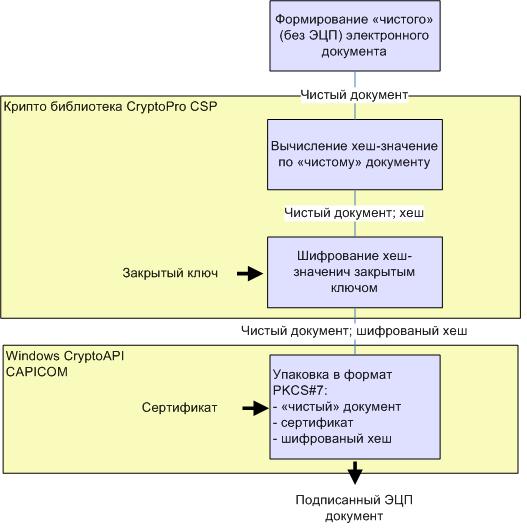
Формат электронного сообщения, подписанного ЭЦП

ЭЦП подписывается не почтовое сообщение, а непосредственно сам электронный документ (edi или xml файл, содержащий данные документа).

Результатом при этом является файл в PKCS#7 формате, включающий:

* исходный документ;
* цифровую подпись;
* сертификат лица, подписавшего документ.

Укрупненный алгоритм получения подписанного документа:



[Пример](http://www.rlisystems.ru/ReleaseOrder/digsig_sample.zip) (zip, 3 КБ, в состав архива входят неподписанный документ, подписанный документ и тестовый сертификат, использованный для подписи)

Базовое программное обеспечение для подписи электронных документов

В качестве базового ядра, обеспечивающего реализацию криптографических алгоритмов для подписания и проверки подписанных документов, используется программное обеспечение компании [КРИПТО-ПРО](http://www.cryptopro.ru/).

Независимо от выбранного Вами сценария получения подписанного документа Вам необходимо приобрести Криптопровайдер КриптоПро CSP 3.6 и установить его на все компьютеры (рабочие станции или серверы), где будет производится непосредственное подписание документов.

Скачать [тестовую версию КриптоПро CSP](http://www.cryptopro.ru/products/csp/downloads) (действительна в течение 90 дней) с сайта КРИПТО-ПРО.

Получение сертификатов ЭЦП

[Инструкция по получению сертификатов ЭЦП .](http://www.globalports.com/ru/terminals/first-container-terminal/documents/)

[Инструкция по установке, настройке криптобиблиотеки и генерации ключей](http://www.rlisystems.ru/faq/faq.html) на сайте компании «Ролис».

При необходимости возможна также генерация временных сертификатов ЭЦП для проведения тестирования разрабатываемого Вами механизма создания ЭЦП для Ваших информационных систем.

Внешние приложения для подписи электронных документов

Ниже описаны те программные решения, которые были найдены и опробованы нами в рамках разработки приемной части электронного обмена с использованием ЭЦП.

Мы не настаиваем на использовании именно этого программного обеспечения, если вы сможете найти более дешевые(или бесплатные)/удобные/доступные альтернативы – можно попробовать использовать их.

**Интерактивная утилита**

Для создания и проверки подписанных ЭЦП документов в интерактивном режиме (при участии пользователя) можно воспользоваться программой КриптоАРМ, демонстрационную версию которой можно скачать [здесь.](http://www.trusted.ru/support/downloads/?product=133)

**Консольная утилита (приложение командной строки)**

Домашняя страничка утилиты: <http://www.cryptopro.ru/cryptopro/products/cryptcp/default.htm>

Тестовая лицензия: <http://www.cryptopro.ru/cryptopro/products/cryptcp/licence.pdf>

Утилита (исполняемый файл): <http://www.cryptopro.ru/cryptopro/products/cryptcp/3-16/cryptcp.exe>

Описание параметров: <http://www.cryptopro.ru/cryptopro/products/cryptcp/3-16/CryptCP.pdf>

Краткое описание использования:

1. Регистрация утилиты

Вводим тестовый серийный номер.

cryptcp.exe -sn P020G-Q0010-A5000-01UXA-XUFFD

2. Подпись документа

Подписываем файл coreor.edi, результат сохраняем в coreor.edi.p7s, сертификат берем из файла ./sergey.cer

cryptcp.exe -sign -f ./sergey.cer -der coreor.edi coreor.edi.p7s

3. Проверка подписи.

cryptcp.exe -verify coreor.edi.p7s

Примеры программной реализации

**Visual Basic**

Одним из вариантов является создание подписи с использованием COM технологии, например из Visual Basic.

Для реализации этого подхода потребуется библиотека CAPICOM (распространяется компанией Microsoft [бесплатно](http://www.microsoft.com/downloads/details.aspx?displaylang=ru&FamilyID=860ee43a-a843-462f-abb5-ff88ea5896f6))

Вместе с данной библиотекой поставляется достаточно обширный набор примеров. Основная проблема при работе с CAPICOM связа с тем, что он не может корректно обработать бинарные данные (не Unicode строки) нечетной длины. Про данную проблему и методы ее решения можно детально почитать тут:

<http://www.cryptopro.ru/cryptopro/products/csp/-faq.htm#11>

<http://www.codeproject.com/KB/security/CapicomUTF8.aspx?display=Print>

В случае отправки сообщений (EDIFACT и XML) для нашей системы это проблема может быть просто решена путем добавления пробела в конец файла для получения документа четной длины.

Вот пример подписи на Visual Basic:

Private Sub sign(ByVal SrcFile As String, ByVal DestFile As String)

' Читаем исходный файл

Dim inputData As Byte() = My.Computer.FileSystem.ReadAllBytes(SrcFile)

' Если файл сдоержит нечетное кол-во байт - нас ожидают проблемы.

' дополняем прочитанный документ пробелом

If ((inputData.Length Mod 2) = 1) Then

System.Array.Resize(inputData, inputData.Length + 1)

inputData(inputData.Length - 1) = 32 ' Space character

End If

' Выбираем сертификат для подписи, искать будем

' в личный сертификатах текущего пользователя системы

Dim Store As CAPICOM.Store = New CAPICOM.Store

Store.Open(CAPICOM.CAPICOM\_STORE\_LOCATION.CAPICOM\_CURRENT\_USER\_STORE, "My",

CAPICOM.CAPICOM\_STORE\_OPEN\_MODE.CAPICOM\_STORE\_OPEN\_READ\_ONLY)

Dim Certificates As CAPICOM.Certificates = Store.Certificates.Find

(CAPICOM.CAPICOM\_CERTIFICATE\_FIND\_TYPE.CAPICOM\_CERTIFICATE\_FIND\_KEY\_USAGE,

CAPICOM.CAPICOM\_KEY\_USAGE.CAPICOM\_DIGITAL\_SIGNATURE\_KEY\_USAGE)

If (Certificates.Count = 0) Then

MessageBox.Show("Нет сертификатов для подписи")

Return

End If

Dim SelectedCerts As CAPICOM.Certificates

' Показываем пользователю диалог для выбора сертификата для подписи

Try

SelectedCerts = Certificates.Select("Выбор сертификата",

"Выберите сертификат для создания подписи")

Catch

MessageBox.Show("Нет сертификатов для подписи")

Return

End Try

' Сертификат, при помощи которого будет производиться подписание

Dim Signer As CAPICOM.Signer = New CAPICOM.Signer

Signer.Certificate = SelectedCerts.Item(1)

' Начинаем подписывать

Dim SignedData As CAPICOM.SignedData = New CAPICOM.SignedData

' Устанавливаем данные

Dim utils As CAPICOM.Utilities = New CAPICOM.Utilities

Dim inputString As String = utils.ByteArrayToBinaryString(inputData)

SignedData.Content = inputString

' Собственно подпись. Чтобы не наступить на грабли с нечетным кол-вом байт

' Требуем выдачу сообщения в виде строки в Base64 виде

Dim outputData = SignedData.Sign(Signer, False,

CAPICOM.CAPICOM\_ENCODING\_TYPE.CAPICOM\_ENCODE\_BASE64)

' Декодируем Base64 методом, который ничего не потеряет.

Dim result() As Byte

result = System.Convert.FromBase64String(outputData)

' Сохраняем результат в файл

My.Computer.FileSystem.WriteAllBytes(DestFile, result, False)

MessageBox.Show("Message Signed")

End Sub

Проект полностью можно скачать [здесь](http://www.rlisystems.ru/ReleaseOrder/SignerVB.zip) (zip, 109 КБ).