Обзор форматов стандарта CAdES

Юрий Строжевский (http://www.strozhevsky.com)

2014 г.

Введение

Данная статья представляет собой обзор стандарта CAdES (CMS Advanced Electronic Signatures). Статья была написана как на основе теоретических исследований, проведённых автором, так и на основе написания собственной реализации создания и проверки подписей форматов CAdES.

В статье представлена информация из последней версии данного стандарта, имеющей номер 2.2.1 и опубликованной в апреле 2013-го года (все версии стандарта CAdES можно получить по следующей ссылке: http://www.etsi.org/deliver/etsi_ts/101700_101799/101733/). В настоящее время в процессе создания находится новый документ описывающий стандарт CAdES и обозначаемый "EN 319-122". Черновой вариант (draft) данного документа можно найти по следующей ссылке: http://docbox.etsi.org/esi/Open/Latest_Drafts/.

В статье будут проанализированы все основные формы электронных подписей, описанные в стандарте CAdES. Также по необходимости будут проанализированы и все атрибуты цифровых подписей, входящих в ту или иную форму CAdES.

Стандарт CAdES невозможно анализировать без первичного анализа стандарта CMS (Cryptographic Message Syntax). В связи с этим в статье также представлен первичный анализ данного стандарта, а также мотивы, связанные с появлением идей и решений, представленных в стандарте CAdES.

Для понимания статьи от читателя требуется общее знание основ создания электронно-цифровых подписей и (в меньшей мере) знакомство с общей нотацией ASN.1.

Для установления "первичного доверия" приведу несколько фактов обо мне, значимых для темы данной статьи:

- Более 15-ти лет профессионального опыта работы программистом;
- Являюсь первичным автором ПО "КриптоАрм" (написано единолично мною в 2003-ем году, вел продукт вплоть до версии 2.5, сейчас отношения к данному ПО не имею);
- Являюсь автором статей "Использование Crypto API" и "ASN.1 простыми словами";
- Являюсь активным участником форума компании "Крипто-Про" с 2005-го года;

Модель угроз для электронно-цифровых подписей формата CMS

Электронно-цифровые подписи формата CMS описаны в "интернет стандарте" STD0070 (он же RFC5652). Для начала анализа приведу общую ASN.1 схему для CMS подписей:

```
SignedData ::= SEQUENCE {
  version CMSVersion,
  digestAlgorithms DigestAlgorithmIdentifiers,
  encapContentInfo EncapsulatedContentInfo,
  certificates [0] IMPLICIT CertificateSet OPTIONAL,
  crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,
  signerInfos SignerInfos }
```

В свою очередь структура типа "SignerInfo" описывается следующей ASN.1 схемой:

```
SignerInfo ::= SEQUENCE {
  version CMSVersion,
  sid SignerIdentifier,
  digestAlgorithm DigestAlgorithmIdentifier,
  signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,
  signatureAlgorithm SignatureAlgorithmIdentifier,
  signature SignatureValue,
  unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL }
```

Для данной формы представления ЭЦП (электронно-цифровой подписи) характерно то, что большинство её атрибутов являются "не подписываемыми". То есть существует только определённый перечень атрибутов, которые включаются в состав данных при создании подписи. Остальные же атрибуты признаются вспомогательными и служат целям переноса дополнительной информации, и, как следствие, исключаются из состава подписываемых (критичных) данных.

Однако возможны ситуации, при которых замена этих "не подписываемых" атрибутов приводит к проблемам при проверке всей ЭЦП. Опишем основные из них:

- Замена (удаление) значений массива "*certificates*";
- Замена (удаление) значений массива "*crls*";
- Замена значения поля "*sid*";

Опишем более подробно все возможные критичные замены не подписываемых атрибутов ЭЦП формата CMS:

- Замена поля "*version*" структуры SignedData:
  - Данный номер версии играет малую роль и практически остаётся без применения;
  - Формально некоторые программы могут выдавать ошибку при подстановке ошибочного номера версии;
- Замена поля "*digestAlgorithms*" структуры SignedData:
  - Приложения могут выдавать ошибку проверки подписи в случае если в подписи определённого подписанта употреблён алгоритм, отсутствующий в общем списке алгоритмов (*digestAlgorithms*). Например ошибку проверки подписи в таком случае выдадут соответствующие функции Microsoft Crypto API;
- Замена поля "*certificates*" структуры SignedData:
  - Стандарт CMS говорит, что сертификат подписанта только может (но не обязан) присутствовать в массиве "*certificates*" ЭЦП CMS. То есть подразумевается, что у проверяющего данную подпись может быть возможность получить сертификат подписанта другими путями. Однако зачастую в современных программах предусмотрено, что сертификат подписанта должен быть включён в состав атрибута "*certificates*" и без его наличия ЭЦП проверена быть не может;
- Замена поля "*crls*" структуры SignedData:
  - В стандарте отсутствуют требования об обязательности присутствия достаточных доказательств подлинности для всех необходимых для проверки подписи сертификатов. То есть опять предполагается, что при проверке подписи подобные доказательства подлинности могут быть взяты из сторонних источников. Однако зачастую (особенно при автоматической

проверке ЭЦП) наличие полных доказательств подлинности при проверке подписи необходимо;

- Изменение поля "*version*" структуры SignerInfo:
  - Изменение данного поля повлияет только на обнаружение формата поля "*sid*" (сделано оно на основании "*issuer + serial*" или на основе хэша публичного ключа сертификата подписанта). Однако в дальнейшем формат поля "*sid*" можно определить по анализу строения самого поля "*sid*";
  - Формально некоторые программы могут выдавать ошибку при нахождении ошибочного номера версии;
- Изменение поля "*sid*" структуры SignerInfo:
  - В случае применения варианта построения "*sid*" по хэшу публичного ключа сертификата возможна атака связанная с подменой сертификата подписанта. Возможной эта атака становится лишь в том случае, когда для одного и того же публичного ключа одним и тем же удостоверяющим центром выдано несколько сертификатов. Так как выданные для одного и того же публичного ключа сертификаты могут отличаться политиками применения выданного сертификата, то замена в подписи одного сертификата подписанта на другой позволит скомпрометировать данную подпись;
- Изменение поля "*digestAlgorithm*" структуры SignerInfo:
  - Стандарт требует, чтобы поле "*digestAlgorithm*" содержало один из алгоритмов, содержащихся в общем поле "*digestAlgorithms*". В случае нарушения данного правила теоретически нарушается структура ЭЦП и её соответствие стандарту. Кроме того при нарушении данного правила некоторые программы проверки ЭЦП могут давать сбой. Например будет невозможно проверить ЭЦП с помощью некоторых стандартных функций Microsoft Crypto API;
- Изменение поля "*signatureAlgorithm*" структуры SignerInfo:
  - Теоретически возможна атака, при которой значение подписи, сделанное на одном криптографическом алгоритме, будет иметь корректное значение для подписи, сделанной на других входных данных и другом алгоритме. Подобная атака по моему мнению сложно реализуема, но возможна;

Кроме собственно угроз, связанных с заменой значений не подписываемых атрибутов, для CMS существует ряд угроз системного характера (то есть угрозы, которые обусловлены самой структурой ЭЦП формата CMS). Объединим все возможные угрозы ЭЦП формата CMS в укрупнённый список:

- В подписи CMS отсутствует доверенное время создания подписи;
  - Фактически это означает, что нет возможности доказать что в данный момент времени данная подписи уже существовала. В свою очередь данное доказательство необходимо в случае рассмотрения вопроса юридической значимости данной подписи в конкретно обозначенное время;
  - Существующий атрибут "*signing-time*" определяет, что время берётся из локального времени в системе, где создаётся данная подпись. Так как такое время может быть легко изменено то значение атрибута "*signing-time*" нельзя считать доверенным;
- В подписи CMS отсутствует информация, что в момент создания подписи подписант имел право создавать данную подпись;
  - Фактически это означает, что в существующем стандарте нет обязательного требования по включению полной цепочки доказательств подлинности для каждого сертификата;
  - Существующий механизм включения в подпись доказательств подлинности (на основе массива "*crls*") предусматривает только передачу значений

данных доказательств. Однако в зачастую необходимо иметь привязку наличия таких доказательств ко времени создания самой подписи. Кроме того такая привязка должна быть выполнена некой третьей, доверенной стороной, выступающей в роли арбитра;

- В подписи CMS отсутствует доверенная информация, что данная подпись была сделана именно на данном определённом сертификате;
  - Как было рассмотрено выше возможна атака на CMS, при которой меняется как значение поля "*sid*", так и значение поля "*certificates*";
- Для подписи формата CMS нет возможности долговременного хранения (установления долговременного хранения к ранее сделанной подписи);
  - Фактически это означает, что нет возможности:
    - Уберечься от компрометации использованных при создании подписи криптоалгоритмов;
    - Уберечься от компрометации использованный при создании подписи криптографических ключей;
    - Уберечься от достижения предельного срока эксплуатации для использованных при создании подписи сертификатов;
    - Уберечься от изменения в будущем структуры цепочек удостоверяющих центров;
- Для подписи формата CMS нет возможности уберечься от изъятия доказательств подлинности отдельных сертификатов;
  - Поля "*certificates*" и "*crls*" могут быть изменены в любой момент;
- Для подписи формата CMS нет информации о типе подписываемого контента;
  - Фактически для собственно подписи отсутствие информации о типе контента представляет минимальную опасность. Однако наличие подобной информации позволяет во-первых правильно обработать данные из подписи на приёмной стороне, а во-вторых обеспечить неотрекаемость при создании подписи (подписант явно знал какой тип контента он подписывает);

Общее описание форматов CAdES

В качестве решения проблем, существующих для подписи формата CMS, существует несколько способов их решения. В данной статье будут рассмотрены способы решения вышеозначенных проблем, предлагаемые стандартом CAdES.

В общем случае стандарт CAdES полностью сохраняет существующую структуру ЭЦП формата CMS, добавляя исключительно только дополнительные подписываемые или не подписываемые атрибуты. То есть сохраняется полная обратная совместимость: подпись, сделанную в формате CAdES, смогут верифицировать все программы, которые могут верифицировать подписи формата CMS. Однако всеми возможностями "усовершенствованной подписи" (подписи формата CAdES) возможно пользоваться только специализированными программами, способными анализировать дополнительные атрибуты CAdES.

В стандарте CAdES существует несколько форм представления усовершенствованной подписи. Фактически эти формы зачастую накладываются одна на другую, образуя с каждым новым "слоем" дополнительную надёжность создаваемой подписи.

Необходимо также сказать, что первые версии стандарта CAdES вышли в 2000-ом году и в настоящее время местами устарели, так как данный стандарт базируется на стандарте

CMS, которые также изменился с 2000-го года. Кроме того в стандарт CAdES периодически добавляются новые форматы и некоторые ранее описанные атрибуты и целые форматы подписей признаются устаревшими.

Кроме дополнительных атрибутов в стандарте CAdES используются данные, описанные в двух внешних стандартах:

- TSP (Time-stamp protocol, RFC3161 (+ RFC5816));
- OCSP (On-line Certificate Status Protocol, RFC2560 (RFC5019));

Стандарт TSP используется для добавления в атрибуты подписи доверенного времени, которое возвращается от внешнего доверенного сервера времени. Стандарт OCSP используется для получения мгновенного ответа о статусе выбранного сертификата.

Общее описание протоколов TSP и OCSP

Протокол TSP (Time-stamping Protocol) предназначен для получения "штампа времени" на определённые входные данные. То есть проще это можно объяснить так: есть некий доверенный внешний сервер, на котором установлен проверенный источник точного времени; есть некие данные у пользователя, на которые он хочет получить "штамп времени". Пользователь формирует запрос к TSP серверу, в который включает значение хэша данных, для которых необходимо получить "штамп времени". Получив подобный запрос TSP сервер формирует ответ, который содержит информацию о точном времени, взятом с часов TSP сервера, сам входной хэш подтверждаемого сообщения, а также цифровую подпись комбинации "время сервера + хэш сообщения". Таким образом возможно получать так называемые "штампы времени", которые во-первых дают информацию о точном и доверенном времени, а во-вторых подтверждают наличие хэша данных в этот момент времени. Получается нечто вроде "электронного нотариуса" - клиент предоставляет данные, а "нотариус" заверяет, что в такое-то время ему действительно были предоставлены эти данные. Ещё раз обращу внимание - TSP серверу передаётся хэш данных, а не сами данные.

Протокол OCSP служит для условно-мгновенного получения актуального статуса для данного сертификата. Обычно для получения статуса сертификата используют анализ CRL (Certificate Revocation List, списки отзыва сертификатов). Однако такие "списки отзыва" удостоверяющими центрами публикуются в среднем раз в день. В промежутке между выпусками CRL статус сертификата фактически неизвестен (то есть если данного сертификата нет в ранее выпущенном CRL то он вполне может быть в будущем CRL). Но для целей создания подписи важно знать именно моментальное, текущее значение статуса действия данного сертификата. В связи с этим был разработан протокол OCSP. Если проще, то принцип работы с данным протоколом можно обрисовать так: у клиента есть определённый сертификат, который нуждается в проверке статуса его действия; есть доверенный сервер, который работает по протоколу OCSP. Данный OCSP сервер имеет некую "базу статусов отзыва" для определённого набора сертификатов. При поступлении запроса от клиента сервер OCSP обращается к своей внутренней базе данных и проверяет статус отзыва данного сертификата. Потом сервер OCSP формирует ответ, содержащий время формирования данного ответа, идентификатор сертификата для которого проводилась проверка и собственно статус проверяемого сертификата. Все эти значения подписываются цифровой подписью на сертификате OCSP сервера. Формально данная схема позволяет обойти проблемы, существующие при использовании CRL. Однако на практике зачастую бывает иная ситуация: часто OCSP сервера вместо специальных "баз данных отзыва" используют информацию из всё тех же CRL. В этом случае получается, что поставщик услуг OCSP сервера берёт на себя ответственность за то, что фактически

клиенту может быть сообщён ошибочный статус сертификата. Также при использовании OCSP следует понимать, что OCSP сервер выдаёт информацию только по строго определённому набору сертификатов. Для сертификатов вне данного набора ответ OCSP сервера будет "статус не определен".

Формат подписи CAdES-BES (Basic Electronic Signature)

Данный формат подписи является простейшим из форматов, описываемых в стандарте CAdES. В то же время атрибуты, включаемые в данном формате в ЭЦП, обязательно должны присутствовать при создании подписей многих других, более сложных форматов подписей CAdES.
В формате CAdES-BES в подпись CMS добавляются следующие обязательные подписываемые атрибуты:

- *Content-type*. Фактически с внесением данного атрибута отчасти решается одна из проблем CMS (отсутствие описания типа подписываемого контента). В данном атрибуте сохраняется тип контента в виде OID (object identifier) для типа контента подписываемых данных. Данный тип атрибута изначально введён в стандарте CMS, однако в формате CAdES-BES данный атрибут приобрёл статус обязательного. Также (в дополнение к требованиям CMS) рекомендуется использование MIME кодирования для основного подписываемого контента. При таком кодировании контента дополнительно добавляется уточняющая информация по его типу (картинка, видеофайл и т.п.);
- *Message-digest*. В данном атрибуте сохраняется хэш подписываемого сообщения, выполненный на алгоритме, который использовался для создания основной подписи;
- *ESS signing-certificate-v2*. Не смотря на название данного атрибута в нём хранится не сам сертификат подписанта, а только особым образом сформированный идентификатор, который в свою очередь однозначно идентифицирует подписанта. Подробную информацию о данном атрибуте можно найти в RFC5035. Данный атрибут предназначен для предотвращения возможности подмены "sid" и значения сертификата подписанта;

Также иногда добавляется ещё один подписываемый атрибут, не входящий в число обязательных:

- *signing-time*. Данный атрибут был введён в стандарте CMS и представляет собой время создания подписи. Однако данное время берётся из локальных часов той системы, на которой создаётся подпись. Так как это время обычно легко может быть изменено, то доверять значению времени, сохранённом в данном атрибуте, как минимум опрометчиво;

Здесь необходимо сделать маленькое отступление лишь косвенно касающееся стандарта CAdES, но имеющее непосредственное отношение к реалиям российского рынка криптографии. Дело в том, что компания "Крипто-Про" начиная с версии 3.6 своего криптопровайдера автоматически по-умолчанию создаёт все ЭЦП в формате CAdES-BES. То есть все описанные выше атрибуты автоматически по-умолчанию добавляются в любую подпись, сделанную с помощью этого криптопровайдера. На самом деле это можно даже считать удобным, ибо даже ранее сделанные подписи можно относительно легко дополнить до более расширенных форматов CAdES. Более подробно почитать о данном нововведении в "Крипто-Про" CSP 3.6 можно на странице по адресу http://cpdn.cryptopro.ru/content/csp36/html/cadesattrs.html.

Формат подписи CAdES-T (Electronic Signature with Time)

Данный формат выдвигает требование обязательного наличия атрибутов, добавляемых в формате CAdES-BES, а также добавляет один новый не подписываемый атрибут:

- *signature-time-stamp*. Этот атрибут привносит в подпись информацию о доверенном времени (ответ от сервера TSP). В качестве данных для создания штампа времени используется значение самой подписи для данного подписанта. Фактически значение данного атрибута подтверждает, что в данный момент времени значение подписи было именно таким, и никаким иным. В свою очередь это позволяет проконтролировать неизменность подписи, а также четко узнать метку времени, полученную от доверенного сервера;

Формат подписи CAdES-C (Electronic Signature with Complete Data References)

Как и предыдущий формат подписи данный формат строится на основе более "младшего" формата. То есть для подписи в формате CAdES-C необходимо наличие всех атрибутов, привносимых в форматах CAdES-BES и CAdES-T.

Кроме этих атрибутов из "младших" форматов формат CAdES-C привносит два очень важных не подписываемых атрибута:

- *complete-certificate-references*. Данный атрибут является массивом специальным образом сформированных идентификаторов для всех сертификатов, которые только могут понадобиться при проверки подписи или какой-либо из её частей. В число сертификатов, для которых формируются эти идентификаторы, входят как сертификат самого подписанта так и сертификаты всех промежуточных и корневых удостоверяющих центров, а также сертификаты серверов OCSP и TSP. Хранение только идентификаторов для данных сертификатов позволяет во-первых сократить размер подписи, а во-вторых предоставляет возможность хранить сами значения сертификатов отдельно от самой подписи. В частности это полезно при создании баз данных подписей, где одни и те же сертификаты многократно используются для создания разных подписей. С помощью данного атрибута можно уберечься от возможного изменения в будущем структуры удостоверяющих центров (УЦ прекратил свою работу, изменил адрес нахождения корневого сертификата и т.п.);
- *complete-revocation-references*. В этом атрибуте также хранится массив идентификаторов, но только теперь эти идентификаторы формируются для массива доказательств подлинности (CRL или OCSP ответов), которые в свою очередь сделаны для сертификатов, необходимых для проверки данной подписи. То есть в атрибуте "*complete-revocation-references*" хранятся идентификаторы доказательств подлинности для сертификатов, идентификаторы которых хранятся в "*complete-certificate-references*". Как и в случае с "*complete-certificate-references*" хранение в данном атрибуте только идентификаторов позволяет снизить размер хранимой подписи. Наличие ссылок на доказательства подлинности для всех необходимых сертификатов позволяет облегчить процесс последующей проверки ЭЦП и устраняет влияние времени (всё необходимое для проверки сертификатов уже присутствует в подписи);

Следует обратить особое внимание на то, что атрибут "*complete-revocation-reverences*" создаётся только для двух возможных видов доказательств подлинности: CRL и OCSP ответов. Кроме того для каждого из этих видов процедура создания идентификатора сильно отличается (нет унификации между видами).

Формат подписи CAdES-X Type 1 (EXtended Electronic Signature)

Данный вид подписи формируется на основе присутствия в подписи всех атрибутов, соответствующих формату CAdES-C.

В данном формате добавляется один не подписываемый атрибут:

- *CAdES-C-time-stamp*. В данном атрибуте содержится штамп времени (ответ от доверенного сервера TSP) на массив данных, который представляет собой объединение значений для следующих атрибутов:
  - Значение подписи для соответствующего подписанта;
  - Значение ранее вычисленного атрибута "*signature-time-stamp*";
  - Значение ранее вычисленного атрибута "*complete-certificate-references*";
  - Значение ранее вычисленного атрибута "*complete-revocation-references*";
- Добавление атрибута "*CAdES-C-time-stamp*" позволяет утверждать, что в заданный момент времени значение атрибутов были именно такими и никакими иными;

Формат подписи CAdES-X Type 2

Данный вид подписи формируется на основе присутствия в подписи всех атрибутов, соответствующих формату CAdES-C.

В данном формате также добавляется только один не подписываемый атрибут:

- *CAdES-C-time-stamped-certs-crls-references*. В данном атрибуте содержится штамп времени (ответ от доверенного сервера TSP) на массив данных, который представляет собой объединение значений для следующих атрибутов:
  - Значение ранее вычисленного атрибута "*complete-certificate-references*";
  - Значение ранее вычисленного атрибута "*complete-revocation-references*";
- Как видно отличие данного формата от CAdES-X Type 1 состоит в меньшем количестве атрибутов, которые подтверждаются ответом от TSP сервера;

Формат подписи CAdES-X Long Type 1 и Type 2 (CAdES-XL)

Данные виды подписи формируется на основе присутствия в подписи всех атрибутов, соответствующих форматам CAdES-X Type 1 и CAdES-X Type 2 соответственно.

В форматах типа "CAdES-X Long" в дополнение к атрибутам добавленным в "не Long версии" добавляются два новых атрибута:

- *complete-certificate-values*;
- *complete-revocation-values*;

В данных атрибутах в подпись добавляются массивы собственно значений для сертификатов и доказательств подлинности, которые необходимы для проверки этих самых сертификатов. Должен отметить, что между атрибутами "*complete-...-references*" и "*complete-...-values*" существует строгая связь. То есть порядок следования в массиве идентификаторов (references) должен соответствовать порядку следования в массиве значений (values). Исключение составляет сертификат подписанта, который подразумевается всегда как первый сертификат в списке.

Добавление массивов значений как для сертификатов, так и для доказательств подлинности снимает проблему возможного изменения структуры удостоверяющих центров в будущем и облегчает проверку подписи в целом.

Также необходимо заметить, что массивы значений не удостоверяются третьей стороной (в данных формах подписи нет штампов времени на массивы значений, но есть на массивы идентификаторов, соответствующих данным массивам значений).

Недостатки подписей форматов от CAdES-BES до CAdES-XL

Недостатки подписей форматов от CAdES-BES до CAdES-XL можно представить следующим списком:

- Слабое использование стандартных атрибутов CMS подписи "*certificates*" и "*crls*";
    - В последних стандартах CMS предусматривается, что в массиве "*certificates*" могут быть как стандартные сертификаты формата X.509, так и произвольные другие версии контейнеров публичных ключей. Однако в стандарте CAdES, позиционирующем себя как более продвинутый стандарт по отношению к CMS, возможно использование исключительно только сертификатов формата X.509. Точно такая же ситуация и с массивом "*crls*": стандарт CMS предусматривает, что кроме хранения CRL в данном массиве могут содержаться произвольные форматы доказательств подлинности, в том числе в массиве "*crls*" могут содержаться и OCSP ответы. Однако в стандарте CAdES работа с массивом "*crls*" ведётся из расчета, что в нём хранятся исключительно только стандартные списки отзыва (CRL), а для хранения OCSP ответов предусматриваются дополнительные структуры описания и дополнительные атрибуты;
- Сложная процедура составления идентификаторов (references);
- Использование атрибута "*signing-time*", в котором содержится не доверенное время создания подписи;
- Использование избыточного подписываемого атрибута "*message-digest*". Значение атрибута "*message-digest*" совместно со значением атрибута "*signing-time*" подтверждает существование данного хэша подписываемого сообщения в данное время. Однако так как в атрибуте "*signing-time*" содержится не доверенное время, то смысл наличия атрибута "*message-digest*" полностью теряется;
- Использование атрибутов "*complete-certificate-value*" и "*complete-revocation-values*" отдельно для каждого подписанта существенно увеличивает общий размер подписи;
    - При наличии в одной подписи нескольких подписантов (или наличии со-подписей) крайне вероятно, что для каждого из подписантов значения в этих атрибутах будут практически одинаковыми;
- Фактически атрибуты "*complete-certificate-values*" и "*complete-certificate-references*" строятся для разных наборов данных. То есть в то время как в атрибуте "*complete-certificate-values*" хранятся все сертификаты, включая сертификат подписанта, в атрибуте "*complete-certificate-references*" хранятся все идентификаторы сертификатов, исключая сертификат подписанта. Подразумевается, что идентификатор сертификата подписанта хранится в отдельном атрибуте - "*ESS signing-certificate-v2*". Подобный подход затрудняет как формирование подписи CAdES, так и её проверку;
- Формат атрибута "*complete-revocation-references*" жестко задан, и определяет, что первично идут доказательства подлинности для сертификата подписанта, а потом -

доказательства подлинности для всех остальных сертификатов, причём именно в том порядке, в котором они были отражены в атрибуте "*complete-certificate-references*". Это также создаёт дополнительную сложность при создании подписей CAdES;

- Форматы атрибутов "*complete-revocation-references*" и "*complete-revocation-values*" отличаются тем, что в "*complete-revocation-references*" элементами массива являются идентификаторы доказательств подлинности в разрезе каждого сертификата, тогда как в "*complete-revocation-values*" все значения доказательств подлинности хранятся без группировки по сертификатам. Отсутствует унификация атрибутов, затрудняется получение подписи CAdES и последующая проверка таких подписей;

Форматы подписи CAdES-A (Archival)

Первоначально формат подписи CAdES-A создавался для того, чтобы сохранять возможность проверки подписи в течение как можно более долгого периода времени (обычный период возможности проверки подписи, сделанной в формате CAdES-A составляет 15 лет).

Но в последних версиях стандарта формат CAdES-A v3 предлагается как наиболее предпочтительный из всех предыдущих форматов CAdES, устраняя практически все недостатки CAdES ранее рассмотренные в этой статье. То есть данный формат уже не ограничен только целями архивного (длительного) хранения подписей, а обеспечивает более удобный формат "улучшенной подписи" в целом. Именно формат CAdES-A v3 в последних версиях стандарта фактически вытеснил почти все другие форматы CAdES.

Однако для понимания форматов серии CAdES-A необходимо первично начать рассмотрение с наиболее ранних форм этого формата.

Первая версия формата CAdES-A была представлена в 2000-ом году и была достаточно плохо проработанной и сложной в применении. Впоследствии вышла вторая версия данного формата (CAdES-A v2).

Формат CAdES-A v2 строится на основе наличия атрибутов, добавленных в форматах CAdES-XL. Собственно в самом формате CAdES-A v2 добавляется только один новый не подписываемый атрибут:

- *archive-time-stamp*. Данный атрибут представляет собой штамп времени на данные, образованные соединением следующих значений:
  - Значением собственно подписываемых данных;
  - Значениями атрибутов "*certificates*" и "*crls*", когда эти атрибуты присутствуют;
  - Значениями всех элементов последовательности SignerInfo для данного подписанта, включая подписываемые и не подписываемые атрибуты;
- Так как атрибутов "*archive-time-stamp*" для одного и того же подписанта может быть несколько (например когда сохраняются штампы времени от нескольких TSP серверов, или когда значение старого атрибута обновляется новым), то перед вычислением значения штампа времени из числа не подписываемых атрибутов убираются все атрибуты типа "*archive-time-stamp*";
- Новые атрибуты "*archive-time-stamp*" могут добавляться в подпись по мере необходимости (например в случае компрометации ранее использованных криптографических алгоритмов или сертификатов);

Проблемы формата CAdES-A v2:

- Подпись формата CAdES-A v2 строится на основе атрибутов, добавленных в форматах CAdES-XL. То есть для построения этого вида архивной подписи необходимо также создать множество других (по сути вспомогательных) атрибутов CAdES, что затрудняет процесс создания архивной подписи;
- Так как атрибут "*archive-time-stamp*" использует полные последовательности "*certificates*" и "*crls*" без дополнительной обработки, то для компрометации атрибуту "*archive-time-stamp*" (а следовательно и формата CAdES-A v2 в целом) достаточно лишь изменить порядок следования сертификатов и доказательств подлинности, сохранённых в не подписываемых атрибутах подписи;
- В данном формате CAdES полностью отсутствует использование идентификаторов как для сертификатов, так и для доказательств подлинности (сертификаты и доказательства подлинности используются только как некие бинарные массивы данных, без рассмотрения их внутренней структуры). В следствие этого затруднено хранение атрибутов "*archive-time-stamp*" отдельно от подписи так как вместе с атрибутом "*archive-time-stamp*" необходимо хранить также и массивы сертификатов и доказательств подлинности. Причём для каждой подписи данные массивы надо хранить отдельно (например нет возможности хранить отдельную таблицу сертификатов и использовать только идентификаторы этих сертификатов);
- Поскольку стандарт CMS изначально опирается на правила кодирования ASN.1 BER (правила ASN.1 DER обязательны для использования только внутри подписываемых атрибутов), то возможно возникновение неоднозначности при кодировании данных (с помощью ASN.1 BER одни и те же данные можно закодировать несколькими различными способами). Так как формат CAdES-A v2 использует не подписываемые атрибуты сообщения, и, кроме того, использует эти атрибуты как обычные бинарные массивы, то возможно изменение хэшей просто в следствие различия в способе кодирования;
- В более ранних версиях стандарта CAdES формат CAdES-A v2 был описан с неточностью. В более новых версиях стандарта были даны уточнения. Однако в настоящий момент возможно существование подписей с двумя вариантами формата CAdES-A v2 в которых хэши вычисляются над различными данными. Стандарт рекомендует при проверке подписей данного формата вычислять два хэша и сравнивать их с существующими значениями. Это, в свою очередь, также затрудняет использование формата CAdES-A v2;

Следующим форматом "архивной" подписи стал CAdES-LT (Long Term). Данный формат может строится на основе наличия атрибутов форматов CAdES начиная с CAdES-T и заканчивая CAdES-A v2. В формате CAdES-LT добавляется один не подписываемый атрибут "*long-term-validation*". Атрибут "*long-term-validation*" представляет собой штамп времени над последовательностью, в которую включаются следующие данные:

- Дата добавления данного атрибута в подпись;
- Штамп времени или поле типа "*EvidenceRecord*" (RFC4998);
- Дополнительные значения для массива "*certificates*" (дополнительные значения имеют тот же тип "*CertificateChoice*");
- Дополнительные значения для массива "*crls*" (дополнительные значения имеют тот же тип "*RevocationChoice*");

Первой отличительной особенностью формата CAdES-LT является использование технологии "*tree-hashing*" (RFC4998) для вычисление хэша для штампа времени. Для вычисления штампа времени хешируются следующие поля:

- *eContentType*. Тип подписанного контента;

- eContent. Используются либо присоединённые к подписи подписанные данные, либо внешние данные, если подпись отсоединённая;
- Значения массива "*certificates*", дополненные значениями массива "*extraCertificates*" из атрибута "*long-term-validation*". Хеширование сводного массива осуществляется по технологии "*tree-hashing*";
- Значение массива "*crls*", дополненные значениями массива "*extraRevocation*" из атрибута "*long-term-validation*". Хеширование сводного массива осуществляется по технологии "*tree-hashing*";
- Все поля структуры "*signerInfos*". При вычислении хэша над "*unsignedAttrs*" применяется технология "*tree-hashing*";

Преимущества применения технологии "*tree-hashing*":
- Уменьшается общее количество хэшей;
- За счет сортировки по бинарным данным, применяемой перед вычислением общего значения хэша, устраняется проблема возможной перестановки внутри хэшированных данных (например теперь можно переставлять значения внутри "*certificates*" - на значение выходного хэша это никак не повлияет);

Второй отличительной особенностью формата CAdES-LT является использование значений типа "*EvidenceRecord*". В состав значений данного типа включаются все необходимые доказательства существования атрибутов подписи. Однако для целей реального применения данный тип значений представляется трудным к использованию и дальнейшей проверке.

Необходимо упомянуть, что можно создавать несколько атрибутов формата CAdES-LT внутри одной подписи (например при компрометации ранее использованных криптоалгоритмов). Однако после добавления первого атрибута "*long-term-validation*" в подпись далее в подпись могут добавляться только новые атрибуты "*long-term-validation*", и никакие другие.

В качестве проблем формата CAdES-LT можно упомянуть:
- Относительную сложность вычисления хешей;
- Возможная необходимость перегенерации некоторых элементов подписи с использованием набора правил ASN.1 DER (если они были сделаны с использованием других наборов правил);
- Относительно сложно восстановить идентификаторы использованных при создании сертификатов и доказательств подлинности. Другими словами: если доказательства подлинности и сертификаты хранятся отдельно от самой подписи (например в некой общей базе данных), то только по атрибуту из CAdES-LT достаточно сложно восстановить какие именно были использованы сертификаты и доказательства подлинности именно в этом криптографическом сообщении;

В настоящее время форматы CAdES-A v2 и CAdES-LT признаны устаревшими. Также в последних версиях стандарта фактически отказываются от использования форматов начиная с CAdES-C и заканчивая CAdES-XL. Вместо всех этих форматов предлагается использовать наиболее новый формат - CAdES-A v3.

Формат CAdES-A v3 может быть построен на основании существующих атрибутов из форматов CAdES-BES, CAdES-T, CAdES-C, CAdES-X, CAdES-XL или CAdES-A v2. В формате CAdES-A v3 добавляется два новых атрибута: "*ats-hash-index*" и "*archive-time-stamp-v3*". Однако практически в подпись добавляется только лишь атрибут "*archive-time-*

stamp-v3", а атрибут "*ats-hash-index*" используется как вспомогательный при создании "*archive-time-stamp-v3*".

Атрибут "*ats-hash-index*" представляет из себя последовательность из четырёх полей. Первое поле представляет собой OID использованного для хэширования алгоритма. Второе поле представляет собой последовательность из хэшей, сделанных отдельно над каждым элементом из массива "*certificates*". Третье поле представляет собой последовательность из хэшей, сделанных отдельно над каждым элементом из массива "*crls*". Четвёртое поле представляет собой последовательность из хэшей, сделанных отдельно над каждым элементом из массива не подписываемых атрибутов.

Атрибут "*archive-time-stamp-v3*" представляет из себя штамп времени, сделанный для следующих данных:

- "*eContentType*" из поля "*encapContentInfo*" поля SignedData;
- Хэш подписываемых данных. Для вычисления хэша используется тот же алгоритм, который будет использоваться для вычисления штампа времени для "*archive-time-stamp-v3*";
- Поля "*version*", "*sid*", "*digestAlgorithm*", "*signedAttrs*", "*signatureAlgorithm*", "*signature*" из поля "*signerInfo*";
- Полное значение атрибута "*ats-hash-index*";

В качестве преимуществ формата CAdES-A v3 можно назвать:

- Введение идентификаторов как для каждого использованного сертификата, так и для каждого использованного доказательства подлинности. Причём идентификаторы вычисляются простым и понятным образом - путём вычисления хэша над общим закодированным ASN.1 значением. Использование идентификаторов позволяет хранить значения сертификатов и доказательств подлинности отдельно от самой подписи (например в некой сводной таблице сертификатов и доказательств подлинности) и при проверке подписи CAdES по существующим идентификаторам получать реальные значения;
    - Следует также упомянуть, что идентификаторы сертификатов и доказательств подлинности, используемые в более ранних форматах CAdES, строятся по совершенно другому принципу. В связи с этим можно считать CAdES-A v3 форматом не совместимым с ранее используемыми;
- Решается проблема с компрометацией подписей путём простой перестановки значений внутри массивов "*certificates*" и "*crls*". Так как храняться все хэши для каждого сертификата, то легко можно восстановить начальную последовательность элементов внутри этих массивов;
- Хэширование значений выполняется по прозрачным алгоритмам. Хотя по сравнению с форматом CAdES-LT формат CAdES-A v3 имеет потенциально больший объём (в байтах), однако вместе с тем работать с форматом CAdES-A v3 значительно проще;

Использование формата CAdES-A v3 решает практически все проблемы CAdES, обозначенные в данной статье ранее. Опишем ещё раз проблемы устаревших форматов CAdES и решения, предложенные в CAdES-A v3:

Проблема в устаревших форматах	**Решение в CAdES-A v3**
Слабое использование стандартных атрибутов CMS подписи "*certificates*" и "*crls*"	В отличие от устаревших форматов CAdES-A v3 полностью использует типы из CMS "CertificateChoice" и "RevocationChoice".
Сложная процедура составления	Используется упрощённая процедура

идентификаторов (references);	составления идентификаторов на основе хеширования полного закодированного ASN.1 значения. Проблему следует считать решенной.
Использование атрибута "*signing-time*", в котором содержится не доверенное время создания подписи	Так как CAdES-A v3 всё ещё строится на основе CAdES-BES то эту проблему можно считать не решённой.
Использование избыточного подписываемого атрибута "*message-digest*".	Точно также и эту проблему можно считать не решённой.
Использование атрибутов "*complete-certificate-value*" и "*complete-revocation-values*" отдельно для каждого подписанта существенно увеличивает общий размер подписи.	В CAdES-A v3 используются стандартные массивы "*certificate*" и "*crls*". Причём при необходимости эти массивы могут дополняться новыми элементами, что никак не скажется на ранее добавленных атрибутах CAdES-A v3.
Фактически атрибуты "*complete-certificate-values*" и "*complete-certificate-references*" строятся для разных наборов данных.	Так как в CAdES-A v3 используется хэш отдельно для каждого элемента из массива "*certificate*", то данную проблему можно считать решённой.
Формат атрибута "*complete-revocation-references*" жестко задан.	Так как в CAdES-A v3 все доказательства подлинности (как CRL, так и OCSP ответы) сохраняются в общем массиве "crls", так же так как идентификаторы для элементов данного массива вычисляются относительно просто, то данную проблему следует считать решенной.
Форматы атрибутов "*complete-revocation-references*" и "*complete-revocation-values*" отличаются.	В формате CAdES-A v3 данная проблема изначально отсутствует.

Заключение

В качестве заключения хочу рассказать о текущем положении дел в России с использованием подписей формата CAdES.

Наиболее активно данный формат продвигает компания "Крипто-Про". В настоящий момент у данной компании есть собственная реализация создания подписей CAdES-BES и CAdES-X Long Type 1. Также данной компанией предложен некий российский стандарт "усовершенствованной подписи", по сути копирующий описание начального стандарта CAdES-X Long Type 1. Кроме того реализованы также программные решения по созданию серверов OCSP и "Time-stamping" (TSP).

Однако хотелось бы обратить внимание, что стандарт CAdES-X достаточно давно (уже около года назад) был заменён на CAdES-A v3. Надеюсь, что после этой статьи компании (в частности и "Крипто-Про") обратят своё внимание на актуальные версии стандарта CAdES и постараются использовать наработки этого стандарта в создании более совершенного российского варианта "усовершенствованной подписи".