Internet Engineering Task Force (IETF)
Request for Comments: 7030
Category: Standards Track
ISSN: 2070-1721

M. Pritikin, Ed.
Cisco Systems, Inc.
P. Yee, Ed.
AKAYLA, Inc.
D. Harkins, Ed.
Aruba Networks

October 2013

Enrollment over Secure Transport

Зачисление через защищённый транспорт

Аннотация

Этот документ описывает зачисление (регистрацию) клиентов с использованием управления сертификатами через сообщения CMS (Certificate Management over CMS или CMC) по защищённому транспорту. Этот профиль называется зачислением через защищённый транспорт (Enrollment over Secure Transport или EST) и описывает простой но полнофункциональный протокол управления сертификатами, ориентированный на клиентов инфраструктуры открытых ключей (Public Key Infrastructure или PKI), которым нужно получать сертификаты клиентов и соответствующие сертификаты удостоверяющих центров (Certification Authority или CA). Протокол также поддерживает генерируемые клиентом или CA пары ключей (открытый и секретный).

Статус документа

Документ относится к категории Internet Standards Track.

Документ является результатом работы IETF[1] и представляет согласованный взгляд сообщества IETF. Документ прошёл открытое обсуждение и был одобрен для публикации IESG[2]. Дополнительную информацию о стандартах Internet можно найти в разделе 2 RFC 5741.

Информацию о текущем статусе документа, ошибках и способах обратной связи можно найти по ссылке http://www.rfc-editor.org/info/rfc7030.

Оглавление

[1]Internet Engineering Task Force - комиссия по решению инженерных задач Internet.
[2]Internet Engineering Steering Group - комиссия по инженерным разработкам Internet.

1. Введение

Этот документ описывает зачисление сертификатов для клиентов с использованием сообщений CMC [RFC5272] по защищённому транспорту. Протокол EST описывает применение протокола защиты транспортного уровня (Transport Layer Security или TLS) 1.1 [RFC4346], 1.2 [RFC5246] или будущей версии и протокола передачи гипертекста (Hypertext Transfer Protocol или HTTP) [RFC2616] для обеспечения аутентифицированных каналов с проверкой полномочий для запросов и откликов PKI [RFC5272].

Архитектурно служба EST размещается между CA и клиентом и выполняет несколько функций, традиционно выделенных роли агентства по регистрации (Registration Authority или RA) в PKI. Природа взаимодействия между сервером EST и CA не рассматривается в этом документе.

EST применяет модель протокола управления сертификатами (Certificate Management Protocol или CMP) [RFC4210] для обновления сертификата CA, но не использует синтаксис и протокол сообщений CMP. Серверы EST расширяемы в части задания новых функций, обеспечивающих дополнительные возможности, отсутствующие в CMC [RFC5272], и этот документ задаёт два таких расширения - одно для запроса атрибутов Certificate Signing Request, другое для запроса созданных сервером ключей.

EST указывает способы защищённой передачи сообщений по протоколу HTTP через TLS (HTTPS) [RFC2818], где заголовки и типы носителей HTTP используются в сочетании с TLS. HTTPS работает по протоколу TCP и этот документ не задаёт EST через HTTP/DTLS[1]/UDP[2]. При подходящей спецификации для комбинации HTTP, DTLS, UDP не возникает требований к EST, которые мешали бы работе на основе такого стека. На рисунке 1 показаны уровни EST.

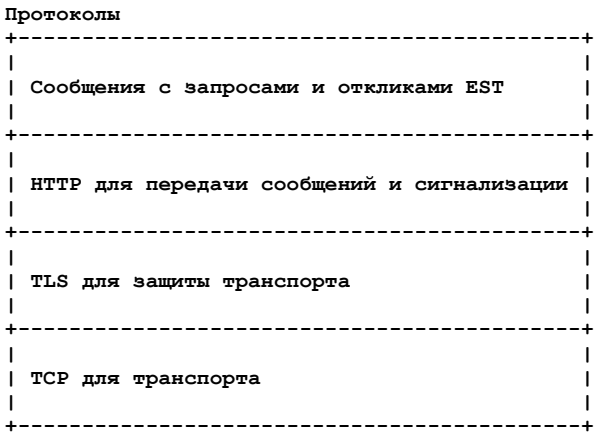

Рисунок 1. Уровни EST.

[1]Datagram Transport Layer Security - защита уровня транспортировки дейтаграмм.
[2]User Datagram Protocol - протокол пользовательских дейтаграмм.

1.1. Терминология

Ключевые слова **необходимо** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не следует** (SHALL NOT), **следует** (SHOULD), **не нужно** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **не рекомендуется** (NOT RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе интерпретируются в соответствии с [RFC2119].

Предполагается, что читатель знаком с терминами и концепциями, описанными в стандарте криптографии с открытым ключом (Public Key Cryptography Standard или PKCS) #10 [RFC2986], HTTPS [RFC2818], CMP [RFC4210], CMC [RFC5272][RFC5273][RFC5274], TLS [RFC4346]. В дополнение к терминологии, заданной в CMC [RFC5272], здесь определён ещё ряд терминов.

EST CA
: Для служб выпуска сертификатов доступ к EST CA осуществляется через сервер EST, CA может размещаться логически «за ним» или быть его частью.

Third-Party Trust Anchor - сторонняя привязка доверия
: Любая привязка доверия (trust anchor или TA), не являющаяся полномочной для иерархии PKI, где сервер EST предоставляет услуги.

Explicit Trust Anchor - явная привязка доверия
: Любая привязка TA, явно заданная на клиенте или сервере для использования при аутентификации EST TLS, например, TA, заданная вручную на клиенте EST или получаемая при начальной загрузке, как описано в параграфе 4.1.1 (см. 3.6. Полномочия сервера и 6. Вопросы безопасности).

Implicit Trust Anchor - неявная привязка доверия
: Любая сторонняя привязка TA, доступная клиенту или серверу при аутентификации TLS, но не указанная специально для применения при аутентификации EST TLS, например, привязки TA, обычно используемые web-браузерами для аутентификации web-серверов, или привязки, обычно используемые серверами для аутентификации установленных производителем свидетельств (credential) клиента, таких как сертификаты, устанавливаемые при производстве кабельных модемов и маршрутизаторов. Модель предоставления полномочий для таких TA отличается от модели, используемой для явных привязок доверия (см. 3.6.1. Клиент, использующий базу Explicit TA, 3.6.2. Клиент, использующий базу Implicit TA, 6. Вопросы безопасности).

Certificate-Less TLS - TLS без сертификатов
: Шифры TLS без сертификатов позволяют выполнить взаимную аутентификацию в случаях отсутствия сертификатов у клиента и сервера или нежелания применять такие сертификаты. В качестве свидетельства применяется слово, фраза, код или ключ, известные клиенту и серверу. Свидетельство должно быть однозначно предоставлено клиенту и серверу, чтобы можно было проверить подлинность отдельного клиента на отдельном сервере.

2. Обзор вариантов применения

В этом разделе представлен информационный обзор вариантов применения для лучшего понимания читателями обсуждения протокола.

Клиенты и серверы EST настраиваются с использованием сведений, обеспечивающих взаимную проверку подлинности и предоставление полномочий. Конкретные данные инициализации зависят от доступных клиенту и серверу методов и могут включать общие секреты, имена и местоположение сетевых служб (например, URI[1] [RFC3986]), сведения о привязках доверия (например, сертификат CA или хэш сертификата TA), ключи и сертификаты зачисления. В зависимости от принятой на предприятии практики закупок и управления сетью часть инициализации может выполнять производитель до поставки клиентского оборудования и программ. В этом случае производитель может предоставлять предприятию данные, такие как привязки доверия, по защищённой процедуре, но это выходит за рамки документа.

Распространение привязок доверия и других сертификатов возможно через сервер EST, однако о подлинности таких данных нельзя судить без наличия автономного (out-of-band) механизма их проверки.

В параграфах 2.1 - 2.3 очень точно отражён текст приложения «Сценарии» из [RFC6403] с изменениями, подходящими для этого профиля. В параграфах 2.1 - 2.6 рассмотрен набор функций EST (см. Рисунок 5) и представлен информационный обзор возможностей EST.

Процесс взаимодействия между клиентом и сервером представлен ниже.

Клиент инициирует защищённую TLS сессию HTTP с сервером EST.

Запрашивается конкретный сервис EST на основе части URI, использованного для сессии.

Клиент и сервер проверяют подлинность друг друга.

Клиент проверяет полномочия сервера на обслуживание клиента.

Сервер убеждается, что клиент уполномочен использовать этот сервер и проверяет сделанный тем запрос.

Сервер действует по запросу клиента.

2.1. Получение сертификатов CA

Клиент EST может запросить копии текущих сертификатов EST CA у сервера EST. Предполагается, что клиент EST выполняет эту операцию до каких-либо иных операций.

В этом документе предполагается, что EST CA имеет сертификат, используемый клиентом для проверки подписанных объектов, выпущенных CA, например, сертификатов и списков отзыва сертификатов (certificate revocation list или CRL), и отличия сертификата от применяемого для проверки подписей сертификатов и CRL, используемых, когда взаимодействия протокола EST требуют дополнительного шифрования.

[1] Uniform Resource Identifier - унифицированный идентификатор ресурсов.

Клиент EST проверяет подлинность и сферу полномочий сервера EST при запросе текущих сертификатов CA. Ниже перечислены доступные варианты, описанные в параграфах 3.3.1. Аутентификация сервера на основе TLS и 3.3.3. Взаимная аутентификация TLS без сертификатов.

- Проверка соответствия HTTPS URI сервера EST сертификату сервера EST с использованием Implicit TA (как при обычном обмене HTTPS). Это позволяет клиенту и серверу EST использовать имеющиеся TA, которые могут быть известны клиенту EST.
- Клиент может использовать ранее распространённую привязку доверия, связанную с сервером EST. Это позволяет клиенту EST использовать имеющийся (возможно, более старый) сертификат CA для запроса текущего сертификата CA.
- При начальной загрузке клиент EST может полагаться на взаимную аутентификацию, выполняемую конечным пользователем в соответствии с параграфом 4.1.1. Распространение сертификатов CA при начальной загрузке.
- Клиент может использовать привязку общих свидетельств (credential) к конкретному серверу EST с шифрами TLS без применения сертификатов.

Проверка подлинности клиента не требуется для этого обмена, поэтому он тривиально поддерживается сервером EST.

2.2. Исходное зачисление

После аутентификации сервера EST и проверки его полномочий на предоставление услуг клиенту EST-клиент может получить сертификат для себя, передавая этому серверу запрос на зачисление.

Сервер EST аутентифицирует и предоставляет полномочия клиенту EST, как указано в параграфах 3.2. Уровень HTTP, 3.3.3. Взаимная аутентификация TLS без сертификатов, 3.7. Полномочия клиента. Описанные в нормативном тексте методы, упомянутые в этом обзоре, включают:

- TLS с ранее выданным сертификатом клиента (например, имеющийся сертификат от EST CA);
- TLS с ранее установленным сертификатом (например, сертификат, установленный производителем или выпущенным сторонней организацией);
- TLS без сертификатов (например, общее свидетельство, распространённое по отдельному каналу);
- HTT с использованием имени и пароля, распространённых по отдельному каналу (out-of-band).

2.2.1. Аутентификация TLS с сертификатом

Если у клиента EST имеется установленные ранее сертификат, выпущенный сторонним CA, этот сертификат может применяться для проверки подлинности запроса клиентом сертификата у сервера EST (если CA признается этим сервером). Клиент EST отвечает сообщение сервера EST с запросом сертификата TLS с использованием уже имеющегося у него сертификата. Сервер EST проверяет этот сертификат и предоставит полномочия клиенту, как указано в параграфе 3.3.2. Аутентификация клиента на основе TLS.

2.2.2. Аутентификация TLS без сертификата

Клиент и сервер EST могут проверить подлинность друг друга с применением шифров TLS без сертификатов (3.3.3. Взаимная аутентификация TLS без сертификатов).

2.2.3. Аутентификация клиента на основе HTTP

Сервер EST может запросить предоставление клиентом EST имени пользователя и пароля с использованием методов аутентификации HTTP Basic или Digest (3.2.3. Аутентификация клиента на основе HTTP). Такой подход желателен, если подлинность клиента EST не удалось проверить при согласовании TLS (3.3.2. Аутентификация клиента на основе TLS) или правила сервера EST требуют для аутентификации дополнительных сведений, как указано в параграфе 3.2.3. В любом случае аутентификация на основе HTTP выполняется только через транспорт с защитой TLS (3.3. Уровень TLS).

2.3. Повторный выпуск сертификата клиента

Клиент EST может обновить имеющийся сертификат или его ключи путём запроса у сервера EST повторного зачисления.

Когда текущий сертификат клиента EST может служть для аутентификации клиента TLS (3.3.2. Аутентификация клиента на основе TLS), клиент представляет этот сертификат серверу EST для своей аутентификации. Когда перевыпущенный сертификат клиента EST не подходит для аутентификации клиента TLS, можно использовать любой метод аутентификации, применённый для начального зачисления. Например, если у клиента имеется дополнительный сертификат, выданный EST CA, который можно использовать для аутентификации клиента TLS, подойдёт этот сертификат. Сообщение с запросом сертификата включает те же значения Subject и SubjectAltName, что и текущий сертификат. Замена имени запрашивается в соответствии с параграфом 4.2.2. Простое перезачисление клиентов.

2.4. Генерация ключей на сервере

Клиент EST может запросить генерируемый сервером сертификат и ключи (4.4. Генерация ключей на стороне сервера).

2.5. Сообщения Full PKI Request

Сообщения Full PKI Request [RFC5272] могут доставляться через EST с использованием функции запроса полного CMC (Full CMC Request). Это даёт доступ к функциям, не предоставляемым простым зачислением (Simple Enrollment). Сообщения Full PKI Request определены в параграфах 3.2 и 4.2 [RFC5272]. Использование EST для транспортировки этих сообщений описано в параграфе 4.3. Полные сообщения CMC.

2.6. Запрос атрибутов CSR

Перед отправкой запроса на зачисление серверу EST клиент EST может запросить у сервера EST набор дополнительных атрибутов, которые клиент будет использовать при последующих запросах на зачисление.

Эти атрибуты могут предоставить дополнительные описательные сведения, к которым сервер EST сам по себе не имеет доступа, такие как MAC[1]-адрес интерфейса у клиента EST. Кроме того, эти атрибуты могут указывать тип запроса на зачисление, например, конкретную эллиптическую кривую или хэш-функцию, которую клиент предположительно будет применять при генерации CSR.

3. Устройство и уровни протокола

На рисунке 2 представлено расширение рисунка 1, описывающее использование уровней. Каждый из аспектов более подробно рассматривается ниже.

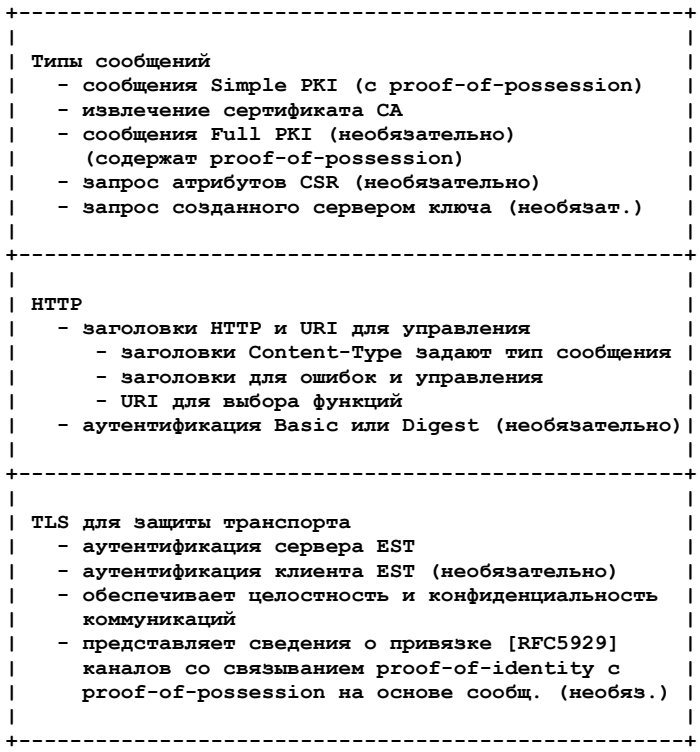

Рисунок 2. Использование протоколов на уровнях EST.

Задание HTTPS как защищённого транспорта для сообщений зачисления вносит два «уровня» передачи сообщений аутентификации и управления - TLS и HTTP. Уровень TLS обеспечивает защиту целостности и конфиденциальности транспорта. Подтверждение идентификации (proof-of-identity) обеспечивается аутентификацией согласования TLS и может обеспечиваться также заголовками уровня HTTP. Тип сообщения и связанные с ошибками и управлением сообщения включаются в заголовки HTTP.

В CMC (параграф 3.1 в [RFC5272]) отмечено, что «Simple PKI Request **недопустимо** применять, если нужно включение proof-of-identity». Поскольку уровни TLS и HTTP могут подтверждать идентификацию для клиентов и серверов EST, применяются типы сообщений Simple PKI.

Обмен сертификатами на уровне TLS обеспечивает метод проверки полномочий для запросов зачисления клиента с использованием имеющихся сертификатов. Такие сертификаты могут быть выпущены CA (у которого клиент запрашивает сертификат) или другой инфраструктурой PKI (например, свидетельство IEEE 802.1AR IDevID[2] [IDevID]).

Доказательство владения (proof-of-possession или POP) отличается от proof-of-identity и включается в сообщения типа Simple PKI, как описано в параграфе 3.4. Подтверждение владения. Связывание proof-of-identity и proof-of-possession описано в параграфе 3.5. Связывание отождествления и сведений POP.

Этот документ также задаёт транспорт для CMC [RFC5272], который соответствует протоколам доставки CMC (CMC Transport Protocols) [RFC5273]. Механизмы CMC POP и proof-of-identity заданы в CMC, а описанные здесь механизмы могут применяться вместе с ними в сообщениях Full PKI.

В обменах протокола могут применяться разные сертификаты. На рисунке 3 приведён информационный обзор. Конечные объекты могут иметь один или несколько сертификатов каждого из типов, указанных на рисунке 3, и используют одну или несколько баз данных о привязках доверия, указанных на рисунке 4.

Сертификат	*Эмитент*	*Применение и описание*
Сертификат сервера EST	CA, обслуживаемый сервером EST	Представляется сервером EST при согласовании TLS. 3.2.1. Заголовки HTTP для управления.
Сертификат сервера EST	CA, аутентифицируемый сторонней TA, например, CA web-сервера	Представляется сервером EST при согласовании TLS. 3.2.1. Заголовки HTTP для управления и 6. Вопросы безопасности.
Сторонний сертификат клиента EST	CA, аутентифицируемый сторонней TA, например, от изготовителя устройства	Представляется серверу EST ещё не зачисленным клиентом EST. 3.3.2. Аутентификация клиента на основе TLS.

[1]Media Access Control - управление доступом к среде.
[2]Initial Device Identifier - исходный идентификатор устройства.

Сертификат клиента EST	CA, обслуживаемый сервером EST	Представляется серверу EST в будущих операциях EST. 3.3.2. Аутентификация клиента на основе TLS.
Сертификат конечного элемента	CA, обслуживаемый сервером EST	Клиент может представить сертификаты, которые предназначены для применений, не связанных с EST, включая сертификаты, которые не могут применяться для операций EST. 4.2.3. Отклик на простое зачисление или перезачисление.

Рисунок 3.Сертификаты и их применение.

База данных TA	***Применение и описание***
База Explicit TA сервера EST	Серверы EST используют эту базу TA для проверки подлинности сертификатов, выпущенных EST CA, включая сертификаты клиентов EST а процессе зачисления. 3.2. Уровень HTTP
База Implicit TA сервера EST	Серверы EST используют эту базу TA для проверки подлинности сертификатов, выпущенных сторонними TA, например, сертификаты клиентов EST, выпущенные изготовителем устройства. База данных Implicit TA может быть отключена. 3.3.2. Аутентификация клиента на основе TLS
База Explicit TA клиента EST	Клиенты EST используют эту базу TA для проверки подлинности сертификатов, выпущенных EST CA, включая сертификаты сервера EST. 3.1. Уровень приложений, 3.3.1. Аутентификация сервера на основе TLS, 3.6.1. Клиент, использующий базу Explicit TA, 4.1.1. Распространение сертификатов CA при начальной загрузке
База Implicit TA клиента EST	Клиенты EST используют эту базу TA для проверки подлинности сервера EST, которые использует внешние сертификаты. База данных Implicit TA может быть отключена. 3.1. Уровень приложений, 3.3.1. Аутентификация сервера на основе TLS, 3.6.2. Клиент, использующий базу Implicit TA, 6. Вопросы безопасности

Рисунок 4.Базы привязок доверия и их использование.

3.1. Уровень приложений

Клиент EST **должен** быть способен генерировать и анализировать сообщения Simple PKI (4.2. Функции для запроса сертификатов клиентами). Генерация и разбор сообщений Full PKI **необязательны** (4.3. Полные сообщения CMC). Клиент также **должен** быть способен запрашивать сертификаты CA у сервера EST и анализировать полученные «плохие» сертификаты (4.1. Распространение сертификатов CA). Запрос атрибутов CSR и анализ возвращённого списка атрибутов **необязательны** (4.5. Атрибуты CSR).

Детали настройки клиентских приложений EST выходят за рамки обсуждения протокола, но нужны для понимания предварительных условий инициирования протокольных операций. **Рекомендуется** настроить для клиента EST базы данных TA (3.3.1. Аутентификация сервера на основе TLS) или секретный ключ (3.3.3. Взаимная аутентификация TLS без сертификатов). Соответствующие этому стандарту реализации **должны** обеспечивать возможность указания Explicit TA. Для удобства пользователей можно задать «отпечаток» Explicit TA для начальной загрузки (4.1.1. Распространение сертификатов CA при начальной загрузке). Настройка базы данных Implicit TA, возможно, путём её включения в дистрибутив клиента EST или из операционной системы, обеспечивает гибкость наряду с предостережениями, указанными в разделе 6. Вопросы безопасности. Соответствующие этому стандарту реализации **должны** обеспечивать возможность отключения использования базы данных Implicit TA.

Клиент EST настраивается с информацией, достаточной для формирования URI сервера EST. Это может быть полный сегмент пути к операции (например, https://www.example.com/.well-known/est/arbitraryLabel1 или https://www.example.com/.well-known/est/), для клиента EST может быть задан кортеж, состоящей из части полномочий (authority) в URI вместе с **необязательной** меткой (например, www.example.com:80 или arbitraryLabel1) или просто связанная с полномочиями часть URI.

3.2. Уровень HTTP

HTTP применяется для доставки сообщений EST. Идентификаторы URI определены для обработки каждого типа носителя (т. е. типа сообщения), как описано в параграфе 3.2.2. HTTP URI для управления. HTTP также применяется для служб аутентификации клиентов при недоступности аутентификации TLS по причине отсутствия сертификата, подходящего для TLS (3.2.3. Аутентификация клиента на основе HTTP). Аутентификация HTTP может также дополнять аутентификацию TLS, если сервер EST хочет получить дополнительные сведения, как указано в параграфе 2.2.3. Аутентификация клиента на основе HTTP. Запрошенные типы носителей служат для передачи сообщений EST, как показано на рисунке 6.

HTTP 1.1 [RFC2616] и выше поддерживает постоянные соединения. Как указано в параграфе 8.1 RFC 2616, постоянные соединения могут служить для снижения нагрузки на сеть и сократить обработку, связанную с множеством запросов HTTP. EST не требует и не исключает постоянных соединений HTTP.

3.2.1. Заголовки HTTP для управления

Значение HTTP Status служит для передачи сведений об успехе или отказе функции EST. Аутентификация HTTP применяется клиентом при запросе от сервера.

Тип носителя в заголовке HTTP Content-Type указывает, какое сообщение EST передаётся. Типы носителей, используемые EST, указаны в параграфе 3.2.4. Типы сообщений.

Перенаправления HTTP (коды 3xx) на тот же web-источник (см. [RFC6454]) клиенту **следует** обрабатывать без ввода от пользователя, если для исходного соединения применены все нужные проверки безопасности (3.3. Уровень TLS и 3.6. Полномочия сервера). Клиент инициирует новое соединение TLS и выполняет все применимые проверки безопасности при перенаправлении на другой web-источник. Перенаправления на другие web-источники требуют от клиента EST получить от пользователя ввод для запросов не-GET или HEAD, как указано в [RFC2616]. Если клиент уже имеет

сгенерированный CSR, включающий отождествление привязки и сведения POP (3.5. Связывание отождествления и сведений POP), CSR нужно создать заново для встраивания tls-unique из новой перенаправленной сессии. Отметим, что пару ключей заново создавать не требуется. Клиент получает нагрузку, связанную с интерфейсом и обработкой и администраторам серверов EST рекомендуется учитывать это.

В [RFC2616] сказано: «HTTP не использует поле Content-Transfer-Encoding (CTE) из RFC 2045», тем не менее, этот документ задаёт использование поля Content-Transfer-Encoding со значением base64 в параграфах 4.1.3, 4.3.1, 4.3.2, 4.4.2, 4.5.2 и приложениях A.1 - A.4. HTTP является бинарно чистым транспортом, поэтому не требуется указывать это для основанных на HTTP протоколах, таких как EST. Реализациям серверов EST **следует** опускать заголовок Content-Transfer-Encoding, если они заранее знают, что клиенты EST не полагаются на это поле. Клиентам EST следует предполагать, что заголовок Content-Transfer-Encoding будет отсутствовать, если он заранее не согласован с сервером EST. Механизм такого согласования выходит за рамки этого документа.[1]

3.2.2. HTTP URI для управления

Сервер EST **должен** поддерживать использование префикса пути /.well-known/, как задано в [RFC5785], и зарегистрированное имя est. Таким образом, действительный путь URI сервера EST начинается с https://www.example.com/.well-known/est. Каждую операцию EST задаёт суффикс пути (Рисунок 5).

Операция	*Путь*	*Описание*
Распространение сертификатов CA (**должно**)	/cacerts	4.1. Распространение сертификатов CA
Зачисление клиентов (**должно**)	/simpleenroll	4.2. Функции для запроса сертификатов клиентами
Повторное зачисление клиентов (**должно**)	/simplereenroll	4.2.2. Простое перезачисление клиентов
Full CMC (**необязательно**)	/fullcmc	4.3. Полные сообщения CMC
Генерация ключей на стороне сервера (**необязательно**)	/serverkeygen	4.4. Генерация ключей на стороне сервера
Атрибуты CSR (**необязательно**)	/csrattrs	4.5. Атрибуты CSR

Рисунок 5.Операции и соответствующие суффиксы URI.

В конце рабочего пути указывается префикс (Рисунок 5) для формирования URI, используемого с HTTP GET или POST для выполнения желаемой операции EST. Примером абсолютного пути URI для операции /cacerts является /.well-known/est/cacerts. Для извлечения сертификатов CA клиент EST будет использовать строку запроса HTTP

```
GET /.well-known/est/cacerts HTTP/1.1
```

Для запроса нового сертификата клиент EST будет использовать строку

```
POST /.well-known/est/simpleenroll HTTP/1.1
```

Использование разных рабочих путей упрощает реализацию серверов, которые не выполняют аутентификацию клиентов при распространении откликов /cacerts.

Сервер EST может предоставлять услуги множеству CA, на что указывает **необязательный** добавочный сегмент пути между зарегистрированным именем приложения и рабочим путём. Для предотвращения конфликтов **недопустимо** совпадение метки CA с любым заданным сегментом рабочего пути. Сервер EST **должен** предоставлять услуги независимо от наличия дополнительного сегмента пути. Ниже показаны 3 действительных URI.

1. https://www.example.com/.well-known/est/cacerts
2. https://www.example.com/.well-known/est/arbitraryLabel1/cacerts
3. https://www.example.com/.well-known/est/arbitraryLabel2/cacerts

В этой спецификации различие между зачислением и обновлением (сменой ключей) явно указывается HTTP URI. При запросе операций /fullcmc в CMC [RFC5272] применяются одинаковые сообщения для обновления сертификата и замены ключей.

Сервер EST может предоставлять дополнительные услуги с использованием других URI.

3.2.3. Аутентификация клиента на основе HTTP

Сервер EST **может** запросить проверку подлинности клиента на основе HTTP. Этот запрос может быть дополнением к успешной аутентификации TLS (3.3.2. Аутентификация клиента на основе TLS), если правила сервера EST требуют дополнительной проверки (например, сервер EST может потребовать от клиента EST «знать» пароль в дополнение к «наличию» клиентского сертификата). Аутентификация клиента на основе HTTP может быть заданным правилами резервным требованием сервера EST в ситуациях, где клиент EST не смог завершить аутентификацию TLS (это может возникать при первом зачислении клиента EST или невозможности применения доступных клиенту EST сертификатов для аутентификации TLS).

Аутентификация HTTP Basic и Digest **должна** выполняться только по протоколу TLS 1.1 [RFC4346] или более новой версии. Шифры NULL и anon применять **недопустимо**, поскольку они не обеспечивают конфиденциальности и не поддерживают взаимной аутентификации с сертификатами и без них, соответственно. Как указано в Certificate Management over CMS (CMC): Transport Protocols [RFC5273], серверу «**недопустимо** предполагать, что клиент поддерживает какой-либо тип аутентификации HTTP, такой как cookie, Basic или Digest». Клиентам **следует** поддерживать механизмы Basic и Digest.

Серверы, желающие применять аутентификацию Basic и Digest отвергают запрос HTTP, используя определённый в HTTP заголовок отклика WWW-Authenticate (параграф 14.47 в [RFC2616]). Предполагается, что клиент повторит запрос, используя подходящий заголовок Authorization Request (параграф 3.2.2 в [RFC2617]), если он способен применять аутентификацию Basic или Digest. Если клиент не может повторить запрос или не поддерживает аутентификацию Basic и Digest, он **должен** прервать соединение.

Клиент **может** указать пустую строку имени пользователя (""), если он представляет пароль, не связанный с именем.

[1] В оригинале этот абзац отсутствует. См. https://www.rfc-editor.org/errata/eid5107. *Прим. перев.*

Поддержка аутентификации клиента на основе HTTP влияет на безопасность, как отмечено в разделе 6. Вопросы безопасности. Клиенту **недопустимо** отвечать на запрос сервером аутентификации HTTP, пока он не предоставил полномочий серверу EST в соответствии с параграфом 3.6. Полномочия сервера.

3.2.4. Типы сообщений

Этот документ использует для сообщений имеющиеся типы носителей, заданные FTP и HTTP [RFC2585], application/pkcs10 [RFC5967], CMC [RFC5272]. Для согласованности с [RFC5273] каждый тип сообщений EST использует заголовок HTTP Content-Type с соответствующим типом носителя.

Тип сообщения (по операциям)	***Тип носителя для запроса Типы носителей для откликов Типы источников***	***Описание запроса Описание отклика***
Распространение сертификатов CA /cacerts	- application/pkcs7-mime [RFC5751]	4.1. Распространение сертификатов CA 4.1.1. Распространение сертификатов CA при начальной загрузке
Функции запроса сертификата клиента /simpleenroll /simplereenroll	application/pkcs10 application/pkcs7-mime [RFC5967] [RFC5751]	4.2. Функции для запроса сертификатов клиентами, 4.2.1. Простое зачисление клиентов 4.2.2. Простое перезачисление клиентов
Full CMC /fullcmc	application/pkcs7-mime application/pkcs7-mime [RFC5751]	4.3.1. Полный запрос CMC 4.3.2. Полный отклик CMC
Генерация ключа на стороне сервера /serverkeygen	application/pkcs10 multipart/mixed (application/pkcs7-mime и application/pkcs8) [RFC5967] [RFC5751] [RFC5958]	Ошибка: источник перекрёстной ссылки не найден 4.4.2. Отклик при генерации ключей на стороне сервера
Атрибуты CSR /csrattrs	- application/csrattrs Этот документ	4.5.1. Запрос атрибутов CSR 4.5.2. Отклик с атрибутами CSR

Рисунок 6. Сообщения EST и соответствующие типы носителей.

3.3. Уровень TLS

TLS обеспечивает аутентификацию, что позволяет принять решение о предоставлении полномочий (authorization). Сервер и клиент EST отвечают за согласование приемлемого набора шифров и взаимную проверку подлинности. Аутентификация TLS обычно выполняется с использованием сертификатов [RFC5280], но возможна и без применения таковых, когда ни клиент, ни сервер не представляют сертификат (3.3.3. Взаимная аутентификация TLS без сертификатов). Сервер EST **должен** аутентифицироваться в процессе согласования TLS, если только клиент не запросил распространение сертификатов CA при начальной загрузке (параграф 4.1.1) или Full CMC (параграф 4.3).

HTTPS [RFC2818] задаёт передачу сообщений HTTP через TLS. Протокол HTTPS **должен** применяться. Для всех взаимодействий EST **должен** использоваться протокол TLS 1.1 [RFC4346] (или более новой версии). **Следует** поддерживать возобновление сессий TLS [RFC5077].

Сведения о привязке канала TLS могут быть помещены в запрос сертификата, как указано в параграфе 3.5. Связывание отождествления и сведений POP, чтобы предоставить серверу EST гарантию того, что аутентифицированный клиент TLS имеет доступ к секретному ключу для запрошенного сертификата. Сервер EST **должен** реализовать 3.5. Связывание отождествления и сведений POP.

3.3.1. Аутентификация сервера на основе TLS

Должна поддерживаться аутентификация сервера TLS с применением сертификатов. Клиент EST проверяет подлинность сервера EST в соответствии с согласованным набором шифров. Ниже приведены сведения для случая шифров с сертификатами, таких как обязательный в TLS 1.1 [RFC4346] шифр TLS_RSA_WITH_3DES_EDE_CBC_SHA. Проверка сертификатов **должна** выполняться в соответствии с [RFC5280]. Сертификат сервера EST **должен** соответствовать профилю [RFC5280].

Клиент проверяет пригодность сертификата сервера TLS с использованием своей базы Explicit TA, а при включённой базе Implicit TA - также с её использованием. Клиент **должен** различать применение баз Explicit TA и Implicit TA в процессе проверки подлинности для поддержки надлежащего контроля полномочий. Клиент EST **должен** проверять полномочия в соответствии с параграфом 3.6. Полномочия сервера. Если проверка сертификата завершается отказом, клиент **может** следовать процедуре, описанной в параграфе 4.1.1. Распространение сертификатов CA при начальной загрузке.

3.3.2. Аутентификация клиента на основе TLS

Аутентификация клиента TLS является **рекомендуемым** методом отождествления клиентов EST. **Можно** применять аутентификацию на основе HTTP (3.2.3. Аутентификация клиента на основе HTTP). Сервер EST проверяет подлинность клиента EST в соответствии с согласованным набором шифров. Ниже приведены сведения для случая шифров с сертификатами, таких как обязательный в TLS 1.1 [RFC4346] шифр TLS_RSA_WITH_3DES_EDE_CBC_SHA. Сервер EST **должен** поддерживать аутентификацию клиентов на основе сертификатов.

Клиент обычно применяет имеющийся сертификат для операций обновления сертификата или смены ключей. Если сертификат, подлежащий обновлению или смене ключей, подходит для согласованного набора шифров, клиент **должен** использовать его для согласования TLS, в иных случаях клиенту **следует** применять другой сертификат, который подходит для набора шифров и содержит такое же отождествление субъекта. При запросе операции зачисления клиент **может** использовать для подтверждения своей подлинности сторонний сертификат.

Проверка сертификатов **должна** выполняться в соответствии с [RFC5280]. Сертификат клиента EST **должен** соответствовать профилю [RFC5280].

Сервер проверяет пригодность сертификата клиента TLS с использованием своей базы Explicit TA, а при включённой базе Implicit TA - также с её использованием. Сервер **должен** различать применение баз Explicit TA и Implicit TA в процессе проверки подлинности для поддержки надлежащего контроля полномочий. Сервер EST **должен** проверять полномочия в соответствии с параграфом 3.7. Полномочия клиента.

Если клиент не поддерживает аутентификацию TLS, он **должен** поддерживать аутентификацию на основе HTTP (3.2.3. Аутентификация клиента на основе HTTP) или аутентификацию TLS ьез сертификатов (3.3.3. Взаимная аутентификация TLS без сертификатов).

3.3.3. Взаимная аутентификация TLS без сертификатов

Шифры TLS без сертификатов обеспечивают способ взаимной проверки подлинности в случае отсутствия сертификатов у клиента и сервера, нежелания применять сертификаты или отсутствия привязок доверия, требуемых для проверки сертификатов. Клиент и сервер **могут** согласовать набор шифров без сертификатов для взаимной аутентификации.

При использовании взаимной аутентификации без сертификатов TLS для развёртывания, шифронабор **должен** быть основан на протоколе, устойчивом к атакам по словарю, и протокол **должен** иметь нулевое раскрытие (zero knowledge). Для этого подходят шифры TLS-SRP[1], т. е. шифры с _SRP_ в имени, указанные в параграфе 2.7 [RFC5054]. В разделе указаны характеристики шифров, подходящих для взаимной аутентификации без сертификатов при зачислении.

Успешная аутентификация с использованием шифров без сертификатов подтверждает знание распространённого секрета, который неявно подтверждает полномочия партнёров в обмене.

3.4. Подтверждение владения

Как указано в параграфе 2.1 CMC [RFC5272], доказательство владения (proof-of-possession или POP) «относится к значению, которое можно использовать для подтверждения того, что секретный ключ, соответствующий открытому ключу, находится во владении конечным объектом и может использоваться им».

Подписанный запрос на зачисление обеспечивает подтверждения владения на основе подписи. Описанный в параграфе 3.5 механизм усиливает это включением необязательного «прямого» доказательства с относящимися к сессии TLS сведениями, охватываемыми подписью запроса на зачисление (это связывает запрос на зачисление с аутентификацией конечной точки соединения TLS).

3.5. Связывание отождествления и сведений POP

Политика сервера будет определять, должны ли клиенты использовать описанный в этом параграфе механизм. Эта спецификация предоставляет метод связывания отождествления и подтверждения владения за счёт включения сведений о текущей аутентифицированной сессии TLS в подписанный запрос сертификации. Клиент может определить, требует ли сервер связывания отождествления и POP, проверив отклик с атрибутами CSR (4.5.2. Отклик с атрибутами CSR). Независимо от такого отклика, клиентам **следует** связывать отождествление и POP путём встраивания уникальных значения tls-unique в запрос сертификации. Если клиент включил такую информацию, сервер **должен** проверять её. Сервер EST **может** отвергать запросы без tls-unique в соответствии со своими правилами.

Связывание отождествления и POP подтверждает серверу, что аутентифицированный клиент TLS владеет секретным ключом, связанным с запросом сертификации и может подписать запрос сертификации после организации сессии TLS. Это служит альтернативой методу «связывания отождествления и сведений POP», заданному в разделе 6 [RFC5272] и доступному при использовании сообщений Full PKI.

Клиент, генерирующий CSR, получает значение tls-unique от подсистемы TLS, как описано в Channel Bindings for TLS [RFC5929]. Операции клиента EST между получением значения tls-unique путём генерации CSR с текущим tls-unique и последующей проверкой этого значения сервером EST являются «фазами прикладного протокола в процессе аутентификации на уровне приложения». Эти операции защищены механизмом функциональной совместимости, описанных в примечаниях по функциональной совместимости (параграф 3.1) Channel Bindings for TLS [RFC5929].

При повторном согласовании **должен** применяться механизм TLS secure_renegotiation [RFC5746].

Значение tls-unique кодируется в формате base64 в соответствии с разделом 4 [RFC4648] и результирующая строка помещается в поле challenge-password запроса сертификации ([RFC2985], параграф 5.4.1). Размер поля challenge-password ограничен 255 байтами (параграф 7.4.9 в [RFC5246] указывает, что ни в одном наборе шифров этой проблемы не возникает). Если атрибут challenge-password отсутствует, клиент не включает необязательных сведений channel-binding (наличие challenge-password указывает включение tls-unique).

Если сервер EST использует внутреннюю (back-end) инфраструктуру для обработки, **рекомендуется** сообщать результаты такой проверки. Например, для этого можно использовать CMC [RFC5272] RA POP Witness Control в сообщении CMC Full PKI Request или сервер EST может использовать аутентифицированного в TLS клиента EST как элемент доверенной архитектуры, который не пересылает недействительные запросы. Подробное обсуждение этого выходит за рамки документа.

При отклонении запроса отклик сервера EST аналогичен отклику, описанному для запросов на зачисление (4.2.3. Отклик на простое зачисление или перезачисление). если включается отклик Full PKI, **должно** устанавливаться CMCFailInfo = popFailed. При включении сообщения об отклонении для человека в нем **следует** приводить информативный текст, указывающий необходимость привязки отождествления к сведениям POP.

[1]Transport Layer Security-Secure Remote Password.

3.6. Полномочия сервера

Клиент **должен** проверять полномочия сервера EST до восприятия каких-либо откликов или ответа на запросы аутентификации HTTP. Метод проверки полномочий зависит от метода аутентификации сервера. При использовании для аутентификации базы данных Explicit TA применяется параграф 3.6.1, при использовании Implicit TA - параграф 3.6.2. Успешная аутентификация с использованием шифров без сертификата предполагает, проверку полномочий сервера. Клиент **может** выполнять начальную загрузку в соответствии с параграфом 4.1.1 в случае отказа при проверке.

3.6.1. Клиент, использующий базу Explicit TA

Когда клиент EST применяет базу данных Explicit TA для проверки сертификата сервера EST, он должен проверить указанный URI или последнее перенаправление HTTP для URI на предмет соответствия отождествления правилам, заданным в параграфе 6.4 [RFC6125] или сертификат сервера EST **должен** содержать расширение использования ключей id-kp-cmcRA [RFC6402].

3.6.2. Клиент, использующий базу Implicit TA

Когда клиент EST применяет базу данных Implicit TA для проверки сертификата сервера EST, он должен проверить указанный URI или последнее перенаправление HTTP для URI на предмет соответствия отождествления правилам, заданным в параграфе 6.4 [RFC6125]. Представленный URI или последнее перенаправление HTTP для URI обеспечивает основу для предоставления полномочий и аутентифицированное отождествление сервера подтверждает полномочия сервера.

3.7. Полномочия клиента

Решение о выдаче сертификата клиенту всегда контролируется локальной политикой CA, которая отражается в конфигурации сервера EST. Данный документ не задаёт ограничений для такой политики. EST предоставляет серверу EST доступ к аутентифицированному отождествлению каждого клиента, например, к клиентскому сертификату TLS в дополнение к любым свидетельствам аутентификации HTTP для поддержки реализации такой политики.

Если сертификат клиента был выпущен EST CA и включает расширение использования ключа id-kp-cmcRA [RFC6402], клиент является агентством регистрации (Registration Authority или RA), как описано в [RFC5272] и [RFC6402]. В этом случае серверу EST **следует** применять политику предоставления полномочий, совместимую с клиентом RA. Например, при обработке запросов /simpleenroll сервер EST может быть настроен на восприятие сведений о привязке POP, которые не относятся к текущей сессии TLS, поскольку аутентифицированный EST клиент RA проверил эти сведения, выступая сервером EST (3.5. Связывание отождествления и сведений POP). Доступны более конкретные механизмы RA при использовании клиентом EST методов /fullcmc.

4. Детали протокольного обмена

Перед обработкой запроса сервер EST проверяет полномочия клиента на получение запрошенных услуг. Точно так же клиент определяет, будет ли он отправлять запросы серверу EST. Эти решения о полномочиях описаны в двух следующих параграфах. Если предположить наличие полномочий у обеих сторон, фактические операции будут соответствовать описаниям двух следующих параграфов.

4.1. Распространение сертификатов CA

Клиент EST может запросить копии текущих сертификатов CA. Эта функция обычно выполняется до прочих функций EST.

4.1.1. Распространение сертификатов CA при начальной загрузке

Возможно, что на клиенте не настроена база данных Implicit TA, которая позволяет при начальной загрузке установить базу данных Explicit TA, как описано в параграфе 4.1.3. Отклик с сертификатами CA. В этом параграфе описан другой метод, с помощью которого минимально настроенный клиент EST может заполнить свою базу данных Explicit TA.

Если клиентское приложение EST не задало базы данных (ни Explicit TA, ни Implicit TA), начальная проверка подлинности и полномочий сервера TLS завершается отказом. Клиент **может** продолжить согласование TLS для доступа к методу /cacerts или /fullcmc. Если клиент EST продолжает использовать неаутентифицированное соединение, он **должен** извлечь данные содержимого HTTP из отклика (4.1.3. Отклик с сертификатами CA или 4.3.2. Полный отклик CMC) и привлечь пользователя (человека) к проверке полномочий сертификата CA с использованием внешних (out-of-band) данных, таких как отпечаток (fingerprint) сертификата CA (например, хэш SHA-256 или SHA-512 [SHS] всего сертификата CA). В отклике /fullcmc это будет элемент управления Publish Trust Anchors (параграф 6.15 в CMC [RFC5272]) внутри отклика Full PKI, который должен быть принят вручную. Пользователь должен подобающим образом проверить данные TA или предоставить при настройке данные отпечатка, требуемые для их проверки.

На запросы аутентификации HTTP **недопустимо** отвечать, если сервер не был аутентифицирован в соответствии с параграфом 3.3.1. Аутентификация сервера на основе TLS или не использована аутентификация без сертификатов, как указано в параграфе 3.3.3. Взаимная аутентификация TLS без сертификатов.

Клиент EST использует отклик /cacerts для организации базы данных Explicit TA с целью последующей аутентификации TLS для сервера EST. Клиентам EST **недопустимо** участвовать в другом протокольном обмене, пока не будет получен отклик /cacerts и организована новая сессия TLS (с аутентификацией по сертификатам TLS).

4.1.2. Запрос сертификатов CA

Клиенты EST запрашивают сведения из базы EST CA TA для CA (в форме сертификатов) через сообщение HTTPS GET с использованием пути к операции /cacerts. Клиенты и серверы EST **должны** поддерживать функцию /cacerts. Клиентам **следует** запрашивать актуальный (up-to-date) отклик до истечения срока действия сохранённых сведений, чтобы обеспечить актуальность базы данных EST CA TA. Серверу EST **не следует** требовать проверки подлинности и полномочий клиента для отклика на такие запросы.

Клиент **должен** аутентифицировать сервер EST, как указано в параграфе 3.3.1. Аутентификация сервера на основе TLS для аутентификации по сертификатам или 3.3.3. Взаимная аутентификация TLS без сертификатов при аутентификации без сертификатов и проверять полномочия сервера в соответствии с параграфом 3.6. Полномочия сервера или следовать процедуре, описанной в 4.1.1. Распространение сертификатов CA при начальной загрузке.

4.1.3. Отклик с сертификатами CA

В случае успеха отклик сервера **должен** иметь код HTTP 200. Другие коды говорят об ошибке и клиент **должен** прервать протокол.

Отклик об успехе **должен** быть certs-only CMC Simple PKI Response ([RFC5272]), содержащим сертификаты, описанные в следующем параграфе. Используется HTTP content-type application/pkcs7-mime. Отклик Simple PKI передаётся с Transfer-Encoding[1] base64 [RFC2045].

Сервер EST **должен** включить в отклик сертификат текущего корневого CA. Сервер EST **должен** включать любые дополнительные сертификаты клиента, которые будут требоваться для создания цепочки от выпущенного EST CA сертификата до текущей точки доверия EST CA TA. Например, если EST CA является подчиненным CA, в отклик включаются все сертификаты подчинённых CA, требуемые для создания цепочки к корневому EST CA.

Серверу EST **следует** включать три сертификата Root CA Key Update (OldWithOld, OldWithNew, NewWithOld) в цепочку отклика. Они определены в параграфе 4.4 CMP [RFC4210]. Клиент EST **должен** быть способен обработать эти сертификаты в отклике. Последний самоподписанный сертификат EST CA (например, NewWithNew) имеет наибольшее значение NotAfter. Если сервер EST не включает сертификаты в отклик, по завершении срока действия сертификата EST CA клиентам EST потребуется повторная инициализация в PKI с начальным распространением сертификатов CA (4.1.1. Распространение сертификатов CA при начальной загрузке), требующим участия пользователя.

После внешней (out-of-band) проверки все остальные сертификаты **должны** быть проверены с использованием обычной проверки пути [RFC5280] (свежий сертификат CA служит в качестве TA) до их использования при построении путей для проверки сертификатов.

Клиент EST **должен** сохранить извлечённый сертификат EST CA в базе данных Explicit TA для последующей аутентификации сервера EST. Клиенту EST **следует** отключить применение записей Implicit TA для этого сервера EST, когда доступна запись Explicit TA. Если клиент отключил базу Implicit TA, а сертификат сервера EST был проверен с использованием записи этой базы, клиент **должен** включать расширение Trusted CA Indication в будущие сессии TLS [RFC6066]. Это указывает серверу, что теперь приемлем лишь сертификат сервера EST, аутентифицируемый записью базы Explicit TA (иначе сервер EST может продолжить использование сертификата, проверяемого лишь с отключённой базой Implicit TA).

Клиенту EST **следует** также делать сведения сертификата CA доступными для программ конечного объекта с целью использования при проверке сертификатов партнёра.

4.2. Функции для запроса сертификатов клиентами

Клиенты EST запрашивают сертификат у сервера EST с помощью HTTPS POST, используя значение пути к операции /simpleenroll. Клиенты EST запрашивают обновление или смену ключей имеющихся сертификатов с помощью HTTP POST, используя путь /simplereenroll. Серверы EST **должны** поддерживать функции /simpleenroll и /simplereenroll.

Клиентам **рекомендуется** получать текущие сертификаты CA, как описано в параграфе 4.1. Распространение сертификатов CA, до вызова функций запроса сертификатов. Это обеспечивает клиенту возможность проверить сертификат сервера EST. Клиент **должен** проверить подлинность сервера EST в соответствии с параграфом 3.3.1, если применяется аутентификация по сертификатам, или в соответствии с параграфом 3.3.3 при аутентификации без сертификатов. Клиент **должен** проверить полномочия сервера в соответствии с параграфом 3.6.

Сервер **должен** проверять подлинность клиента в соответствии с параграфом 3.3.2 при использовании аутентификации по сертификатам или в соответствии с параграфом 3.3.3, если применяется необязательная аутентификация без сертификатов. Сервер **должен** проверять полномочия клиента в соответствии с параграфом 3.7. Сервер EST **должен** проверять значение tls-unique в соответствии с параграфом 3.5, если оно представлено клиентом.

Сервер **может** воспринять запрос сертификата для проверки администратором вручную (в параграфе 4.2.3 описано использование для таких случаев отклика HTTP 202, передаваемого клиенту EST).

4.2.1. Простое зачисление клиентов

При отправке HTTPS POST для /simpleenroll клиент **должен** включать Simple PKI Request, как указано в параграфе 3.1 [RFC5272] (т. е. PKCS #10 Certification Request [RFC2986]).

Подпись запроса сертификации (Certification Signing Request или CSR) обеспечивает подтверждение владения секретным ключом клиента серверу EST. Если расширение CSR KeyUsage указывает, что секретный ключ может применяться для создания цифровых подписей, клиент **должен** создать подпись CSR с использованием этого ключа. Если ключ может служить для создания цифровых подписей, но запрошенное расширение CSR KeyUsage запрещает генерацию цифровых подписей, подпись CSR все равно **можно** создать с использованием секретного ключа, но ключ **недопустимо** применять для иных операций с подписью (в соответствии с рекомендациями по подтверждению владения для RA или CA, как описано в [SP-800-57-Part-1]). Использование операций /fullcmc предоставляет доступ к расширенным методам подтверждения владения, которые применяются в случае невозможности использования пары ключей для генерации цифровой подписи (см. 4.3. Полные сообщения CMC).

Здесь применяется тип носителя (content-type) HTTP application/pkcs10. Формат сообщения задан в [RFC5967] с Transfer-Encoding[1] base64 [RFC2045].

Если подлинность клиента EST проверена с ранее установленным сертификатом от стороннего CA (см. 2.2.1. Аутентификация TLS с сертификатом), клиент **может** включить в CSR атрибут ChangeSubjectName, как указано в [RFC6402], для запроса смены subjectName и SubjectAltName в новом сертификате.

[1] В оригинале ошибочно сказано Content-Transfer-Encoding. См. https://www.rfc-editor.org/errata/eid5904. *Прим. перев.*

Клиент EST **может** запрашивать дополнительные сертификаты даже при использовании имеющегося сертификата при аутентификации клиента TLS. Например, клиент может использовать имеющийся сертификат для аутентификации клиента TLS при запросе сертификата, который не может применяться для проверки подлинности клиента TLS.

4.2.2. Простое перезачисление клиентов

Клиенты EST обновляют сертификаты или их ключи с помощью HTTPS POST, используя путь к операции /simplereenroll.

В запросе сертификата применяется такой же формат, как в запросе simpleenroll с тем же типом носителя HTTP (content-type). Поле Subject в запросе и расширение SubjectAltName **должны** быть идентичны соответствующим полям в сертификате, который обновляется или меняет ключи. В CSR **может** быть включён атрибут ChangeSubjectName, определённый в [RFC6402], для запроса смены этих полей в новом сертификате.

Если Subject Public Key Info в запросе сертификации совпадает со значением в текущем сертификате клиента, сервер EST обновляет клиентский сертификат. Если сведения об открытом ключе в запросе сертификации отличаются от данных в текущем сертификате клиента, сервер EST обновляет ключи клиентского сертификата.

4.2.3. Отклик на простое зачисление или перезачисление

Если зачисление успешно, отклик сервера **должен** содержать код отклика HTTP 200 с типом носителя application/pkcs7-mime.

Откликом об успехе **должен** быть CMC Simple PKI Response, содержащий только выпущенные сертификаты, как определено в [RFC5272]. Применяется тип носителя HTTP application/pkcs7-mime с параметром smime-type certs-only, как задано в [RFC5273].

При возникновении проблемы сервер **должен** отвечать кодом ошибки 4xx или 5xx HTTP [RFC2616]. В данные отклика **может** включаться Simple PKI Response с типом носителя HTTP application/pkcs7-mime (4.3.2. Полный отклик CMC) для передачи отклика об ошибке. Если тип носителя не указан, данные отклика должны содержать понятный человеку текст сообщения об ошибке с объяснением причины отклонения запроса (например, указание неполноты атрибутов CSR). Сервер **может** использовать тип носителя text/plain [RFC2046] для предназначенных человеку сообщений об ошибках.[1]

Ответ сервера с кодом HTTP 202 [RFC2616] указывает, что запрос воспринят для обработк, но отклик ещё не доступен. Сервер **должен** включить заголовок Retry-After, как определено для кода HTTP 503 и **может** также включить информацию, понятную человеку. Клиент **должен** ждать по меньшей мере в течение retry-after, прежде чем повторять тот же запрос. Клиент повторяет изначальный запрос на зачисление по истечении времени retry-after. Клиенту **следует** записывать такие события в журнал или информировать о них конечного пользователя. Сервер отвечает за поддержку всех состояний, требуемых для распознавания и обработки операций повтора, поскольку клиент такие состояния не поддерживает и просто повторяет один запрос, пока не получит другой код отклика. Все остальные коды обрабатываются в соответствии с HTTP [RFC2616].

Если клиент закрывает соединения TLS, ожидая завершения Retry-After, он инициирует новое соединение TLS и выполняет все применимые проверки безопасности. Если клиент уже создал CSR с привязкой отождествления к данным POP (3.5. Связывание отождествления и сведений POP), потребуется создать CSR заново с включением tls-unique из новой, перенаправленной сессии. Отметим, что пару ключей заново создавать не требуется. Обработка и интерфейс нагружают клиента и администраторам серверов EST рекомендуется учитывать это.

Клиент EST **может** сделать отклик с сертификатом и связанным с ним секретным ключом доступным для использования программами конечного объекта в качестве сертификата конечного объекта.

4.3. Полные сообщения CMC

Клиент EST может запросить сертификат у сервера EST с помощью HTTPS POST с путём к операции /fullcmc. Поддержка функции /fullcmc для клиента и сервера является **необязательной**.

4.3.1. Полный запрос CMC

Если HTTP POST с /fullcmc не является действительным Full PKI Request, сервер **должен** отвергнуть сообщение. Используется тип носителя HTTP application/pkcs7-mime с параметром smime-type CMC-request, как указано в [RFC5273]. Телом сообщения является двоичное представление PKI Request с Transfer-Encoding[2] base64 [RFC2045].

4.3.2. Полный отклик CMC

При успешном зачислении отклик сервера **должен** включать код HTTP 200 с типом носителя application/pkcs7-mime, как указано в [RFC5273]. Данные отклика включают Simple PKI Response с параметром smime-type certs-only или Full PKI Response с smime-type CMC-response, как указано в параграфе 3.2.1 [RFC5751]. Телом сообщения является двоичное представление PKI Response с Transfer-Encoding2 base64 [RFC2045].

При отклонении запроса сервер **должен** указать код HTTP 4xx или HTTP 5xx. Для любого отклика CMC об ошибке в данные отклика **должен** быть включён отклик CMC с типом носителя application/pkcs7-mime.

Остальные коды обрабатываются в соответствии с параграфом 4.2.3 или HTTP [RFC2616]. Например, клиент интерпретирует код HTTP 404 или 501, чтобы указать, что эта служба не реализована.

4.4. Генерация ключей на стороне сервера

Клиент EST может запросить секретный ключ и связанный с ним сертификат у сервера EST, используя HTTPS POST с путём к операции /serverkeygen. Поддержка функции /serverkeygen **не обязательна**.

[1] В оригинале этот абзац отличается. См. https://www.rfc-editor.org/errata/eid5108. *Прим. перев.*
[2] В оригинале ошибочно сказано Content-Transfer-Encoding. См. https://www.rfc-editor.org/errata/eid5904. *Прим. перев.*

Клиент **должен** проверить подлинность сервера EST в соответствии с параграфом 3.3.1, если применяется аутентификация по сертификатам, или в соответствии с параграфом 3.3.3 при аутентификации без сертификатов. Клиент **должен** проверить полномочия сервера в соответствии с параграфом 3.6.

Сервер **должен** проверять подлинность клиента в соответствии с параграфом 3.3.2 при использовании аутентификации по сертификатам или в соответствии с параграфом 3.3.3, если применяется необязательная аутентификация без сертификатов. Сервер EST по своему усмотрению применяет проверку полномочий или логику, чтобы определить, следует ли предоставлять секретный ключ и сертификат.

Шифронаборы с алгоритмом защиты конфиденциальности NULL применять **недопустимо**, поскольку они будут раскрывать содержимое незащищённого секретного ключа.

Реализация сервера отвечает за подобающую генерацию случайных чисел и ключа [RFC4086], а архивирование созданных ключей определяет политика CA. Пара ключей и сертификат передаются через сессию TLS. Шифр, применяемый для возврата секретного ключа и сертификата **должен** обеспечивать конфиденциальность, соизмеримую с доставляемым клиенту секретным ключом.

Клиент EST **может** запрашивать дополнительные сертификаты даже при использовании для аутентификации клиента TLS имеющегося сертификата. Например, клиент может применять имеющийся сертификат для аутентификации клиента TLS при запросе сертификата, который не подходит для аутентификации клиента TLS.

4.4.1. Запрос генерации ключа на стороне сервера

Запрос сертификата делается через HTTPS POST с использованием того же формата, как для расширений пути /simpleenroll и /simplereenroll с таким же типом носителя и транспортным кодированием.

Во всех отношениях серверу **следует** обращаться с CSR как при зачислении или перезачислении CSR, единственным отличием является то, что сервер **должен** игнорировать открытый ключ и подпись в CSR. Они включаются в запрос лишь для возможности повторного использования имеющихся баз кода для создания и анализа таких запросов.

Если клиент хочет получить секретный ключ с шифрованием, внешним по отношению к транспорту TLS, применяемому EST, и дополняющим его, или правила требуют доставки ключа в такой форме, клиент **должен** включить в CSR атрибут, указывающий ключ шифрования. Поддерживается симметричное и асимметричное шифрование, как описано ниже. Клиент **должен** также включить в CSR атрибут SMIMECapabilities (параграф 2.5 в [RFC2633]) для указания алгоритмов шифрования ключей, которые клиент желает применять.

Клиента настоятельно **рекомендуется** запрашивать защиту возвращаемого секретного ключа с использованием CMS EnvelopedData в дополнение к предоставляемой TLS защите для предотвращения несанкционированного раскрытия.

4.4.1.1. Запрос симметричного шифрования секретного ключа

Для указания симметричного ключа шифрования созданного сервером секретного ключа клиент **должен** включить атрибут DecryptKeyIdentifier (параграф 2.2.5 в [RFC4108]), задающий идентификатор секретного ключа, который будет применяться для шифрования. Хотя этот атрибут изначально предназначался для указания секретного ключа из микрокода (firmware), он полностью соответствует требованиям к указанию секретного ключа для шифрования созданного секретного (private) ключа. Если у сервера нет секретного ключа, соответствующего указанному идентификатору, запрос **должен** прерываться с возвратом клиенту ошибки. Распространение ключа, указанного DecryptKeyIdentifier, генератору ключей выходит за рамки этого документа.

4.4.1.2. Запрос асимметричного шифрования секретного ключа

Для указания асимметричного ключа шифрования созданного сервером секретного ключа клиент должен включить атрибут AsymmetricDecryptKeyIdentifier в форме id-aa-asymmDecryptKeyID OBJECT IDENTIFIER ::= { id-aa 54 }. Значение атрибута asymmetric-decrypt-key-identifier имеет тип ASN.1 AsymmetricDecryptKeyIdentifier ::= OCTET STRING (представление ASN.1 задано в [X.680]). Если у сервера нет секретного ключа, соответствующего указанному идентификатору, запрос **должен** прерываться с возвратом клиенту ошибки. Распространение ключа, указанного AsymmetricDecryptKeyIdentifier, генератору ключей выходит за рамки этого документа. Если указанный ключ связан с сертификатом X.509, этот ключ **должен** явно поддерживать keyTransport или keyAgreement или его использование **должно** быть неограниченным.

4.4.2. Отклик при генерации ключей на стороне сервера

При успешном запросе отклик сервера должен иметь код HTTP 200 с типом носителя multipart/mixed из двух частей - секретный ключ и данные сертификата. Формат возвращаемых данных секретного ключа зависит от секретного ключа, применяемого для дополнительного шифрования (сверх TLS). Если дополнительное шифрование не применяется, данные секретного ключа **должны** помещаться в application/pkcs8 с кодированием base64 DER-представления [X.690] PrivateKeyInfo с Transfer-Encoding[1] base64 [RFC2045].

При использовании дополнительного шифрования секретный ключ помещается в CMS SignedData. Данные SignedData подписываются создавшей секретный ключ стороной, которая может быть сервером EST или EST CA. Дополнительная защита SignedData обеспечивается размещением в CMS EnvelopedData, как описано в разделе 4 [RFC5958]. Далее показано использование EncryptedData в зависимости от заданного клиентом типа ключа защиты.

- Если клиент указал ключ симметричного шифрования для защиты созданного сервером секретного ключа, содержимое EnvelopedData шифруется с использованием указанного в запросе ключа. Поле EnvelopedData RecipientInfo **должно** указывать метод управления ключами шифрования kekri. Для версии указывается значение 4, идентификатор ключа шифрования ключей (kekid) имеет значение DecryptKeyIdentifier из параграфа 4.4.1.1, в keyEncryptionAlgorithm указывается один из алгоритмов переноса ключа (key wrap), которые клиент включил в возможности SMIMECapabilities, сопровождающие запрос, encryptedKey содержит зашифрованный ключ.

[1] В оригинале ошибочно сказано Content-Transfer-Encoding. См. https://www.rfc-editor.org/errata/eid5904. *Прим. перев.*

- Если клиент указал подходящий для транспортных операций ключ асимметричного шифрования для защиты созданного сервером секретного ключа, содержимое EnvelopedData шифруется с использованием сгенерированного случайного симметричного ключа. Для ключа симметричного шифрования **следует** обеспечивать криптостойкость, эквивалентную стойкости указанного клиентом асимметричного ключа. Поле EnvelopedData RecipientInfo **должно** указывать метод управления ключами шифрования KeyTransRecipientInfo (ktri). В KeyTransRecipientInfo поле RecipientIdentifier (rid) содержит значение subjectKeyIdentifier, копируемое из атрибута, определённого в параграфе 4.4.1.2, или сервер определяет связанное значение issuerAndSerialNumber из атрибута, версия выводится из выбора rid [RFC5652], в keyEncryptionAlgorithm указывается один из алгоритмов переноса ключей (key wrap), указанных клиентом в возможностях SMIMECapabilities, сопровождающих запрос, encryptedKey содержит ключ шифрования.

- Если клиент указал подходящий для операций согласования ключей ключ асимметричного шифрования для защиты созданного сервером секретного ключа, содержимое EnvelopedData шифруется с использованием сгенерированного случайного симметричного ключа. Для ключа симметричного шифрования **следует** обеспечивать криптостойкость, эквивалентную стойкости указанного клиентом асимметричного ключа. Поле EnvelopedData RecipientInfo **должно** указывать метод управления ключами шифрования KeyAgreeRecipientInfo (kari). В KeyAgreeRecipientInfo тип, версия, источник и ключевой материал пользователя (ukm) такие же как в [RFC5652], в keyEncryptionAlgorithm указывается один из алгоритмов переноса ключа (key wrap), включённых клиентом в возможности SMIMECapabilities, сопровождающие запрос. Идентификатор ключа получателя копируется из атрибута, определённого в параграфе 4.4.1.2, в subjectKeyIdentifier или сервер определяет IssuerAndSerialNumber в соответствии со значением, представленным в атрибуте.

Во всех трёх вариантах дополнительного шифрования EnvelopedData возвращается в отклике как application/pkcs7-mime с параметром smime-type server-generated-key и Transfer-Encoding base64.

Данные сертификата передаются в application/pkcs7-mime и точно соответствуют отклику с сертификатом для /simpleenroll.

Когда запрос отвергается, сервер **должен** указать ошибку HTTP 4xx или HTTP 5xx. Если тип носителя не задан, данные отклика **должны** быть понятным человеку текстом сообщения об ошибке.[1]

4.5. Атрибуты CSR

Политика CA может разрешать включение предоставленных клиентом атрибутов в выдаваемые сертификаты и некоторые из таких атрибутов могут содержать сведения, недоступные CA. Кроме того, CA может хотеть сертифицировать некий тип открытых ключей, а клиент может не знать этого заранее. Поэтому клиентам **следует** запрашивать список ожидаемых алгоритмов, которые нужны или желательны для CA в запросе на зачисление или требуются локальными правилами.

Серверу EST **не следует** требовать проверки подлинности или полномочий клиента для ответа на этот запрос.

Атрибуты CSR в запросе не обязательно, но клиентам следует осознавать, что CA могут отвергать запросы на зачисление, не соответствующие политике CA.

4.5.1. Запрос атрибутов CSR

Клиент EST запрашивает у CA список желаемых атрибутов CSR, передавая серверу EST сообщение HTTPS GET с путём к операции /csrattrs.

4.5.2. Отклик с атрибутами CSR

Если заданные локально правила для аутентифицированного клиента EST указывают предоставление CSR Attributes Response, отклик сервера **должен** включать код HTTP 200. Код HTTP 204 или 404 говорит о недоступности CSR Attributes Response. Независимо от кода отклика сервер EST и CA **могут** отклонять любые последующие запросы на зачисление по любой причине, например, из-за неполноты атрибутов CSR в запросе.

Отклики на запросы атрибутов **должны** представляться с типом носителя application/csrattrs и Transfer-Encoding[2] base64 [RFC2045]. Синтаксис тела application/csrattrs показан ниже.

```
AttrOrOID ::= CHOICE {
      oid OBJECT IDENTIFIER,
      attribute Attribute{YouNeedToDefineOrReferenceAnObjectSet}
}
```
[3]

Сервер EST может включать OID или атрибуты [RFC2986], включения которых в запрос сертификации он требует от клиента. Клиент **должен** игнорировать любые неизвестные OID и атрибуты. Когда сервер кодирует атрибуты CSR в форме пустой последовательности (SEQUENCE), это означает отсутствие у сервера конкретных сведений, которые он желает видеть в запросе клиента (эта функциональность эквивалентна коду отклика HTTP 204 или 404).

Если CA требуется определённая криптосистема или использование определённой схемы подписи (например, сертификация открытого ключа на основе некой эллиптической кривой или подпись с использованием определённого хэш-алгоритма), он **должен** предоставить сведения об этом в CSR Attribute Response. Если серверу EST требуется привязка отождествления к сведениям POP (3.5. Связывание отождествления и сведений POP), он **должен** включить challengePassword OID в CSR Attributes Response.

Структуре CSR Attributes Response **следует** максимально отражать структуру CSR в запросе. Запросы на использование определённой схемы подписи (например, конкретной хэш-функции) представляются как OID для отражения в SignatureAlgorithm структуры CSR. Запросы на использование определённой криптосистемы (например, сертификация открытого ключа на основе некой эллиптической кривой) представляются как атрибут для отражения в AlgorithmIdentifier структуры SubjectPublicKeyInfo с типом, указывающим алгоритм, и значениями с конкретными параметрами этого алгоритма. Запросы информативных сведений от клиента выполняются с помощью атрибута,

[1] В оригинале этот абзац отличается. См. https://www.rfc-editor.org/errata/eid5108. *Прим. перев.*
[2] В оригинале ошибочно сказано Content-Transfer-Encoding. См. https://www.rfc-editor.org/errata/eid5904. *Прим. перев.*
[3] В оригинале этот фрагмент содержал ошибки. См. https://www.rfc-editor.org/errata/eid4384. *Прим. перев.*

который представляется как атрибуты CSR с типом, указывающим extensionRequest [RFC2985], и значениями, задающими конкретные атрибуты, которые желательно включить в расширения результирующего сертификата.

Последовательность - это собой DER[1]-представление [X.690] с кодированием base64 (раздел 4 в [RFC4648]). Полученный в результате текст формирует тело application/csrattr без заголовков. Например, если CA хочет от клиента запрос на сертификацию, содержащий challengePassword (указывает запрос привязки отождествления к сведениям POP, см. параграф 3.5), extensionRequest [RFC2307] с MAC[2]-адресом клиента, применение эллиптической кривой secp384r1 и подписи с хэш-функцией SHA384, это может иметь вид

```
OID:         challengePassword (1.2.840.113549.1.9.7)
Attribute:   type = extensionRequest (1.2.840.113549.1.9.14)
             value = macAddress (1.3.6.1.1.1.1.22)
Attribute:   type = id-ecPublicKey (1.2.840.10045.2.1)
             value = secp384r1 (1.3.132.0.34)
OID:         ecdsaWithSHA384 (1.2.840.10045.4.3.3)
```

Кодирование в ASN.1 SEQUENCE даёт

```
30 41 06 09 2a 86 48 86 f7 0d 01 09 07 30 12 06 07 2a 86 48 ce 3d
02 01 31 07 06 05 2b 81 04 00 22 30 16 06 09 2a 86 48 86 f7 0d 01
09 0e 31 09 06 07 2b 06 01 01 01 01 16 06 08 2a 86 48 ce 3d 04 03
03
```

Последующее кодирование base64 представляет ASN.1 SEQUENCE в форме

```
MEEGCSqGSIb3DQEJBzASBgcqhkjOPQIBMQcGBSuBBAAiMBYGCSqGSIb3DQEJDjEJ
BgcrBgEBAQEWBggqhkjOPQQDAw==
```

5. Взаимодействие с IANA

В параграфе 4.4.1.2 дано определение OID, зарегистрированного в arc, делегированном IANA рабочей группе PKIX. Агентство IANA обновило реестр общеизвестных URI в соответствии с шаблоном из [RFC5785].

```
URI suffix: est
Change controller: IETF
```

Агентство IANA обновило реестр Application Media Types в соответствии с шаблоном из [RFC6838]. Субтип носителя для атрибутов CSR в CSR Attributes Response указывается в форме application/csrattrs.

```
Type name: application
Subtype name: csrattrs
Required parameters: None
Optional parameters: None
Encoding considerations: binary;
Security Considerations:
  Clients request a list of attributes that servers wish to be in
  certification requests.  The request/response is normally done
  in a TLS-protected tunnel.
Interoperability considerations: None
Published specification: This memo.
Applications which use this media type: Enrollment over Secure
Transport (EST)
Additional information:
  Magic number(s): None
  File extension: .csrattrs
Person & email address to contact for further information:
  Dan Harkins <dharkins@arubanetworks.com>
Restrictions on usage: None
Author: Dan Harkins <dharkins@arubanetworks.com>
Intended usage: COMMON
Change controller: The IESG <iesg@ietf.org>
```

Тип носителя application/pkcs7-mime задаёт необязательный параметр smime-type [RFC5751] с набором конкретных значений. Этот документ добавляет server-generated-key как значение параметра для Server-side Key Generation Response.

6. Вопросы безопасности

Поддержка аутентификации Basic, как задано в HTTP [RFC2617], разрешает серверу доступ к паролю клиента в открытом виде (cleartext). Это обеспечивает поддержку унаследованных баз «имя-пароль», но раскрывает пароли серверу EST. Применение PIN или одноразовых паролей позволяет смягчить последствия такого раскрытия и клиентам EST **рекомендуется** применять такие свидетельства лишь один раз для получения сертификата клиента (который будет применяться при последующем взаимодействии с сервером EST).

При использовании клиентом базы Implicit TA для проверки сертификата (3. Устройство и уровни протокола) проверка полномочий выполняется в соответствии с параграфом 3.6.2. В такой ситуации клиент подтверждает, что сервер является ответчиком, сертифицированным третьей стороной, но невозможно проверить полномочия ответчика выступать в качестве RA для PKI, куда клиент пытается зачислиться. Клиентам, использующим базу Implicit TA **рекомендуется** применять только основанную на TLS аутентификацию клиента (для предотвращения раскрытия сведения при аутентификации клиента на основе HTTP). Таким клиентам **рекомендуется** включать в запросы привязку отождествления к сведениям POP (3.5. Связывание отождествления и сведений POP), чтобы такие запросы не пересылались реальному серверу EST злоумышленником MITM[3]. **Рекомендуется** тщательно контролировать базу данных Implicit TA, применяемую для аутентификации сервера EST, чтобы снизить шансы стороннего CA со слабой практикой сертификации стать доверенным. Отключение базы Implicit TA после успешного получения отклика с распространением сертификатов CA (4.1.3. Отклик с сертификатами CA) снижает уязвимость первого обмена TLS.

[1]Distinguished Encoding Rules - правила отличительного кодирования.
[2]Media Access Control - управление доступом к среде.
[3]Man in the middle - перехват и изменение данных с участием человека.

Ниже указаны свойства шифров TLS без сертификатов, обеспечивающих защиту и выполняющих взаимную аутентификацию при зачислении.

- При активной атаке возможна лишь утечка информации о корректности хотя бы одной догадки о секрете.
- Атакующий может получить преимущество лишь за счёт взаимодействия, а не расчётов.
- Имеются меры противодействия, такие как экспоненциальная отсрочка после определённого числа неудачных попыток для срыва повторяющихся активных атак.

Применение шифра без сертификата, не обладающего указанными свойствами, делает результаты зачисления недействительными и может приводить к выдаче сертификатов непроверенным или несанкционированным объектам.

При использовании шифра TLS без сертификатов общий секрет, служащий для проверки подлинности и полномочий, не может быть передан объекту, не являющемуся участником обмена (т. е. не клиенту и не серверу). Любое расширение числа знающих общий секрет аннулирует защиту, обеспечиваемую шифром без сертификатов. Раскрытие общего секрета, применяемого шифром без сертификатов, сторонним объектам позволяет подменить (impersonation) клиента, что может приводить к повреждению клиентской базы точек доверия.

Шифры TLS, имена которых включают _EXPORT_ или _DES_, применять **недопустимо**. Такие шифры не обеспечивают достаточного уровня защиты - 40-битовая криптография в 2013 г. уже не обеспечивала приемлемой защиты, а алгоритм DES сочтён устаревшим.

Как описано в параграфе 6.7 CMC [RFC5272]: «Для ключей, которые могут служить ключами подписи, подписывание запроса сертификата с помощью секретного ключа служит подтверждением владения (POP) парой ключей». Включение tls-unique в запрос сертификации связывает POP с подтверждением отождествления TLS, обеспечивая выполнение операции POP в активной сессии TLS. Для сервера это означает, что аутентифицированный клиент имеет доступ к секретному ключу. Если известно, что аутентифицированный клиент обладает определёнными возможностями, такими как аппаратная защита свидетельств (credential) аутентификации и хранения ключей, это усиливает допущение, но не служит доказательством.

Метод генерации ключей на стороне сервера позволяет доставлять ключи клиенту через соединение TLS без защиты на прикладном уровне. Распространение материала секретных ключей по своей природе связано с риском. При распространении секретных ключей используется согласованный для TLS шифр. Ключи не защищаются предпочтительным методом упаковки ключей, таким как AES Key Wrap [RFC3394], или как указано в [RFC5958], поскольку шифрование секретного ключа в дополнение к обеспечиваемому транспортом TLS является необязательным. Серверам EST **рекомендуется** не поддерживать эту операцию по умолчанию. Клиентам **рекомендуется** не запрашивать эту услугу, пока нет убедительных операционных преимуществ. Применение базы Implicit TA **не рекомендуется** при генерации ключей на стороне сервера. **Рекомендуется** использовать CMS Server-Side Key Generation Response.

В части атрибутов CSR, которые CA может перечислять для включения в запрос на зачисление, реальных проблем безопасности передаваемого содержимого не возникает, но способный вмешаться в диалог злоумышленник может исключить атрибуты, которые могут понадобиться серверу, включить нежелательные для сервера атрибуты или сделать бессмысленными другие атрибуты, нужные серверу.

Правила кодирования ASN.1 (например, DER и BER) имеют структуру TLV[1] и легко создать вредоносное содержимое с некорректным полем размера, что может приводить к переполнению буфера. Правила кодирования ASN.1 разрешают произвольную глубину вложенности, что позволяет создавать вредоносное содержимое для переполнения стека. Интерпретаторам структур ASN.1 следует учитывать указанные проблемы и принимать соответствующие меры для защиты от переполнения буфера и стека, а также вредоносного содержимого в целом.

7. Литература

7.1. Нормативные документы

[RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2585] Housley, R. and P. Hoffman, "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP", RFC 2585, May 1999.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[RFC2633] Ramsdell, B., "S/MIME Version 3 Message Specification", RFC 2633, June 1999.

[RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, November 2000.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.

[RFC4108] Housley, R., "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages", RFC 4108, August 2005.

[1] Type-length-value - тип, размер, значение.

[RFC4210] Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", RFC 4210, September 2005.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, January 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, June 2008.

[RFC5273] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC): Transport Protocols", RFC 5273, June 2008.

[RFC5274] Schaad, J. and M. Myers, "Certificate Management Messages over CMS (CMC): Compliance Requirements", RFC 5274, June 2008.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, September 2009.

[RFC5746] Rescorla, E., Ray, M., Dispensa, S., and N. Oskov, "Transport Layer Security (TLS) Renegotiation Indication Extension", RFC 5746, February 2010.

[RFC5751] Ramsdell, B. and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification", RFC 5751, January 2010.

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, April 2010.

[RFC5929] Altman, J., Williams, N., and L. Zhu, "Channel Bindings for TLS", RFC 5929, July 2010.

[RFC5958] Turner, S., "Asymmetric Key Packages", RFC 5958, August 2010.

[RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, March 2011.

[RFC6402] Schaad, J., "Certificate Management over CMS (CMC) Updates", RFC 6402, November 2011.

[RFC6454] Barth, A., "The Web Origin Concept", RFC 6454, December 2011.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, January 2013.

[X.680] ITU-T Recommendation X.680 (2008) | ISO/IEC 8824-1:2008, "Abstract Syntax Notation One (ASN.1): Specification of basic notation", November 2008, <http://www.itu.int/rec/T-REC-X.680-200811-I/en>.

[X.690] ITU-T Recommendation X.690 (2008) | ISO/IEC 8825-1:2008, "ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", November 2008, <http://www.itu.int/rec/T-REC-X.690-200811-I/en>.

7.2. Дополнительная литература

[IDevID] IEEE Standards Association, "IEEE 802.1AR Secure Device Identifier", December 2009, <http://standards.ieee.org/findstds/standard/802.1AR-2009.html>.

[RFC2307] Howard, L., "An Approach for Using LDAP as a Network Information Service", RFC 2307, March 1998.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC2985] Nystrom, M. and B. Kaliski, "PKCS #9: Selected Object Classes and Attribute Types Version 2.0", RFC 2985, November 2000.

[RFC3394] Schaad, J. and R. Housley, "Advanced Encryption Standard (AES) Key Wrap Algorithm", RFC 3394, September 2002.

[RFC5054] Taylor, D., Wu, T., Mavrogiannopoulos, N., and T. Perrin, "Using the Secure Remote Password (SRP) Protocol for TLS Authentication", RFC 5054, November 2007.

[RFC5967] Turner, S., "The application/pkcs10 Media Type", RFC 5967, August 2010.

[RFC6403] Zieglar, L., Turner, S., and M. Peck, "Suite B Profile of Certificate Management over CMS", RFC 6403, November 2011.

[SHS] National Institute of Standards and Technology, "Secure Hash Standard (SHS)", Federal Information Processing Standard Publication 180-4, March 2012, <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>.

[SP-800-57-Part-1] National Institute of Standards and Technology, "Recommendation for Key Management - Part 1: General (Revision 3)", July 2012, <http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57_part1_rev3_general.pdf>.

Приложение A. Примеры сообщений (ненормативные)

В этом приложении рассмотрены варианты применения с подробным описанием сообщений на каждом уровне TLS.

A.1. Получение сертификатов CA

Ниже представлен пример действительного обмена /cacerts. В процессе начального согласования TLS клиент может игнорировать необязательный «запрос сертификата», созданный сервером, и обрабатывать вместо этого запрос HTTP GET.

```
GET /.well-known/est/cacerts HTTP/1.1
User-Agent: curl/7.22.0 (i686-pc-linux-gnu) libcurl/7.22.0 OpenS
SL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3
Host: 192.0.2.1:8085
Accept: */*
```

В ответ сервер передаёт текущие сертификаты CA.

```
HTTP/1.1 200 OK
Status: 200 OK
Content-Type: application/pkcs7-mime
Transfer-Encoding[1]: base64
Content-Length: 4246

MIIMOQYJKoZIhvcNAQcCoIIMKjCCDCYCAQExADALBgkqhkiG9w0BBwGgggwMMIIC
+zCCAeOgAwIBAgIJAJpY3nUZO3qcMA0GCSqGSIb3DQEBBQUAMBsxGTAXBgNVBAMT
EGVzdEV4YW1wbGVDQSBPd08wHhcNMTMwNTA5MDM1MzMxWhcNMTQwNTA5MDM1MzMx
WjAbMRkwFwYDVQQDExBlc3RFeGFtcGxlQ0EgT3dPMIIBIjANBgkqhkiG9w0BAQEF
AAOCAQ8AMIIBCgKCAQEAwDqpiHopaICubpRqbpEN7LqTIqWELFIA9qDDheHIKuyO
HW/ZAP7Rl4S5ZU6gaLW/ksseBUxdmox3KNyvtyjehIofTu28eZWhgy6/LCEGWR3P
K+fgPBA0l0JfJR/8oeXZa70oLVQc3hI4kCeqjFMs+biYH0vp/RluhftyZ5kzQyH1
EGsRkw1/qUKkTZ8PCF8VFlYfqmUoqsaRTyZbjII4J+Y6/jEG+p7QreW9zcz4sPe8
3c/uhwMLOWQkZtKsQtgo5CpfYMjuAmk4Q2joQq2vcxlc+WNKHf+wbrDb11ORZril
9ISlI94oumcRz3uBG1Yg7z83hdDfasmdfbp8gOSNFQIDAQABo0IwQDAPBgNVHRMB
Af8EBTADAQH/MB0GA1UdDgQWBBQITTKxMqATXrfc4ffpCIbt6Gsz0jAOBgNVHQ8B
Af8EBAMCAQYwDQYJKoZIhvcNAQEFBQADggEBACPnQPu5WReUGuCMS0nBOGa2tXh6
uZP4mS3J1qEfDePam/IiU9ssyYdcDwhVvKMoP4gI/yu4XFqhdpIoy/PyD4T15MT7
KADCxXkh5rM1IqMui7FvBKLWYGdy9sjEf90wAkBjHBe/TMO1NNw3uELyONSkHIvo
X0pu6aPmm/moIMyGi46niFse1iWlXXldGLkOQsh0e7U+wpBX07QpOr2KB2+Yf+uA
KY1SWzEG23bUxXlvcbUMgANDGj5r6z+niKL0VlApip/iCuVEEOcZ91UlmJjVLQWA
x6ie+v84oM+pIojiGM0C4XWcVlKKEgcMOsN3S4lvm8Ptpq0GLoIJY8NTD20wggMD
MIIB66ADAgECAgEBMA0GCSqGSIb3DQEBBQUAMBsxGTAXBgNVBAMTEGVzdEV4YW1w
bGVDQSBPd08wHhcNMTMwNTA5MDM1MzMyWhcNMTQwNTA5MDM1MzMyWjAbMRkwFwYD
VQQDExBlc3RFeGFtcGxlQ0EgTndPMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAnn3rZ3rMJHwf7MD9K4mubxHAvtdnrsQf5OfgtMhRIL4aePNhAdgPyj8C
loxOgD3UTV+dQ1ViOzVxPN7acikoOnkIdRpjpOpkyMo+KkvHMQXGnQTbsMAv1qWt
9S12DMpo0GOA1e4Ge3ud5YPOTR/q6PvjN51IEwYKksG7CglwZwB+5JbwhYr2D/0u
btGltriRVixPWrvt+wz/ITp5rcjh/8RS3LE8tQy3kTNhJF3Y/esR2sSgOiPNgIto
CATysbaINEPr4MemqML4tDpR/aG9y+8Qe7s1LyMFvDletp2mmBykAC/7nOat/pwU
lB0sN524D1XAgz8ZKvWrkh+ZaOr3hwIDAQABo1IwUDAOBgNVHQ8BAf8EBAMCBLAw
HQYDVR0OBBYEFLHEaeZbowSn2Jejizu/uWqyMkI8MB8GA1UdIwQYMBaAFAhNMrEy
oBNet9zh9+kIhu3oazPSMA0GCSqGSIb3DQEBBQUAA4IBAQCLDkL7aLNV6hSOkIqH
q+shV9YLO56/tj00vY/jV5skgDHk5d0B+OGortKVuGa57+v0avTrlJns3bNW8Ntv
zkDEhmd00Ak02aPsi4wRHLFgttUf9HdEHAuTkAESPTU43DiptjkfHhtBMfsFrCkd
sxWzCz+prDOMHYfUEkhRVV++1zyGEX6ov1Ap2IU2p3E+ASihL/amxTEQAsbwjUTI
R52zoL6nMPzpbKeZi2M0eEBVF8sDueA9Hjo6woLjgJqV0/yc5vC2HAxUOhx0cWTY
GcRBgL/yOyQLKiY5TKBH951OjQ4vhF2HmcoO7DkcNLYJOge16ssx4ogBHul20VgF
XJJjMIIDAzCCAeugAwIBAgIBAjANBgkqhkiG9w0BAQUFADAbMRkwFwYDVQQDExBl
c3RFeGFtcGxlQ0EgTndOMB4XDTEzMDUwOTAzNTMzMloXDTE0MDUwOTAzNTMzMlow
GzEZMBcGA1UEAxMQZXN0RXhhbXBsZUNBIE93TjCCASIwDQYJKoZIhvcNAQEBBQAD
ggEPADCCAQoCggEBAMA6qYh6KWiArm6Uam6RDey6kyKlhCxSAPagw4XhyCrsjh1v
2QD+0ZeEuWVOoGi1v5LLHgVMXZqMdyjcr7co3oSKH07tvHmVoYMuvywhBlkdzyvn
4DwQNJdCXyUf/KHl2Wu9KC1UHN4SOJAnqoxTLPm4mB9L6f0ZboX7cmeZM0Mh9RBr
EZMNf6lCpE2fDwhfFRZWH6plKKrGkU8mW4yCOCfmOv4xBvqe0K3lvc3M+LD3vN3P
7ocDCzlkJGbSrELYKOQqX2DI7gJpOENo6EKtr3MZXPljSh3/sG6w29dTkWa4pfSE
pSPeKLpnEc97gRtWIO8/N4XQ32rJnX26fIDkjRUCAwEAAaNSMFAwDgYDVR0PAQH/
BAQDAgSwMB0GA1UdDgQWBBQITTKxMqATXrfc4ffpCIbt6Gsz0jAfBgNVHSMEGDAW
gBSxxGnmW6MEp9iXo4s7v7lqsjJCPDANBgkqhkiG9w0BAQUFAAOCAQEALhDaE6Mp
BINBsJozdbXlijrWxL1CSv8f4GwpUFk3CgZjibt/qW9UoaNR4E58yRopuEhjwFZK
2w8YtRqx8IZoFhcoLkpBDfgLLwhoztzbYvOVKQMidjBlkBEVNR5MWdrs7F/AxWuy
iZ2+8AnR8GwqEIbCD0A7xIghmWEMh/BVI9C7GLqd6PxKrTAjuDfEpfdWhU/uYKmK
cL3XDbSwr30j2EQyaTV/3W0Tn2UfuxdwDQ4ZJs9G+Mw50s7AG6CpISyOIFmX6/bU
DpJXGLiLwfJ9C/aum9nylYuGCJ68BuTrCs9567KGfXEXI0mdFFCL7TaVR43kjsg3
c43kZ7369MeEZzCCAvswggHjoAMCAQICCQDprp3DmjOyETANBgkqhkiG9w0BAQUF
ADAbMRkwFwYDVQQDExBlc3RFeGFtcGxlQ0EgTndOMB4XDTEzMDUwOTAzNTMzMloX
DTE0MDUwOTAzNTMzMlowGzEZMBcGA1UEAxMQZXN0RXhhbXBsZUNBIE53TjCCASIw
DQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAJ5962d6zCR8H+zA/SuJrm8RwL7X
Z67EH+Tn4LTIUSC+GnjzYQHYD8o/ApaMToA91E1fnUNVYjs1cTze2nIpKDp5CHUa
Y6TqZMjKPipLxzEFxp0E27DAL9alrfUtdgzKaNBjgNXuBnt7neWDzk0f6uj74zed
SBMGCpLBuwoJcGcAfuSW8IWK9g/9Lm7Rpba4kVYsT1q77fsM/yE6ea3I4f/EUtyx
PLUMt5EzYSRd2P3rEdrEoDojzYCLaAgE8rG2iDRD6+DHpqjC+LQ6Uf2hvcvvEHu7
NS8jBbw5XradppgcpAAv+5zmrf6cFJQdLDeduA9VwIM/GSr1q5IfmWjq94cCAwEA
AaNCMEAwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUscRp5lujBKfYl6OLO7+5
arIyQjwwDgYDVR0PAQH/BAQDAgEGMA0GCSqGSIb3DQEBBQUAA4IBAQBCz/CWdYvn
GM/SdCdEiom5A1VxaW8nKgCWg/EyWtAIiaHQuViB+jTUAE9lona2MbJoFHW8U5e8
9dCP0rJpA9UYXXhWvFQzd5ZWpms4wUYt1j3gqqd36KorJIAuPigVng13yKytxM7c
```

[1]В оригинале ошибочно сказано Content-Transfer-Encoding. См. https://www.rfc-editor.org/errata/eid5926. *Прим. перев.*

```
VmxQnh0aux3aEnEyRGAhGalHp0RaKdgPRzUaGtipJTNBkSV5S4kD4yDCPHMNbBu+
OcluerwEpbz6GvE7CpXl2jrTBZSqBsFelq0iz4kk9++9CnwZwrVgdzklhRfJ1Z4j
NkLruwbQ+o4NvBZsXiKxNfn3K2o3SK8AuaEyDWkq18+5rjcfprRO8x4YTW+6mXPq
jM0MAGNDEW+1oQAxAA==
```

A.2. Атрибуты CSR

Ниже приведён пример действительного обмена /csrattrs. При таком обмене клиент EST аутентифицирует себя с применением имеющегося сертификата, выпущенного CA, которому сервер EST предоставляет услуги. Исходное согласование идентично представленному в примере исходного зачисления. Запрос HTTP GET имеет вид

```
GET /.well-known/est/csrattrs HTTP/1.1
User-Agent: curl/7.22.0 (i686-pc-linux-gnu) libcurl/7.22.0 OpenS
SL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3
Host: 192.0.2.1:8085
Accept: */*
```

В отклике сервер указывает предложенные атрибуты, которые подходят для аутентификации клиента. В этом примере сервер EST также включает два примера атрибутов, которые клиент будет игнорировать, если они ему не известны.

```
HTTP/1.1 200 OK
Status: 200 OK
Content-Type: application/csrattrs
Transfer-Encoding[1]: base64
Content-Length: 171

MHwGBysGAQEBARYwIgYDiDcBMRsTGVBhcnNlIFNFVCBhcyAyLjk5OS4xIGRhdGEG
CSqGSIb3DQEJBzAsBgOINwIxJQYDiDcDBgOINwQTGVBhcnNlIFNFVCBhcyAyLjk5
OS4yIGRhdGEGCSskAwMCCAEBCwYJYIZIAWUDBAIC
```

A.3. Зачисление и повторное зачисление

В следующем примере показан обмен /simpleenroll. Сообщения с данными для /simplereenroll аналогичны. В процессе обмена клиент EST использует распространённые по отдельному каналу (out-of-band) имя пользователя и пароль для аутентификации себя на сервере EST. Это обычно поведение HTTP WWW-Authenticate, включённое здесь для информации. При использовании имеющегося сертификата клиента TLS сервер может не запрашивать заголовок HTTP WWW-Authenticate, например, в операции /simplereenroll.

При начальном согласовании TLS клиент может игнорировать необязательный «запрос сертификата», созданный сервером, и обрабатывать вместо этого запрос HTTP POST. В отклике на исходную попытку HTTP POST сервер запрашивает у клиента WWW-Authenticate (это возможно даже при использовании клиентом сертификата, как указано в нормативном тексте выше).

```
HTTP/1.1 401 Unauthorized
Content-Length: 0
WWW-Authenticate: Digest qop="auth", realm="estrealm",
nonce="1368141352"
```

В последующий запрос HTTP POST включается имя пользователя и пароль вместе с полным содержимым application/pkcs10.

```
POST /.well-known/est/simpleenroll HTTP/1.1
Authorization: Digest username="estuser", realm="estrealm", nonc
e="1368141352", uri="/.well-known/est/simpleenroll", cnonce="M
TYwMzg3", nc=00000001, qop="auth", response="144cc27f96046f1d70e
b16db20f75f22"
Host: 192.0.2.1:8085
Accept: */*
Content-Type: application/pkcs10
Transfer-Encoding1: base64
Content-Length: 882

MIIChTCCAW0CAQAwHzEdMBsGA1UEAxMUZGVtb3N0ZXA0IDEzNjgxNDEzNTIwggEi
MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQClNp+kdz+Nj8XpEp9kaumWxDZ3
eFYJpQKz9ddD5e5OzUeCm103ZIXQIxc0eVtMCatnRr3dnZRCAxGjwbqoB3eKt29/
XSQffVv+odbyw0WdkQOIbntCQry8YdcBZ+8LjI/N7M2krmjmoSLmLwU2V4aNKf0Y
MLR5Krmah3Ik31jmYCSvwTnv6mx6pr2pTJ82JavhTEIIt/fAYq1RYhkM1CXoBL+y
hEoDanN7TzC94skfS3VV+f53J9SkUxTYcy1Rw0k3VXfxWwy+cSKEPREl7I6k0YeK
tDEVAgBIEYM/L1S69RXTLujirwnqSRjOquzkAkD31BE961KZCxeYGrhxaR4PAgMB
AAGgITAfBgkqhkiG9w0BCQcxEhMQK3JyQ2lyLzcrRVl1NTBUNDANBgkqhkiG9w0B
AQUFAAOCAQEARBv0AJeXaHpl1MFIdzWqoi1dOCf6U+qaYWcBzpLADvJrPK1qx5pq
wXM830A1O+7RvrFv+nyd6VF2rl/MrNp+IsKuA9LYWIBjVe/LXoBO8dB/KxrYl16c
VUS+Yydi1m/a+DaftYSRGolMLtWeiqbc2SDBr2kHXW1TR130hIcpwmr29kC2Kzur
5thsuj276FGL1vPu0dRfGQfx4WWa9uAHBgz6tW37CepZsrUKe/0pfVhr2oHxApYh
cHGBQDQHVTFVjHccdUjAXicrtbsVhU5o1lPv7f4lEApv3SBQmJcaq5O832BzHw7n
PyMFcM15E9gtUVee5C62bVwuk/tbnGsbwQ==
```

Сервер EST применяет имя пользователя и пароль для проверки подлинности и полномочий и отвечает с выданным сертификатом.

```
HTTP/1.1 200 OK
Status: 200 OK
Content-Type: application/pkcs7-mime; smime-type=certs-only
Transfer-Encoding1: base64
Content-Length: 1122

MIIDOAYJKoZIhvcNAQcCoIIDKTCCAyUCAQExADALBgkqhkiG9w0BBwGgggMLMIID
BzCCAe+gAwIBAgIBFTANBgkqhkiG9w0BAQUFADAbMRkwFwYDVQQDExBlc3RFeGFt
cGxlQ0EgTndOMB4XDTEzMDUwOTIzMTU1M1oXDTE0MDUwOTIzMTU1M1owHzEdMBsG
```

[1] В оригинале ошибочно сказано Content-Transfer-Encoding. См. https://www.rfc-editor.org/errata/eid5926. *Прим. перев.*

```
A1UEAxMUZGVtb3N0ZXA0IDEzNjgxNDEzNTIwggEiMA0GCSqGSIb3DQEBAQUAA4IB
DwAwggEKAoIBAQClNp+kdz+Nj8XpEp9kaumWxDZ3eFYJpQKz9ddD5e5OzUeCm103
ZIXQIxc0eVtMCatnRr3dnZRCAxGjwbqoB3eKt29/XSQffVv+odbyw0WdkQOIbntC
Qry8YdcBZ+8LjI/N7M2krmjmoSLmLwU2V4aNKf0YMLR5Krmah3Ik31jmYCSvwTnv
6mx6pr2pTJ82JavhTEIIt/fAYq1RYhkM1CXoBL+yhEoDanN7TzC94skfS3VV+f53
J9SkUxTYcy1Rw0k3VXfxWwy+cSKEPREl7I6k0YeKtDEVAgBIEYM/L1S69RXTLuji
rwnqSRjOquzkAkD31BE961KZCxeYGrhxaR4PAgMBAAGjUjBQMA4GA1UdDwEB/wQE
AwIEsDAdBgNVHQ4EFgQU/qDdB6ii6icQ8wGMXvy1jfE4xtUwHwYDVR0jBBgwFoAU
scRp5lujBKfYl6OLO7+5arIyQjwwDQYJKoZIhvcNAQEFBQADggEBACmxg1hvL6+7
a+lFTARoxainBx5gxdZ9omSb0L+qL+4PDvg/+KHzKsDnMCrcU6M4YP5n0EDKmGa6
4lY8fbET4tt7juJg6ixb95/760Th0vuctwkGr6+D6ETTfqyHnrbhX3lAhnB+0Ja7
o1gv4CWxh1I8aRaTXdpOHORvN0SMXdcrlCys2vrtOl+LjR2a3kajJO6eQ5leOdzF
QlZfOPhaLWen0e2BLNJI0vsC2Fa+2LMCnfC38XfGALa5A8e7fNHXWZBjXZLBCza3
rEs9Mlh2CjA/ocSC/WxmMvd+Eqnt/FpggRy+F8IZSRvBaRUCtGE1lgDmu6AFUxce
R4POrT2xz8ChADEA
```

A.4. Генерация ключа сервером

Следующий пример представляет действительный обмен /serverkeygen, в котором клиент EST аутентифицирует себя, используя имеющийся сертификат, выданный CA, которому сервер EST предоставляет услуги. Исходное согласование идентично приведённому в примере зачисления. HTTP POST имеет вид

```
POST /.well-known/est/serverkeygen HTTP/1.1
Host: 192.0.2.1:8085
Accept: */*
Expect: 100-continue
Content-Type: application/pkcs10
Transfer-Encoding[1]: base64
Content-Length: 963

MIICwTCCAakCAQAwWzE+MDwGA1UEAxM1c2VydmVyS2V5R2VuIHJlcSBieSBjbGll
bnQgaW4gZGVtbyBzdGVwIDEyIDEzNjgxNDE5NTUxGTAXBgNVBAUTEFBJRDpXaWRn
ZXQgU046MTAwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCvE1/6m4A/
3/L32Suyzbf28LM9y8CQfp0aepa7o20BSfluvm8HXR44mlV+wpieM8H5n3Ub3RIo
RUun/FllIzK9uV7UrkqJ3Yzmq2NOoTd4C+OPsV/RPTu873dhFrssDk3P4NIphlSS
sSIkt5rhz7wYbCqCFR5Aphe/30Jx7D+xBI5Rs8e6vRS8IpuImh71BHiLfhq9AFhz
4ZJsOUSVpUmqUogFsM7SOQ6XI4dl+djhpjT+YTJ6hQ2PXrqdch3KsTQ8c6aKs+e2
5QJxh7O8JHVlPHo4YIxXtAYSutcbbTN5TXWFCWSrWDJ+zuMmk2yU+dio1oW7YR7V
ftAvazJ3laQbAgMBAAGgITAfBgkqhkiG9w0BCQcxEhMQZEZzQVhtSm5qb2tCdER2
cjANBgkqhkiG9w0BAQUFAAOCAQEAR+I0EQB+hSjrLCAjnVH6bZdHUNGszIdwx1iu
L4n+0XK3SfEzeOMkC4T74yFGKj3redS1Ht9atYUPb0D1Qi9Jf9Co8eLblo1l19A6
GaS798ofxIF0Pl0Dr6/GqjheqJEIbcDTAJq+kvDihyQ4GQnhosygIZHvKppQlebA
gvp2RJSnMroPCe6RgTU9E2fmI9rin/9PyXeWFF1namp+lYbTGwjv1aE1ikhjCLlH
veHhCdgOExPw+fqhKhHjp+0ZKBlo2bC3pqRWvDTiZuwt9UpFFfGtuxvpTp44oS/j
M/965hWIw/5dshY/wQjIfYs07bbq2ERbpJiw9bAQY34gyoVmEQ==
```

Поскольку атрибут DecryptKeyIdentifier не включён в этот запрос, отклик не содержит дополнительного шифрования сверх TLS. Отклик сервера EST имеет вид

```
HTTP/1.1 200 OK
Status: 200 OK
Content-Type: multipart/mixed ; boundary=estServerExampleBoundary
Content-Length: 3219
```

Это преамбула, которая будет игнорироваться, хотя для estServer удобно включить пояснительное поле с контактными данными или сведениями о поддержке.

```
--estServerExampleBoundary
Content-Type: application/pkcs8
Content-Transfer-Encoding: base64

MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDPwKtwJ7TjMgA+
Poj64V909ryql0foP1hU4Yq5y8/bOP5ZTe6ArgVhUye099Ac+dfdwpyP/DESiujU
F/dS62Vck3UWNbnw+4O38FUp0enLbbjSTud48KpEW6+FzkeuAanPGZMA1wKyrYy9
rD5tQOOJU/CBVhUrITyYLZNYUe4agbpcR0wMtrRr2E58Mu8wQ80ryk6nkL7COk5Z
IQdNRxldk7DFvpA85Yn1stumoGRtVLW51iXeTS1LtXwhuUb/j6Lds3vvAiJ2SiZ0
Y3rxPlnJVyFmR8Mf2TBOjzuFqva/VLD2ayQjgaGEjq2ZWHXelQAOZ6N3lrChojEK
FGq93yOhAgMBAAECggEBALQ5az/nYjd5x9Y3f7NMUffwl+jRRfMHCMTRx/u4AEAo
KBYm0hFVZZtxfM+z7xlD8G0Th6gs2hFA6gwcIlUPmiX+UaOLxht0xWaLGgYmcNAm
BiCDjLBQ7xRQCWtlcK9WCA5+HBWtcEy6244rXxh+IyWd6NT6bXC165AEcX87y/e3
JFJ7XFNeDP656s2DmxSCci+iDte6SaEm7sJvYGu16qevJeMThcQcC9/rJjXkvpGL
IKK2px5idad4Pb6+QHpqj3d4oM8djO6wYUvrH8XQLqAaF8Hd5lFWVU57nSYY+H79
GaNDTfRTUL6AXr7kmMsKVFOJ0JjZExUCVMZtGiqhB6UCgYEA639OtdWLZCzyZFMe
p7VlRddoz0VAtrU2dxnEb4cWD8Gerg8uNvp8OG84gH+6MbPwz4ZYWKCgDFqyrAlw
SF02n9Sovh93eoJ51atSbfeYUkLtB8L/HVk5/CBGEsV9MUkdMF0+B43YlhyEDyKW
fX2+0UeHLFgRrfpSzP2cXduEieMCgYEA4db/SIrwN2+g1Gjo3oE09kd89VGjVRer
srbcqc7DcPXP6Lw42sx96h4jVWWqHVo3DfwFBdUb1LH2cnVXQjgDUHdNdpl01cf/
BFYCFINi2qKMqiJYswkhYxZ1BLz/zuQTDbPFL2PgLniKFG2aFLrTS3S/tgeB5QwI
RPigH3kfI6sCgYAPqsCJyFMlrvfRRNZdQewi4VnPsEPF4/hjpAs1gD8vfSoZWlkw
vylUd9HCerzgYaA7rixieQ0sxTvtxhL6PXlM2NEBFQbV16hPFL6/IiG4F0u9oHNo
eG8rHtqKlSjnBn4yoYFm70Dhe7QtbZelcaAoPCH6CUHj2St5B8ZHWDtREQKBgHNp
wER+XIy4C2UByCANv9csaXulIOdXlXNbaCGPfOm5dWrm5ddLMf33MO9vaSRe+ku3
Q4nbgsGLwPp1ZQZ+QZNZpMi7W6306yp4GdAJ5Pb+oww/ST0VqW5OB7dILyK4A9S4
zkiNrf+Rsl8GM/vsDhc9rsuDwqofIAq/VHVBHNzJAoGBAOHQof5L6iGHOHcxLazx
4MGvRTpmzU/PX6Q3QxqpetEGFEDZAaL58L67SSS3fFBnKrVAidF6llC1bAH1aoRa
fYHUDi45xBoroy0hBwrnTKRxppua4UK75FUH5PPJfR6cCvw5stRkzIevTZHhozkX
pM7PYH/x4BiBmgQ3bhOqTp4H
```

[1]В оригинале ошибочно сказано Content-Transfer-Encoding. См. https://www.rfc-editor.org/errata/eid5926. *Прим. перев.*

```
--estServerExampleBoundary
Content-Type: application/pkcs7-mime; smime-type=certs-only
Content-Transfer-Encoding: base64

MIIDRQYJKoZIhvcNAQcCoIIDNjCCAzICAQExADALBgkqhkiG9w0BBwGgggMYMIID
FDCCAfygAwIBAgIBFjANBgkqhkiG9w0BAQUFADAbMRkwFwYDVQQDExBlc3RFeGFt
cGxlQ0EgTndOMB4XDTEzMDUwOTIzMjU1NloXDTE0MDUwOTIzMjU1NlowLDEqMCgG
A1UEAxMhc2VydmVyc2lkZSBrZXkgZ2VuZXJhdGVkIHJlc3BvbnNlMIIBIjANBgkq
hkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAz8CrcCe04zIAPj6I+uFfdPa8qpdH6D9Y
VOGKucvP2zj+WU3ugK4FYVMntPfQHPnX3cKcj/wxEoro1Bf3UutlXJN1FjW58PuD
t/BVKdHpy2240k7nePCqRFuvhc5HrgGpzxmTANcCsq2Mvaw+bUDjiVPwgVYVKyE8
mC2TWFHuGoG6XEdMDLa0a9hOfDLvMEPNK8pOp5C+wjpOWSEHTUcZXZOwxb6QPOWJ
9bLbpqBkbVS1udYl3k0tS7V8IblG/4+i3bN77wIidkomdGN68T5ZyVchZkfDH9kw
To87har2v1Sw9mskI4GhhI6tmVh13pUADmejd5awoaIxChRqvd8joQIDAQABo1Iw
UDAOBgNVHQ8BAf8EBAMCBLAwHQYDVR0OBBYEFKeZixu9F+appDX2SS5HaxmV6Jr4
MB8GA1UdIwQYMBaAFLHEaeZbowSn2Jejizu/uWqyMkI8MA0GCSqGSIb3DQEBBQUA
A4IBAQBHhLmRAKrnTapqqBObDM9IQDQPuwW+fW1gYwZKlSm/IWIwHEZL1igXhpjj
rf4xqpIkiJMmkaOeoXA8PFniX0/lZM9FQSM/j89CUf5dMoAqWj8s17xuXu9L/hVe
XjjXHsL40WuDG6tMPN9vcT8tE3ruor608MKSHFX/NEM6+AaNVSUPTmB33BgYB1Wa
E7pn3JMN6pjIxsHnF4pKi8qvoTSVVjaCEwUe8Q/fw1yvjoHoYJtyMn4v5Kb3Rt+m
s8Yie1tcfVQrjQutqr34/IJsKdPziZwi92KZa+1958A6M9O/p5OI0up9ZXKg2DEC
1O9qT0GyYJ6sxAyKiGTOxk6jMddDoQAxAA==
--estServerExampleBoundary--
```

Это эпилог, который также игнорируется.

Приложение B. Участники работы и благодарности

Редакторы документа благодарны Stephen Kent, Vinod Arjun, Jan Vilhuber, Sean Turner, Russ Housley и другим за их отклики и прототипы ранних версий документа. Спасибо авторам [RFC6403], на основе которого была создана эта спецификация.

Адреса авторов

Max Pritikin (editor)
Cisco Systems, Inc.
510 McCarthy Drive
Milpitas, CA 95035
USA
EMail: pritikin@cisco.com

Peter E. Yee (editor)
AKAYLA, Inc.
7150 Moorland Drive
Clarksville, MD 21029
USA
EMail: peter@akayla.com

Dan Harkins (editor)
Aruba Networks
1322 Crossman Avenue
Sunnyvale, CA 94089-1113
USA
EMail: dharkins@arubanetworks.com

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru