK. Рекомендуется последняя версия JDK 7; не рекомендуется (пока) JDK 8.

! Версии JDK 7/8 64-бит неполные, - нет интерфейсных классов с эллиптическими кривыми и др., - поэтому необходимо устанавливать JDK 7 32-бит.

Допускается JDK 6, хотя этот инструмент будет снят с поддержки/развития, и поэтому может в дальнейшем иметь уязвимости безопасности.

Криптопровайдер ambprovider.jar и библиотеки ambapi.jar, ambasn.jar, pdfapi.jar (не упакованный формат) надо разместить в папку:

Windows - JDK_HOME\jre\lib\ext\

И соответственно для других платформ.

Примечания.

1) Типовые функции

Типовые функции для работы с PDF документами на стороне сервера, обычно, включают:

- подписать (добавить одну подпись);
- проверить подписи (все);
- получить количество подписей на документе (и их имена);
- проверить заданную подпись (одну по имени);
- была ли изменена редакция (заданная подпись); может использоваться, чтобы определить, какая из подписей ошибочна;
- сравнить сформированный документ (оригинал до подписания) и подписанный по содержанию/ контенту;

и др.

Перечень функций зависит от приложения и согласования требований разработчиков приложений.

2) Быстродействие

Второй фактор – требуемое быстродействие. Хранилище ключей, - это хранилище PKCS12 (*.pfx файл).

При формировании подписи обращение к хранилищу PKCS12 (чтение, расшифрование ключа и т.д.) – трудоемкая операция, снижающая быстродействие системы. Наиболее оптимальным является подход, при котором ключ подписи считывается в память при старте приложения. Но реализация при этом зависит от количества ключей, используемых для подписи и т.д., т.е. требуется согласование с разработчиками приложений.

Классический *.pfx файл (сформированный, например, в Windows операцией экспорта сертификата и его ключа) содержит один ключ подписи, т.е. можно несколько ключей (если необходимо) хранить в нескольких файлах.

Но можно также в одном файле формата PKCS12 хранить много ключей и сертификатов. Если в одном хранилище есть ряд ключей, то они «структурированы» по алиасам (alias). Ключ, сертификат ключа и его цепочка в хранилище связаны одним алиасом. Требуемый ключ подписи должен выбираться по алиасу разработчиками приложений.

Ниже приведены примеры для случая, когда при формировании подписи идет обращение к хранилищу PKCS12, которое содержит один ключ подписи сертификат ключа и его цепочку (при получении сертификата с ЦСК CESARIS – это резервная копия ключа, т.е. файл *.pfx).

Пример в файле TestSignaturePdf.java

Во-вторых, при установке на сервере JDK обязательно установить режим ”server” в настройках Application Server – быстродействие выше примерно в 2 раза.

Загрузка криптопровайдера

Загрузка криптопровайдера Java AMBProvider обязательна:
1) Может быть выполнена статически, конфигурированием файла Java java.security:

security.provider.10= com.amb.security.provider.AMBProvider

2) Может быть выполнена динамически:

```
try {
    Provider provider = Security.getProvider("AMB");
    if (provider == null) {
       provider = new AMBProvider();
       Security.addProvider(provider)          }
 } catch (Exception e) {
    throw new RuntimeException(e.toString());
 }
```

Проверить подпись

filename – файл документа, который надо проверить.

```
    FileInputStream fis = new FileInputStream(fileName);

    SignaturePdf signaturePdf = new SignaturePdf();
    //инициализация проверки
    signaturePdf.initVerify(fis);

    //получить имена подписей (алиасы)
    //имена идут по порядку наложения подписей: Sign_1, Sign_2, Sign_3, Sign_4,
    Vector<String> signNames = signaturePdf.getSignatureNames();
    if (signNames.size() == 0) { //документ не подписан
       System.out.println("Not signed");
    }

    //проверить все подписи
    //если успешно - возвращает все подписи в формате PKCS7, иначе null
    PdfPKCS7[] pkcs7 = signaturePdf.verify();
    if (pkcs7 != null) { //OK
       System.out.println("All signature is OK");
    }

    //если есть ошибка - найти какая из подписей «битая»
    PdfPKCS7 p7;
    for (int i = 0; i < signNames.size(); i++) {
       p7 = signaturePdf.verify(signNames.get(i));
       if (p7 == null) {
          System.out.println(signNames.get(i) + " signature is BAD");
```

```
        } else {
            //можно получить доп.инфо о подписи
            //дата подписания документа (локального компа подписанта)
            Calendar signingDate = p7.getSignDate();

            //дата подписания документа согласно метки времени, если была добавлена
            Calendar timeStampDate = p7.getTimeStampDate();
            //и т.д.

            //сертификат подписанта
            X509Certificate cert = p7.getSigningCertificate();

            //проверить валидность сертификата по сроку действия на данный момент
            try {
                cert.checkValidity();
            } catch (CertificateExpiredException e) {
                // certificate has expired.
            } catch (CertificateNotYetValidException ex) {
                //certificate is not yet valid
            }

            //ФИО подписанта (common name)
            String fio = X509Utils.getSubjectDNField(cert, "CN");
            //дата выпуска сертификата
            Date beginDate = cert.getNotBefore();
            //дата окончания действия сертификата
            Date endDate = cert.getNotAfter();

        }
    }
```

Функция подписать

```
//добавить подпись
//хранилище ключей подписи
String pfxFile = path + "my_133573.pfx";
//пароль к хранилищу
String pfxPassword = "1";

fis = new FileInputStream(pfxFile);
signaturePdf = new SignaturePdf(fis, pfxPassword);
 fis.close();

 //список алиасов ключей в хранилище PKCS12
List<String> aliases = signaturePdf.getListAliases();
//выбираем требуемый алиас; здесь всего один ключ - поэтому индекс = 0
String alias = aliases.get(0);
//выбираем сертификат ключа по его алиасу
X509Certificate signCert = signaturePdf.getCertificateFromStore(alias);

//инициализация подписи - алгоритм подписи такой же, как у сертификата ключа
signaturePdf.initSign(signCert);
```

```
//Добавить метку времени к подписи - задать URL службы метки времени
//если не надо добавлять метку, то задать null
String tsaUrl = "http://cesaris.itsway.kiev.ua/tsa/srv/";

//добавить к подписи списки отозванных сертификатов
boolean addCRL = true;

//Поток для подписанного документа
ByteArrayOutputStream output = new ByteArrayOutputStream();
fis = new FileInputStream(fileName);
//подписать
signaturePdf.sign(fis, tsaUrl, addCRL, output);

//сохраним в файл
FileOutputStream fos = new FileOutputStream(path + "bill_2.pdf");
fos.write(output.toByteArray());
fos.close();

//проверим, что bill_1.pdf, bill_2.pdf - это документы с одним содержанием/контентом
boolean b = SignaturePdf.compareContent(new FileInputStream(path+"bill_1.pdf"), new
FileInputStream(path+"bill_2.pdf"));
if (b) {
        System.out.println("Compare content is OK");
 }
```

Дополнительная информация по подписи

В разделе (выше) «Проверить подпись» мы получили (извлекли из PDF) все подписи, и выбрали из них одну по имени подписи (PdfPKCS7 p7;) и получили доп.информацию:

```
//дата подписания документа (локального компа подписанта) – есть всегда
Calendar signingDate = p7.getSignDate();

//дата подписания документа согласно метки времени, если была добавлена
Calendar timeStampDate = p7.getTimeStampDate();
//если timeStampDate == null – нет метки времени
```

Теперь извлечем расширенную информацию по подписи.

```
//метка времени
TimeStampToken  tsToken = p7.getTimeStampToken();
if (tsToken != null) {//метка времени есть
        System.out.println("    Данные об органе, выдавшем метку времени: " +
                        tsToken.getSignerInfo().getIssuerName().getName());
        //проверить подпись метки времени
        SignerInfo si = tsToken.verify();
        if (si != null) {
                System.out.println("    Подпись метки времени ОК.");

                //проверить, что метка времени относится к этому документу (по
        значению хеш)
                if (!p7.verifyTimestampImprint()) {
```

```
                    System.out.println("Ошибка: Метка времени не связана з этим документом.");
                    }

                    //цепочка сертификатов подписи метки времени
                    // 0 – подписи; 1 – ЦСК; 2 – корневой
                    X509Certificate[] tsCerts = tsToken.getCertificates();
                    //информация из сертификатов
                    …
            } else
                    System.out.println("    Ошибка подписи метки времени.");

    } else
            System.out.println("\n  Подпись документа не содержит метку времени.");
```

Дополнительная информация из сертификата

В разделе (выше) «Проверить подпись» мы получили (извлекли из PDF) все подписи и затем цепочку сертификатов конкретной подписи, т.е.:

```
…
Vector<String> names = pdfapi.getSignatureNames();//имена подписи
…
По имени подписи извлекаем конкретную подпись p7 с ее проверкой:
PdfPKCS7 p7 = pdfapi.verify(names.get(i));
```

Получим доп.информацию из сертификатов

```
import java.security.cert.Certificate;
import java.security.cert.X509Certificate;

import com.amb.asn1.x509.X509Utils;


…
// цепочка сертификатов подписи:  0 – подписи; 1 – ЦСК; 2 – корневой
Certificate[] certs = p7.getCertificates();

X509Certificate cert = certs[0];//сертификат подписанта
//Извлечь Common Name (для сертификата юзера – это Ф.И.О., для сертификата печати – название организации
String subjecrCN = X509Utils.getSubjectDNField(cert, "CN");
//также можно для полей: O – организация; OU – подразделение организации, и др.

//Проверить подпись сертификатов по всей цепочке
X509Certificate certUser = (X509Certificate)certs[0];//юзер
X509Certificate certCA = (X509Certificate)certs[1];//промежуточный ЦСК
X509Certificate certRoot = (X509Certificate)certs[2];//корневой ЦСК
try {
        certUser.verify(certCA.getPublicKey());
} catch (Exception e) {
        System.out.println("Ошибка подписи сертификата подписи: " + e.toString());
}
try {
        certCA.verify(certRoot.getPublicKey());
```

```
        } catch (Exception e) {
                System.out.println("Ошибка подписи сертификата промежуточного ЦСК: " +
        e.toString());
        }
        try {
                certRoot.verify(certRoot.getPublicKey());
        } catch (Exception e) {
                System.out.println("Ошибка подписи сертификата корневого ЦСК: " +
e.toString());
        }

        //Идентификатор ключа подписи субъекта/подписанта – байт массив (до 32 байт)
        byte[] ski = X509Utils.getSubjectKeyIdentifier(certUser);
        //Идентификатор ключа подписи эмитента/ЦСК – байт массив (до 32 байт)
        byte[] aki = X509Utils. getAuthorityKeyIdentifier (certUser);

        //извлечь из сертификата пользователя код ИНН и ЕДРПОУ (если есть)
        //код ИНН – только один, обязательно
        //код ЕДРПОУ – не обязательно (для физ.лиц); обязательно для представителя
(директор, бух…) организации, может быть несколько ЕДРПОУ

        //получить все расширения DirectoryAttributes (OID=”2.5.29.9”).
        String inn = null; // ИНН
        ArrayList<String> edrpous = new ArrayList<String>(); // ЕДРПОУ

        Attribute[] attrs = X509Utils.getSubjectDirectoryAttributes(certUser);
        if (attrs != null) {
                String s;
                // INN
                for (int m = 0; m < attrs.length; m++) {
                        s = attrs[m].getOID().toString();
                        if (s.equals("1.2.804.2.1.1.1.11.1.4.1.1")) {
                           inn = (String)((ASN1PrintableString)attrs[i].valueAt(0)).getValue();
                           break;
                        }
                }
                // EDRPOU
                if (attrs.length > 1) {
                   for (int m = 0; m < attrs.length; m++) {
                      s = attrs[m].getOID().toString();
                      if (s.equals("1.2.804.2.1.1.1.11.1.4.2.1")) {

edrpous.add((String)((ASN1PrintableString)attrs[i].valueAt(0)).getValue());
                        }
                   }
                }
        }
```

Вопросы и предложения просьбы направлять по адресу
tech@itsway.kiev.ua