Internet Engineering Task Force (IETF)
Request for Comments: 8995
Category: Standards Track
ISSN: 2070-1721

M. Pritikin
Cisco
M. Richardson
Sandelman Software Works
T. Eckert
Futurewei USA
M. Behringer

K. Watsen
Watsen Networks

May 2021

Bootstrapping Remote Secure Key Infrastructure (BRSKI)

Инфраструктура ключей защищённой начальной загрузки

Аннотация

Этот документ определяет автоматизированную начальную загрузку автономной плоскости управления (Autonomic Control Plane или ACP), для чего запускается защищённая инфраструктура ключей (Secure Key Infrastructure). Это выполняется с использованием установленных изготовителем сертификатов X.509 в сочетании со службой предоставления полномочий (authorizing service) изготовителя через сеть (online) или автономно (offline). Это называется протоколом инфраструктуры защищённых удалённых ключей для начальной загрузки (Bootstrapping Remote Secure Key Infrastructure или BRSKI). Начальная загрузка нового устройства может происходить при использовании маршрутизируемого адреса и облачного сервиса, при наличии лишь локальной связности (link-local) или в изолированной (ограниченной) сети. Поддерживаются модели развёртывания с менее строгими требованиями безопасности. Начальная загрузка завершается, когда криптографическое отождествление (идентификатор) новой инфраструктуры ключей успешно на устройстве. Организованное защищённое соединение можно также использовать для развёртывания локально выпущенного сертификата на устройстве.

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 7841.

Информацию о текущем статусе документа, ошибках и способах обратной связи можно найти по ссылке https://www.rfc-editor.org/info/rfc8995.

Оглавление

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

1. Введение

Протокол BRSKI предоставляет решение для защищённой автоматической (zero-touch) начальной загрузки новых (ненастроенных) устройств, называемых заявителями (pledge). У заявителей имеется исходный идентификатор устройства (Initial Device Identifier или IDevID), установленный на заводе.

BRSKI произносится как brewski - это разговорное название пива в Канаде и некоторых частях Среднего запада США [brewski].

В этом документе представлены в первую очередь потребности провайдеров (ISP) и автономной плоскости управления (Autonomic Control Plane или ACP) ориентированной на предприятия интегрированной модели автономных сетей (Autonomic Networking Integrated Model and Approach или ANIMA) [RFC8994]. Этот процесс начальной загрузки соответствует требованиям обеспечения по умолчанию защиты всех операций из параграфа 3.3 в [RFC7575]. Другим пользователям протокола BRSKI потребуется предоставить свои заявления о применимости, включающие свои соображения безопасности и приватности для развёртывания. В разделе 9 подробно описана применимость такого использования ACP.

Протокол BRSKI требует значительного обмена данными между владельцем и изготовителем. В принятый по умолчанию режимах протокол обеспечивает криптографическую передачу управления исходному владельцу. В наиболее строгих режимах используются сведения из канала продажи, чтобы узнать владельца заранее. Возможна перепродажа устройств с разрешения изготовителя. Механизмы перехода прав собственности без разрешения производителя не включены в эту версию протокола, но могут быть разработаны в будущих версиях.

Этот документ описывает, как заявитель обнаруживает элемент своего сетевого домена (или обнаруживается им) для выполнения начальной загрузки. Этот элемент (устройство) называется регистратором (registrar). До начала каких-либо операций заявителю и регистратору необходимо установить взаимное доверие.

1. Регистратор проверяет подлинность заявителя - что это? что его отождествляет?
2. Регистратор проверяет полномочия заявителя - это моё? нужно ли мне это? каковы шансы, что это скомпрометировано?
3. Заявитель проверяет подлинность регистратора - что отождествляет регистратор?
4. Заявитель проверяет полномочия регистратора - нужно ли присоединяться к этой сети?

Этот документ детализирует протоколы и сообщения для ответов на поставленные выше вопросы. Используется соединение TLS и сертификат заявителя в форме PKIX (X.509v3, IEEE 802.1AR IDevID [IDevID]) для ответа на вопросы 1 и 2. Применяется новый артефакт «ваучер (voucher), который регистратор получает от уполномоченного изготовителем агентства подписей (Manufacturer Authorized Signing Authority или MASA) и передаёт заявителю для ответа на вопросы 3 и 4.

Посредник (proxy) обеспечивает очень ограниченную связность между заявителем и регистратором.

Детали синтаксиса ваучеров подробно описаны в [RFC8366]. Этот документ подробно рассматривает механизмы автоматизированного протокола получения ваучеров, включая определение сообщения voucher-request, задающего незначительное расширение формата ваучеров (3. Артефакт запроса ваучера), заданного в [RFC8366].

BRSKI делает хранящийся у заявителя корневой сертификат X.509 достаточным для проверки отождествления регистратора. В процессе организуется соединение TLS, которое можно напрямую использовать для регистрации через защищённый транспорт (Enrollment over Secure Transport или EST). Фактически BRSKI обеспечивает автоматизированный механизм для распределения сертификатовр при начальной загрузке (Bootstrap Distribution of CA Certificates), описанный в параграфе 4.1.1 [RFC7030], где заявитель «**должен** […] привлекать человека для проверки полномочий сертификата CA с использованием внешних (out-of-band) данных». С помощью BRSKI заявитель может автоматизировать процесс применения ваучеров. Интеграция с полным механизмом EST необязательна и тривиальна.

Инфраструктура BRSKI достаточно гибка, чтобы поддерживать другие инфраструктуры ключей начальной загрузки, такие как решение на основе симметричных ключей, но это выходит за рамки документа.

1.1. Прежние подходы к начальной загрузке

Буквально «подтянуть себя за шнурки» (pull yourself up by the bootstraps) нереально. Точно так же невозможно создать защищённую инфраструктуру ключей без привлечения внешних сил. Сегодня общепризнано, что исходные соединения между узлами не защищены, пока не завершено распространение ключей или не подготовлен зависящий от домена ключевой материал (зачастую это заранее распространённые ключи, включая SIM[1]-карты). Имеющиеся автоматизированные механизмы, известные как незащищённое доверие при первом использовании (Trust on First Use или TOFU) [RFC7435], «воскрешение утёнка» (resurrecting duckling) [Stajano99theresurrecting] или предварительная подготовка (pre-staging).

Другой подход состоял в попытке минимизировать действия пользователя в процессе начальной загрузки, не исключая их полностью. Исходный протокол EST [RFC7030] снижает число пользовательских действий при начальной загрузке, но не предлагает решений, позволяющих сделать автономными (без участия пользователя) ряд операций.

- Использование базы неявных привязок доверияе (Trust Anchor или TA) [RFC7030] для аутентификации связанной с владельцем службы (неавтономное решение, поскольку требует защищённого распространение URL).
- Привлечение пользователя (человека) для проверки полномочий сертификата CA с использованием внешних (out-of-band) данных (неавтономное решение, поскольку требует участия человека),
- Использование настроенной базы явных TA (неавтономное решение, поскольку распространение базы явных TA не является автономным).
- Использование метода взаимной аутентификации TLS без сертификатов (неавтономное решение, поскольку распространение симметричных ключей не является автономным).

Эти методы не соответствуют требованиям автоматизации (zero-touch).

Существуют технологии «звонка домой» (call home), когда заявитель сначала организует соединение с общеизвестной службой производителя, используя базовую модель аутентификации «клиент-сервер». После взаимной проверки подлинности заявителю передаются соответствующие свидетельства для аутентификации целевого домена. С этим связано несколько проблем и ограничений:

- заявителю требуется соединение со службой производителя в реальном масштабе времени;
- службе производителя раскрывается отождествление домена (проблема приватности);
- производитель отвечает за принятие решений о предоставлении полномочий (проблема ответственности).

BRSKI решает эти проблемы, задавая расширения протокола EST для автоматического распространения ваучеров.

1.2. 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 BCP 14 [RFC2119] [RFC8174] тогда и только тогда, когда они выделены шрифтом, как показано здесь.

Ниже приведены определения используемых в документе терминов.

ANI
: Автономная сетевая инфраструктура (Autonomic Networking Infrastructure), определённая в [RFC8993]. В разделе 9 указаны конкретные требования к зачвителям, посредникам и регистраторам, являющимся частью ANI.

Circuit Proxy
: Реализация Join Proxy с поддержкой состояний. Это предполагаемый тип посредников.

drop-ship
: Физическое распределение оборудования с принятыми по умолчанию заводскими настройками (factory default) в конечный пункт назначения. В автоматизированных (zero-touch) сценариях при прямой поставке не применяется предварительная настройка или подготовка.

Domain - домен
: Набор элементов с общей локальной привязкой доверия. Это включает посредника (proxy), регистратора, удостоверяющий центр (CA) домена, компоненты управления и все остальные элементы, уже входящие в домен.

Domain CA - удостоверяющий центр домена
: Удостоверяющий центр (Certification Authority или CA) домена обеспечивает функции сертификации для домена. По меньшей мере он предоставляет функциональность сертификации для регистраторов и поддерживает секретные ключи, которые задаёт домен. Кроме того, он может сертифицировать все элементы.

[1] Subscriber Identification Module - модуль идентификации подписчика (абонента).

domainID

IDentity с уникальным значением, основанным на сертификате CA для регистратора. Расчёт значений идентификаторов описан в параграфе 5.8.2. Расчёт domainID.

Enrollment - зачисление

Процесс, в котором устройство представляет в сеть ключевой материал и получает связанное с сетью отождествление. Например, отправка запроса сертификата агентству CA и получение сертификата в ответ.

IDevID

Сертификат X.509 исходного идентификатора устройства (Initial Device Identifier), устанавливаемый производителем в новом оборудовании. Термин заимствован из 802.1AR [IDevID].

imprint - принятие отпечатка (впечатывание)

Процесс, где устройство получает криптографический ключевой материал для отождествления и доверия к будущим взаимодействиям с сетью. Этот термин взят из работы Конрада Лоренца (Konrad Lorenz) по биологии утят: «в критический период утёнок будет предполагать, что все, похожее на утку-мать, фактически является его матерью». Эквивалентом для устройства является получение отпечатка (fingerprint) сертификата корневого удостоверяющего центра сети. Устройство, принявшее отпечаток от злоумышленника, постигнет судьба, похожая на судьбу утёнка, принявшего волка за мать. Защищённое впечатывание является основным вопросом этого документа [imprinting]. Аналогия с работой Лоренца впервые была отмечена в [Stajano99theresurrecting].

IPIP Proxy

Вариант прокси без учёта состояния.

Join Proxy - посредник присоединения

Элемент домена, помогающий заявителю присоединиться к домену. Join Proxy упрощает взаимодействие для устройств, обнаруживающих себя в среде, где они не имеют связности, пока не будут проверены в качестве членов домена. Для простоты в этом документе Join Proxy иногда называется просто посредником или прокси. Заявитель не знает, что он взаимодействует с посредником, а не напрямую с регистратором.

Join Registrar (and Coordinator) - регистратор (и координатор) присоединения

Представитель домена, настроенный (возможно, автоматически) для решения вопроса о присоединении к домену нового устройства. Администратор домена взаимодействует с регистратором (и координатором) присоединения для управления этим процессом. Обычно Join Registrar находится «внутри» домена. Для простоты в этом документе используется просто термин «регистратор». В [RFC8993] применяется термин «агент автономной службы регистратора присоединения» (Autonomic Service Agent или ASA). Применяется также сокращение JRC.

LDevID

Сертификат X.509 локального идентификатора устройства (Local Device Identifier), устанавливаемый владельцем оборудования. Термин заимствован из 802.1AR [IDevID].

Manufacturer - производитель

Производителем в этом документе называется организация, создавшая устройство. Обычно это исходный производитель оборудования (original equipment manufacturer или OEM), но в более сложных ситуациях может обозначать реселлера (value added retailer или VAR) и даже системного интегратора. Целью BRSKI в общем случае является исключение небольших различий между каналами продажи. Это делается для того, чтобы обеспечить возможность поставки всем заказчикам одинаковых устройств с унифицированной загрузкой микрокода (firmware). Это сокращает расходы производителя, а также номенклатуру продукции с «полевой» поддержкой и повышает шансы того, что микрокод будет актуальным (up to date).

MASA Audit-Log - журнал аудита MASA

Анонимизированный список прежних владельцев, поддерживаемый агентом MASA по устройствам (заявителям), как описано в параграфе 5.8.1. Отклик MASA Audit-Log.

MASA Service - служба MASA

Сторонняя служба MASA в общедоступной сети Internet. Агент MASA подписывает ваучеры, а также поддерживает репозиторий сведений журнала аудита событий начальной загрузки с защитой приватности. Служба не отслеживает владение.

nonced

Ваучер (или запрос), содержащий одноразовое значение nonce (обычный случай).

nonceless

Ваучер (или запрос), не содержащий nonce или основанный на точности часов для проверки срока действия.

offline

Когда архитектурный компонент не может в реальном масштабе времени взаимодействовать с партнёром в результате его отключения или проблем в сети, операцию называют автономной (offline).

Ownership Tracker - трекер владения

Служба слежения за владением в общедоступной сети Internet. Ownership Tracker использует бизнес-процессы для точного отслеживания владения всеми устройствами доменами, которые приобрели устройства. Эта служба не обязательна, но позволяет производителям предоставлять дополнительные услуги в случаях, когда их каналы продаж и распространения позволяют точно отслеживать владение. Данные отслеживания владения указываются в ваучерах, как описано в [RFC8366].

Pledge - заявитель

Предполагаемое (ненастроенное) устройства, имеющее заводское отождествление.

(Public) Key Infrastructure - инфраструктура (открытых) ключей

Набор систем и процессов, которые поддерживают работу системы открытых ключей. Регистратор выступает агентством регистрации (Registration Authority), см. [RFC5280] и раздел 7 в [RFC5272].

TOFU

Доверие при первом использовании (Trust on First Use), применяемое аналогично описанному в [RFC7435]. Заявитель не принимает рещений о защите, а просто доверяет первому регистратору, с которым он связался. Это также называется моделью «воскресающего утёнка» (resurrecting duckling).

Voucher - ваучер

Подписанный артефакт от агента MASA, указывающий криптографическое отождествление регистратора, которому заявителю следует доверять. Имеются разные типы ваучеров в зависимости от организации доверия. Разные типы ваучеров определены в [RFC8366].

1.3. Область действия

1.3.1. Среда поддержки

BRSKI может поддерживать большие платформы маршрутизации с мультимегабитными соединениями, установленные в центрах обработки данных (ЦОД) с контролируемым доступом. Кроме того, оно способно работать с тысячами устройств, размещённых в небезопасных средах, таких как предоставляемые провайдерами (ISP) оконечные устройства клиентов (Customer Premises Equipment или CPE), отгружаемые напрямую конечным пользователям. Явно поддерживаются ситуации, когда оборудование поставляется с распределенного склада напрямую в целевое место по запросу владельца домена. Такой запас устройств (SKU) может поддерживаться для множества возможных владельцев доменов, которые заранее не знают, какое из устройств будет поставлено в конкретное место.

Процесс начальной загрузки может занимать несколько минут в зависимости от сетевой инфраструктуры и скорости обработки в устройстве. Сетевые коммуникации само по себе не оптимизированы по скорости, а соображения приватности позволяют заявителю не анонсировать своё присутствие через широковещание.

Кочевым или мобильным устройствам часто требуется получать свидетельства для доступа в сеть из новых мест. Примером может служить мобильный телефон, перемещающийся между операторами или вышками сотовой связи. Обычно это называют «передачей обслуживания» (handoff). BRSKI не обеспечивает передачу обслуживания с малой задержкой, которая обычно требуется в таких ситуациях. В таких решениях можно применять BRSKI для организации взаимоотношений (LDevID) с владельцем «домашнего» домена. Полученные свидетельства затем используются для предоставления свидетельств, более подходящих для передачи обслуживания с малой задержкой.

1.3.2. Среды с ограничениями

Возникают вопросы о пригодности этого решения в общем случае для сетей «Интернета вещей» (Internet of Things или IoT). Это зависит от возможностей рассматриваемых устройств. Для описания границ лучше всего подходит терминология [RFC7228].

Описанное здесь решение нацелено, прежде всего, на устройства без ограничений (Class 2+ [RFC7228]), работающие в сетях без ограничений, и в целом не предназначено для устройств с ограничениями, работающих в сетях с ограничениями, таких как сети 802.15.4 со слабым питанием и потерями (Low-Power and Lossy Network или LLN).

В частности, здесь описаны аспекты протокола, которые могут приводить к коллапсу насыщения или истощению питания промежуточных маршрутизаторов с батарейным питанием в LLN. В сетях такого типа это решение не следует применять. Ограничения в основном связаны с большим размером свидетельств и ключей, требуемых для аутентификации устройств. Определение методов с симметричными ключами, удовлетворяющих эксплуатационным требованиям, выходит за рамки документа, но операции базового протокола (согласование и структуры подписей TLS) имеют достаточную гибкость алгоритмов для поддержки таких методов, когда они определены.

Протокол впечатывания (imprint), описанный здесь, может применяться устройствами без ограничений по энергопотреблению, присоединяемых к сети без ограничений (например, «умные» лампочки обычно подключаются к электросети и используют беспроводную технологию 802.11). Протокол может также применяться устройствами без ограничений в сетях, где нет ограничений по питанию, но имеются иные ограничения (например, сети 802.15.4). Содержимое сертификата и процесс, с помощью которого решаются 4 отмеченных выше вопроса, применимы к устройствам с ограничениями. Это просто протокол передачи отпечатков по кабелям (on-the-wire imprint), который может быть неприемлемым.

1.3.3. Управление доступом в сеть

В этом документе предполагается, что контроль доступа в сеть уже выполнен, не требуется или интегрирован посредником и регистратором так, что самому устройству не нужно заботиться о деталях. Хотя применение X.509 IDevID совместимо с IEEE 802.1AR [IDevID] и может быть согласовано с методами контроля доступа в сеть 802.1X, здесь это используется для проверки подлинности заявителя, а не для контроля доступа в сеть. Сочетание этого протокола с контролем доступа в сеть, возможно на основе расширяемого протокола аутентификации (Extensible Authentication Protocol или EAP) [RFC3748] выходит за рамки этого документа.

1.3.4. Bootstrapping - это не Booting

Этот документ описывает начальную загрузку (bootstrapping) как протокол для нахождения локальной привязки доверия. Предполагается, что эта привязка вместе с любыми данными конфигурации, установленными позднее, сохраняется на устройстве при его перезагрузке (booting). Начальная загрузка происходит нечасто, например, при смене владельца или сбросе к заводским настройкам.

1.4. Применение новой инфраструктуры ключей и следующие шаги

В результате применения описанного здесь протокола изначально загружаемые (bootstrap) устройства имеют общую привязку доверия (CA домена). Сертификаты конечных элементов (end-entity или EE) может выпускать CA домена. Это позволяет реализовать в домене защищённые функции, например,

- управление устройствами;
- аутентификация маршрутизации;
- обнаружение служб.

Основным ожидаемым преимуществом является возможность использовать свидетельства, предоставляемые протоколом для защиты автономной плоскости управления (Autonomic Control Plane или ACP) [RFC8994].

1.5. Требования к устройствам автономной сетевой инфраструктуры

Протокол BRSKI может применяться во множестве сред. Некоторые из описанных в документе опций выходят за рамки ANI. В этом параграфе рассмотрены базовые требования к устройствам ANI.

Для устройств, намеревающихся стать частью инфраструктуры ANI [RFC8993], включающей ACP [RFC8994], протокол BRSKI **должен** быть реализован. Заявитель должен выполнять обнаружение посредников, как описано в параграфе

4.1, с использованием анонсов DULL M_FLOOD (Discovery Unsolicited Link-Local) [RFC8990] базового протокола автономной сигнализации (GeneRic Autonomic Signaling Protocol или GRASP).

После успешной проверки артефакта ваучера **должна** возвращаться телеметрия состояния (5.7. Телеметрия статуса заявителя BRSKI). Заявитель ANIMA ANI **должен** реализовать расширения автоматизации EST, описанные в параграфе 5.9. Интеграция EST с начальной загрузкой, дополняющие EST [RFC7030] для лучшей поддержки автоматизированных устройств без конечного пользователя. Агент ANI Join Registrar ASA **должен** поддерживать все операции BRSKI и указанные выше операции EST. Всем устройствам ANI **следует** поддерживать функции посредника BRSKI с использованием прокси-устройств через ACP (см. 4.3. Обнаружение посредника и связь с регистратором).

2. Обзор архитектуры

В этом разделе описаны логические элементы схемы начальной загрузки, представленной упрощенно на рисунке 1.

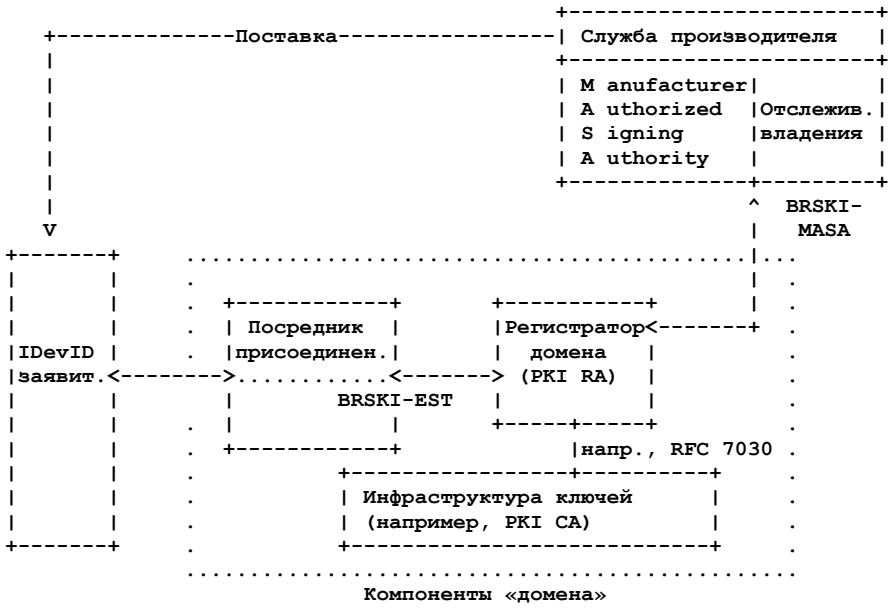

Рисунок 1. Обзор архитектуры.

Здесь предполагается сеть в оборудованием разных производителей. В такой среде могут присутствовать службы каждого производителя, поддерживающего эту спецификацию, или интегратор может обеспечивать общую службу, уполномоченную несколькими производителями. Возможность поддержки интегратором службы отслеживания владения для нескольких производителей маловероятна, поскольку для этого требуется интеграция каналов продаж разных производителей.

Домен - это управляемая сетевая инфраструктура с инфраструктурой ключей, к которой присоединяется заявитель. Домен обеспечивает начальное подключение устройства, достаточное для начальной загрузки через прокси. Регистратор домена аутентифицирует заявителя, принимает решения о предоставлении полномочий и распространяет ваучеры, полученные от службы производителя, а также может выступать как PKI CA.

2.1. Поведение заявителя

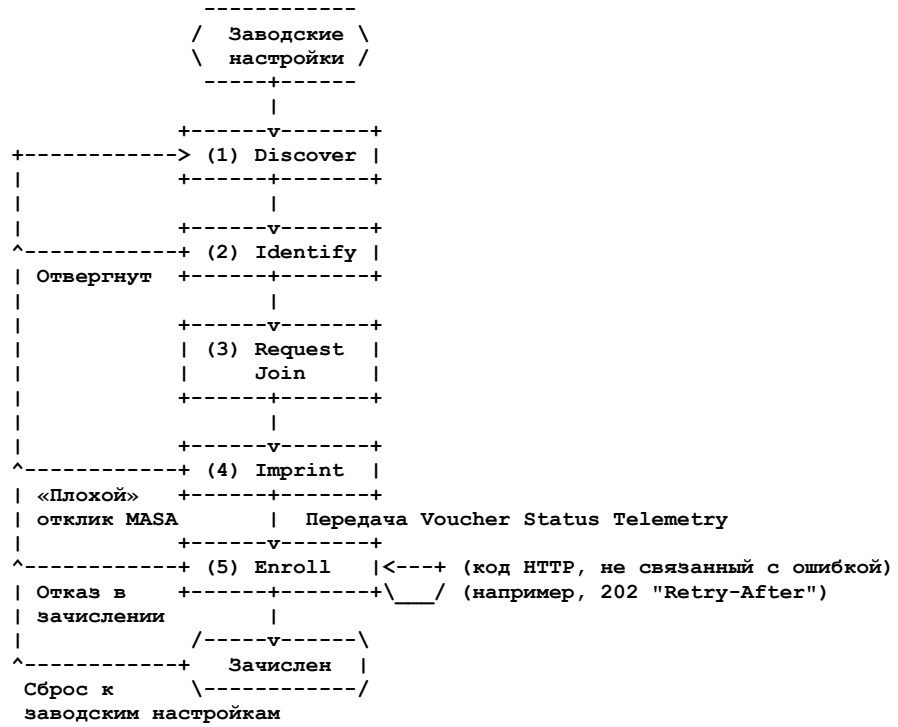

Рисунок 2. Состояния заявителя.

Заявитель выполняет шаги, описанные ниже на высоком уровне.

1. Обнаружение (Discover) канала связи с маршрутизатором.
2. Отождествление (Identify) самого себя, выполняемое представлением отождествления X.509 IDevID обнаруженному регистратору (через прокси) в согласовании TLS. Свидетельства регистратора в этот момент принимаются лишь временно.
3. Запрос на присоединение (Request join) к обнаруженному регистратору. Включается уникальное одноразовое значение (nonce), гарантирующее связывание любых откликов с конкретной попыткой начальной загрузки.
4. Впечатывание (Imprint) на регистраторе. Это требует проверки предоставленного службой производителя ваучера. Ваучер содержит информацию, достаточную для для полной аутентификации регистратора заявителем. Ниже это рассмотрено более подробно.
5. Зачисление (Enroll). После впечатывания между регистратором и заявителем будет аутентифицированное соединение TLS (HTTPS). Можно использовать EST [RFC7030] для получения сертификата домена от регистратора.

Заявитель после этого становится членом домена, может управляться им и будет повторять элементы обнаружения начальной загрузки лишь при возврате к заводским настройкам.

Эта спецификация подробно описывает интеграцию с зачислением EST, чтобы заявители могли получать локально выпущенные сертификаты, хотя в будущем возможна интеграция с любой службой представления состояний (Representational State Transfer или REST) [REST].

2.2. Защищённое впечатывание с использованием ваучеров

Ваучер - это криптографически защищённый артефакт (использующий цифровую подпись) для заявителя, позволяющий автоматическое (zero-touch) впечатывание его в домен регистратора. Формат и криптографические механизмы для ваучеров подробно описаны в [RFC8366].

Ваучеры обеспечивают гибкий механизм защиты впечатывания на основе проверки ваучера. На нижних уровнях защиты агент MASA может без разбора выдавать ваучеры и регистрировать (log) заявки на владение от доменов. На высших уровнях защиты выпус ваучеров может быть интегрирован с комплексными каналами продаж, но это выходит за рамки документа. Интеграция с каналом продаж будет подтверждать фактическое (правовое) владение заявителем со стороны домена. Это обеспечивает гибкость для множества вариантов применения на основе одного базового протокольного механизма для заявителя и регистратора, который будет развернут в «полевых» условиях. Службы MASA обладают гибкостью, позволяющей применять имеющиеся механизмы заявления прав и экспериментировать с разными уровнями защиты.

Ваучеры обеспечивают подписанный, но не шифрованный канал связи между заявителем, MASA и регистратором. Регистратор сохраняет контроль над решениями для транспорта и политики, позволяя применять локальные правила безопасности сети домена.

2.3. Исходный идентификатор устройства

Аутентификация заявителя и подписание запроса ваучера выполняются с использованием сертификата в форме PKIX, установленного в процессе производства. Это идентификатор 802.1AR IDevID, обеспечивающий основу для аутентификации заявителя в протокольном обмене, описываемом здесь. Иерархия PKI с общим корнем не требуется, каждый производитель устройств имеет свой корневой сертификат. IDevID позволяет, в частности, указанное ниже.

- Однозначная идентификация заявителя по отличительному имени (Distinguished Name или DN) и subjectAltName (SAN) в IDevID. Однозначная идентификация заявителя в объектах ваучеров выводится из этих параметров, как описано ниже. В параграфе 10.3 рассматривается влияние идентификатора на приватность.
- Обеспечение регистратору криптографической аутентификации заявителя (параграф 5.3).
- Защита автоматического обнаружения MASA заявителя регистратором (параграф 2.8).
- Подписание запроса ваучера с помощью IDevID заявителя (раздел 3).
- Обеспечение агенту MASA криптографической аутентификации заявителя (параграф 5.5.5).

В параграфе 7.2.13 (редакция 2009 г.) и 8.10.3 (редакция 2018 г.) [IDevID] обсуждаются расширения keyUsage и extendedKeyUsage для сертификата IDevID. В [IDevID] указано, что добавление ограничений в сертификат сужает его применимость для долгосрочных сертификатов. Эта спецификация подчёркивает данное обстоятельство и **рекомендует** не задавать ограничений на использование ключей. Это согласуется с параграфом 4.2.1.3 в [RFC5280], где не требуется ограничивать использование ключей для сертификатов конечных объектов.

2.3.1. Идентификация заявителя

В контексте BRSKI с заявителем однозначно связан серийный номер (serial-number). Этот номер используется в поле ваучера serial-number или voucher-request (3. Артефакт запроса ваучера) и локальных правилах регистратора или MASA (5. Детали протокола (заявитель - регистратор - MASA)).

Поле (сертификата) serialNumber определено в параграфе 4.1.2.2 [RFC5280]. В ASN.1 это называется CertificateSerialNumber. К данной спецификации это поле **не** имеет отношения. Не следует путать это поле с serial-number, определенным в этом документе или [IDevID] и параграфе 2.31 в [RFC4519].

Серийный номер определён в Приложении A.1 к [RFC5280] как X520SerialNumber с тегом OID id-at-serialNumber.

Серийный номер устройства (X520SerialNumber) используется заявителем для создания поля serial-number, помещаемого в voucher-request. Для этого значение преобразуется в тип type string.

Пример печатного представления поля serialNumber представлен в параграфе 2.31 (WI-3005) [RFC4519]. В этом параграфе дополнительно представлены атрибуты равенства и синтаксиса.

В силу имеющихся процессов предоставления отождествлений устройств некоторые изготовители хранят серийные номера в других полях. Регистраторам **следует** поддерживать настройку по изготовителям для поиска эквивалента серийного номера в других полях.

Как указано в параграфе 5.5, регистратор **должен** заново извлечь serialNumber из сертификата TLS заявителя. Он может обратиться к serial-number в запросе заявителя при возникновении возможности путаницы с номерами.

2.3.2. Расширение MASA URI

Этот документ определяет новое некритическое расширение сертификата PKIX для передачи MASA URI, предназначенное для применения в сертификате IDevID. Представление URI дано в параграфе 7.4 [RFC5280]. URI обеспечивает сведения о полномочиях. дерево BRSKI /.well-known [RFC8615] описано в разделе 5.

В этом расширении **может** быть полное значение URI, включая scheme, authority и path. Полное значение URI обычно применяется для диагностики или экспериментов. Как правило (и с учётом систем с ограничениями) идентификатор **следует** сокращать до authority в предположении схемы https:// (параграф 2.7.3 в [RFC7230]) и пути /.well-known/brski.

Регистратор может предположить наличие в расширении лишь элемента authority, если в расширении нет символов /.

В параграфе 7.4 [RFC5280] названы различные схемы, которые **должны** поддерживаться, включая облегченный протокол доступа к каталогам (Lightweight Directory Access Protocol или LDAP), HTTP, FTP. Однако регистратор **должен** использовать HTTPS для соединения BRSKI-MASA. Идентификация нового расширения показана ниже.

```
<CODE BEGINS>
MASAURLExtnModule-2016 { iso(1) identified-organization(3) dod(6)
internet(1) security(5) mechanisms(5) pkix(7)
id-mod(0) id-mod-MASAURLExtn2016(96) }

DEFINITIONS IMPLICIT TAGS ::= BEGIN

-- EXPORTS ALL --

IMPORTS
EXTENSION
FROM PKIX-CommonTypes-2009
  { iso(1) identified-organization(3) dod(6) internet(1)
    security(5) mechanisms(5) pkix(7) id-mod(0)
    id-mod-pkixCommon-02(57) }

id-pe FROM PKIX1Explicit-2009
  { iso(1) identified-organization(3) dod(6) internet(1)
     security(5) mechanisms(5) pkix(7) id-mod(0)
     id-mod-pkix1-explicit-02(51) } ;

MASACertExtensions EXTENSION ::= { ext-MASAURL, ... }
ext-MASAURL EXTENSION ::= { SYNTAX MASAURLSyntax
IDENTIFIED BY id-pe-masa-url }

id-pe-masa-url OBJECT IDENTIFIER ::= { id-pe 32 }

MASAURLSyntax ::= IA5String

END
<CODE ENDS>
```

Рисунок 3. Модуль MASAURL ASN.1.

Выбор id-pe основывается на рекомендациях параграфа 4.2.2 в [RFC5280]: «Эти расширения могут служить для направления приложений к online-сведениям об эмитенте субъекта.» MASA URL содержит именно online-сведения о конкретном субъекте.

2.4. Поток протокола

Представление потока показано на рисунке 4. При первой начальной загрузке новое устройство (заявитель) использует автоматическое обнаружение локальной службы (GeneRic Autonomic Signaling Protocol - GRASP или Multicast DNS - mDNS) для нахождения посредника в присоединении (Join Proxy). Посредник соединяет заявителя с локальным регистратором (JRC).

Найдя кандидата в регистраторы, новый заявитель передаёт регистратору некоторые сведения о себе, включая серийный номер, в форме voucher-request и свой сертификат IDevID как часть сессии TLS.

Регистратор может определить, ожидал ли он появления такого устройства, и найти MASA. Местоположение MASA обычно указывается в расширении IDevID. После определения приемлемости MASA вся информация из начального voucher-request (включая серийный номер устройства) передаётся через Internet по защищённому TLS каналу изготовителю устройства вместе по сведениями о регистраторе и владельце.

На основе представленных сведений и других источников информации (таких как записи о продажах) изготовитель может применить правила для решения вопроса о принятии заявки регистратора на владение устройством. Если заявка принимается, выдаётся ваучер, предписывающий устройству принять нового владельца.

Ваучер возвращается регистратору (а не устройству напрямую), чтобы регистратор мог проверить ваучер, журналы аудита MASA и иные источники для решения вопроса о подлинности устройства и возможности выполнения его начальной загрузки.

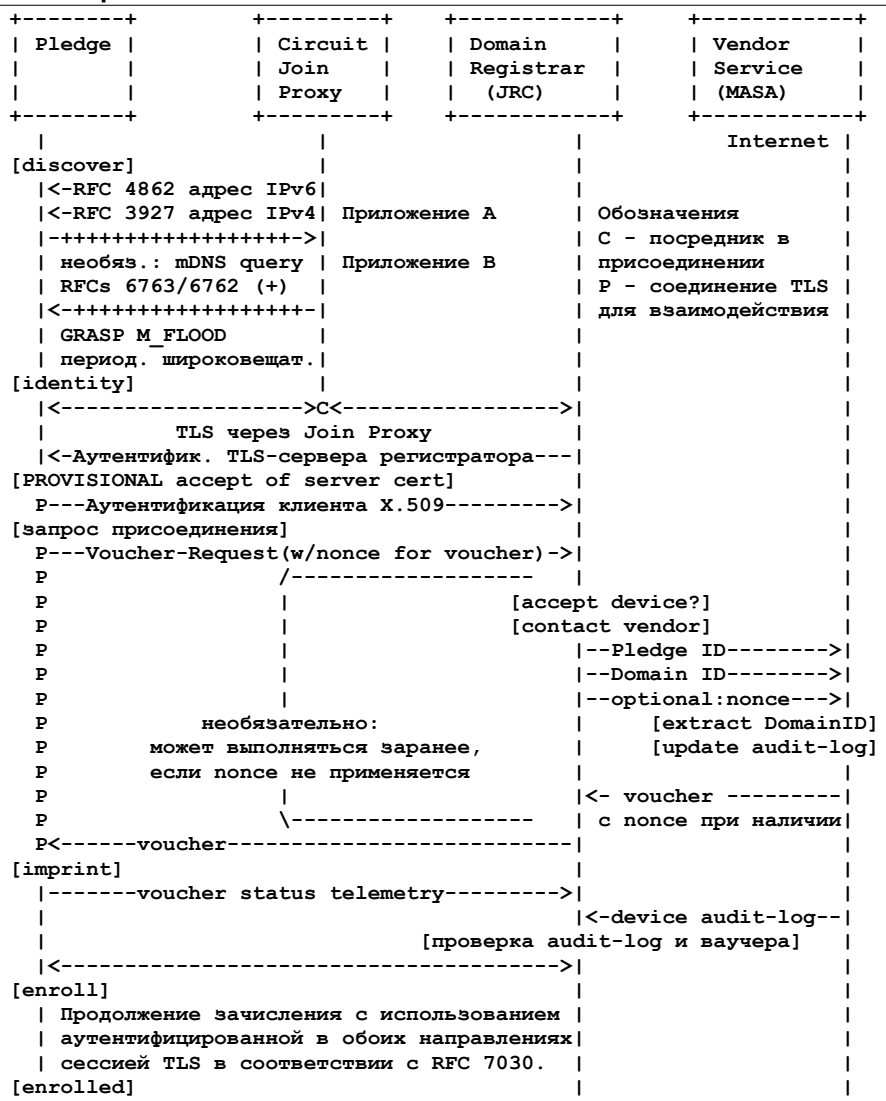

Рисунок 4. Временная диаграмма протокола.

В подписанном ваучере фильтрация сведений невозможна, поскольку это двоичное решение - «да-нет». Регистратор применяет к ваучеру локальную политику и в случае восприятия ваучера возвращает его заявителю для впечатывания.

Ваучер также включает привязку доверия, которую заявитель использует для представления владельца. Это используется для начальной загрузки из среды, где устройство имеет «встроенное» доверие лишь к изготовителю, в среду, где владелец имеет PKI-отпечаток на устройстве.

Когда BRSKI следует за EST, этот единственный отпечаток дополнительно используется в PKI владельца и LDevID для устройства. Последующие этапы отчётности предоставляют потоки сведений, указывающих результат процесса - успех или неудача.

2.5. Компоненты архитектуры

2.5.1. Заявитель

Заявитель является устройством, которое пытается присоединиться. Предполагается, что заявитель взаимодействует с Join Proxy используя локальное соединение (link-local). В большинстве случаев у заявителя нет других соединений, пока тот не завершит процесс зачисления и не получит то или иное свидетельство (credential) от сети.

2.5.2. Посредник присоединения

Join Proxy обеспечивает соединение HTTPS между заявителем и регистратором. механизм Circuit Proxy описан в разделе 4. Дополнительные механизмы, включая протокол приложений с ограничениями (Constrained Application Protocol или CoAP) и механизм IP in IP (IPIP) без поддержки состояний требуют дополнительной работы.

2.5.3. Регистратор домена

Регистратор домена выступает как клиент BRSKI-MASA при запросе ваучеров у MASA (5.4. Организация BRSKI-MASA TLS). Регистратор служит сервером BRSKI-EST, когда заявители запрашивают ваучеры (5.1. Организация BRSKI-EST TLS). Регистратор служит сервером BRSKI-EST агентства регистрации (Registration Authority) при запросе заявителем сертификата конечного объекта через соединение BRSKI-EST (5.9. Интеграция EST с начальной загрузкой).

Регистратор использует базу неявных привязок доверия (Implicit Trust Anchor) для аутентификации сертификата сервера MASA TLS соединения BRSKI-MASA. Настройка и распространение привязок доверия выходят за рамки этой спецификации.

Регистратор использует другую базу Implicit TA для аутентификации клиентского сертификата TLS заявителя в соединении BRSKI-EST. Настройка и распространение точек доверия клиента BRSKI-EST выходят за рамки этой

спецификации. Отметим, что привязки доверия в корне (/), исключённые из базы данных, будут влиять на выбор устройств изготовителя, воспринимаемых в качестве заявителей, а также будут служит для ограничения набора MASA, которые являются доверенными для зачисления.

2.5.4. Служба изготовителя
Служба изготовителя обеспечивает две логически разделённых функции - MASA, как описано в параграфах 5.5 и 5.6, а также отслеживание и аудит владения, как описано в параграфах 5.7 и 5.8.

2.5.5. Инфраструктура открытых ключей
Инфраструктура открытых ключей (Public Key Infrastructure или PKI) управляет сертификатами соответствующего домена, обеспечивая привязки доверия для него и позволяя зачислять заявителей с сертификатами домена.

Ваучеры обеспечивают метод распространения одной точки доверия PKI (как pinned-domain-cert). Распространение полного набора привязок доверия возможно через необязательную интеграцию с EST.

Регистратор домена выступает в качестве Registration Authority [RFC5272], запрашивая сертификаты заявителей в PKI.

Ожидания от PKI не изменились по сравнению с EST [RFC7030] и этот документ не вносит дополнительных архитектурных требований к PKI.

2.6. Проверка времени в сертификате

2.6.1. Отсутствие часов
При начальной загрузке многие устройства не знают текущего времени. Такие механизмы, как протоколы сетевого времени (Network Time Protocol) не могут быть защищены до завершения начальной загрузки. Поэтому определённая в этом документе модель начальной загрузки не требует знания текущего времени. Заявитель **может** игнорировать все временные метки в ваучере и периоды действия сертификата, если ему неизвестно текущее время.

Заявитель раскрывается со значениями даты в 5 элементах: notBefore и notAfter в сертификате регистратора, created-on и expires-on в ваучере, а также подписи в CMS[1], содержащие signingTime. Заявитель с часами, которым он доверяет, **должен** проверить указанные значения времени во всех обрабатываемых сертификатах и подписях.

Если ваучер содержит nonce, заявитель **должен** подтвердить соответствие исходному voucher-request от заявителя. Это подтверждает свежесть ваучера (5.2. Запрос заявителем ваучера у регистратора).

2.6.2. Неограниченный срок действия IDevID
В сертификатах с длительным сроком действия «**следует** устанавливать значение GeneralizedTime = 99991231235959Z» в поле notAfter, как описано в [RFC5280].

Некоторые развёрнутые системы управления IDevID не соответствуют требованиям 802.1AR к неограниченному сроку действия и имеют обычно срок действия сертификата не более 3 лет. Регистраторам **следует** быть настраиваемыми по производителям, чтобы они игнорировали сроки действия заявителей, когда те не соответствуют рекомендациям [RFC5280].

2.7. Облачный регистратор
Имеются операционно открытые сети, где устройства получают доступ в Internet без проверки полномочий (unauthenticated). В таких случаях домен управления для устройств должен быть открыт из сети Internet. Случаи, когда устройство может загрузиться и получить доступ Internet, менее вероятны в сфере действия ANIMA ACP, но могут стать более важными в будущем. В области действия ANIMA ACP новые устройства будут помещаться в карантин за посредником присоединения (Join Proxy).

Кроме того, имеются ситуации с совершенно новыми инсталляциями, где устройства могут иметь тот или иной тип восходящего канала (uplink) управления, пригодного для использования, например, сеть 3G. В таких будущих ситуациях устройство может использовать интерфейс управления, чтобы узнать, что оно должно настроить себя для работы в качестве локального регистратора. Для поддержки таких случаев заявитель **может** обратиться по общеизвестному URI облачного регистратора, если не удаётся обнаружить локальный регистратор или цель использования регистратора не включает локальный регистратор. Если заявитель использует общеизвестный идентификатор URI для обращения к облачному регистратору, **должна** использоваться заданная изготовителем база Implicit TA [RFC7030] для аутентификации этой службы в соответствии с [RFC6125]. Приемлемо использования DNS-ID для проверки и это может включать шаблонные компоненты в левой части. Это согласуется с настройкой URI сервера EST человеком [RFC7030], которая тоже зависит от [RFC6125].

2.8. Определение MASA для контактов
Регистратор должен иметь возможность контакта с агентом MASA, которому доверяет заявитель, для получения ваучеров. Идентификатор устройства IDevID обычно содержит MASA URL, как указано в параграфе 2.3. **Рекомендуется** применять этот механизм.

В некоторых случаях может быть сложно обеспечить требуемые расширения X.509 в IDevID заявителя из-за сложности согласования производства заявителя с разработкой и выпуском программного обеспечения. Поэтому в качестве финальной меры регистратор **можно** настроить вручную или распространять с MASA URL для каждого изготовителя. Отметим, что регистратор может выбирать настроенный MASA URL лишь на основе привязки доверия, поэтому изготовители могут применять такой подход лишь при наличии гарантии работы одного MASA URL для всех заявителей, связанных с каждой привязкой доверия.

[1]Cryptographic Message Syntax - синтаксис криптографический сообщений.

3. Артефакт запроса ваучера

Для запроса ваучеров применяется voucher-request. Семантика описана ниже в модуле YANG. Заявитель формирует voucher-request, подписывает его с помощью IDevID и представляет регистратору. Регистратор формирует свой voucher-request, подписывает его с использованием своих ключей и представляет MASA.

В voucher-request заявителя используется лист proximity-registrar-cert, позволяющий заявителю подтвердить близость регистратора. Эта сетевая близость (network proximity) является следствием свойств в контексте ACP - заявитель подключается к Join Proxy (раздел 4) с использованием соединения IPv6 по локальному каналу (link-local). Хотя Join Proxy не участвует значимо в криптографии соединения TLS (например, через Channel Binding), регистратор может видеть, что соединение выполняется через приватный адрес ACP (ULA) посредника присоединения и не может приходить извне ACP. Поэтому заявитель должен находиться не далее одного интервала пересылки (hop) IPv6 link-local от имеющегося узла ACP.

Другим пользователям BRSKI потребуется определить иные типы подтверждений, если описанная выше сетевая близость не подходит им.

Лист prior-signed-voucher-request используется в voucher-request регистратора. При наличии листа он является подписанным артефактом voucher-request от заявителя. Это обеспечивает регистратору способ пересылки подписанного заявителем запроса агенту MASA, завершающей передачу подписанного листа proximity-registrar-cert.

Если не указано иное (вне артефакта voucher-request), структура подписи соответствует заданной для ваучеров [RFC8366].

3.1. Ваучеры без Nonce

Регистратор **может** извлекать ваучеры без nonce, передавая voucher-request без nonce агенту MASA для получения ваучеров, применяемых при отсутствии у регистратора связности с MASA. Лист prior-signed-voucher-request в этом случае не включается. Регистратору также нужно знать серийный номер заявителя. Этот документ не задаёт механизма автоматического получения этих сведений. Обычно это делается путём сканирования штрихкода или QR-кода на упаковке или через интеграцию с каналами продаж.

3.2. Диаграмма дерева

На рисунке 5 приведено дерево высокоуровневого представления документа voucher-request, созданного на основе артефакта ваучера, описанного в [RFC8366]. Дерево использует нотацию и синтаксис [RFC8340]. Каждый узел полностью описан в параграфе 3.4. Модуль YANG.

```
module: ietf-voucher-request
 grouping voucher-request-grouping
  +-- voucher
     +-- created-on?                      yang:date-and-time
     +-- expires-on?                      yang:date-and-time
     +-- assertion?                       enumeration
     +-- serial-number                    string
     +-- idevid-issuer?                   binary
     +-- pinned-domain-cert?              binary
     +-- domain-cert-revocation-checks?   boolean
     +-- nonce?                           binary
     +-- last-renewal-date?               yang:date-and-time
     +-- prior-signed-voucher-request?    binary
     +-- proximity-registrar-cert?        binary
```

Рисунок 5. Дерево YANG для Voucher-Request.

3.3. Примеры

В этом параграфе приведены примеры, иллюстрирующие voucher-request в форме JSON до упаковки в CMS. Правила кодирования JSON задают представление двоичных объектов в формате base64 (раздел 4 в [RFC4648]). Содержимое кодированных (base64) сертификатов опущено для краткости, детальные примеры приведены в приложении C.2. Примеры следуют правилам кодирования, заданным в [RFC7951].

Пример 1

Ниже показан запрос voucher-request от заявителя. Лист assertion указан как proximity, а сертификат TLS регистратора включён в лист proximity-registrar-cert (5.2. Запрос заявителем ваучера у регистратора).

```
{
    "ietf-voucher-request:voucher": {
        "assertion": "proximity",
        "nonce": "62a2e7693d82fcda2624de58fb6722e5",
        "serial-number" : "JADA123456789",
        "created-on": "2017-01-01T00:00:00.000Z",
        "proximity-registrar-cert": "base64encodedvalue=="
    }
}
```

Рисунок 6. Представление JSON для примера Voucher-Request.

Пример 2

Ниже показан запрос voucher-request от регистратора. Лист prior-signed-voucher-request содержит voucher-request от заявителя (как в первом примере), который является двоичным объектом с подписью CMS. В используемой форме JSON он должен кодироваться в формате base64. Листья nonce и assertion из запроса заявителя перенесены в запрос регистратора. Значение serial-number извлечено из клиентского сертификата заявителя в соединении TLS (5.5. Запрос регистратором ваучера у MASA).

```
{
    "ietf-voucher-request:voucher": {
        "assertion" : "proximity",
        "nonce": "62a2e7693d82fcda2624de58fb6722e5",
        "created-on": "2017-01-01T00:00:02.000Z",
        "idevid-issuer": "base64encodedvalue==",
        "serial-number": "JADA123456789",
        "prior-signed-voucher-request": "base64encodedvalue=="
    }
}
```

Рисунок 7. Представление JSON для Prior-Signed Voucher-Request.

Пример 3

Ниже показан запрос voucher-request от регистратора. Лист prior-signed-voucher-request не содержит ни voucher-request от заявителя, ни листа nonce. Эта форма может применяться регистратором, запрашивающим ваучер, когда заявитель не может взаимодействовать с регистратором (например, выключен или ещё не распакован) и поэтому не может представить nonce. Этот вариант наиболее полезен в случаях, когда регистратору известно, что он не может получить доступ к MASA в процессе развёртывания (5.5. Запрос регистратором ваучера у MASA).

```
{
    "ietf-voucher-request:voucher": {
        "created-on":    "2017-01-01T00:00:02.000Z",
        "idevid-issuer": "base64encodedvalue==",
        "serial-number": "JADA123456789"
    }
}
```

Рисунок 8. Представление JSON для Offline Voucher-Request.

3.4. Модуль YANG

Ниже представлен модуль YANG [RFC7950] формально преобразующий ваучер [RFC8366] в voucher-request. Этот модуль YANG ссылается на [ITU.X690].

```
<CODE BEGINS> file "ietf-voucher-request@2021-05-20.yang"
module ietf-voucher-request {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-voucher-request";
  prefix vcr;

  import ietf-restconf {
    prefix rc;
    description
      "Этот оператор импорта предназначен лишь для доступа к
       расширению yang-data из RFC 8040.";
    reference
      "RFC 8040: RESTCONF Protocol";
  }
  import ietf-voucher {
    prefix vch;
    description
      "Этот модуль задаёт формат для ваучера, который изготовитель
       заявителя создаёт или делегирует (MASA) для защищённого
       назначения заявителя владельцем, чтобы тот мог организовать
       защищённое соединение с сетевой инфраструктурой владельца.";
    reference
      "RFC 8366: A Voucher Artifact for
       Bootstrapping Protocols";
  }

  organization
    "IETF ANIMA Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/anima/>
     WG List:  <mailto:anima@ietf.org>
     Author:   Kent Watsen
               <mailto:kent+ietf@watsen.net>
     Author:   Michael H. Behringer
               <mailto:Michael.H.Behringer@gmail.com>
     Author:   Toerless Eckert
               <mailto:tte+ietf@cs.fau.de>
     Author:   Max Pritikin
               <mailto:pritikin@cisco.com>
     Author:   Michael Richardson
               <mailto:mcr+ietf@sandelman.ca>";
  description
    "Этот модуль задаёт формат voucher-request, являющегося
     надмножеством voucher. Модуль обеспечивает агенту
     сведения для рассмотрения voucher-request.

     Ключевые слова ДОЛЖНО, НЕДОПУСТИМО, ТРЕБУЕТСЯ, НУЖНО, НЕ НУЖНО,
     СЛЕДУЕТ, НЕ СЛЕДУЕТ, РЕКОМЕНДУЕТСЯ, НЕ РЕКОМЕНДУЕТСЯ, МОЖНО,
     НЕОБЯЗАТЕЛЬНО в этом документе трактуются в соответствии с
     BCP 14 (RFC 2119) (RFC 8174) тогда и только тогда, когда они
     указаны заглавными буквами, как показано здесь.
```

```
      Авторские права (Copyright (c) 2021) принадлежат IETF Trust и
      лицам, указанным как авторы. Все права защищены.

      Распространение и применение модуля в исходной или двоичной
      форме с изменениями или без таковых разрешено в соответствии с
      лицензией Simplified BSD License, изложенной в параграфе 4.c
      IETF Trust's Legal Provisions Relating to IETF Documents
      (https://trustee.ietf.org/license-info).

      Эта версия модуля YANG является частью RFC 8995, где правовые
      аспекты приведены более полно.";

    revision 2021-05-20 {
      description
        "Исходный выпуск";
      reference
        "RFC 8995: Bootstrapping Remote Secure Key Infrastructure
         (BRSKI)";
    }

    // Оператор верхнего уровня
    rc:yang-data voucher-request-artifact {
      uses voucher-request-grouping;
    }

    // Группировка для будущего использования

    grouping voucher-request-grouping {
      description
        "Группировка для использования и расширения в будущем.";
      uses vch:voucher-artifact-grouping {
        refine "voucher/created-on" {
          mandatory false;
        }
        refine "voucher/pinned-domain-cert" {
          mandatory false;
          description
            "Поле pinned-domain-cert недействительно в voucher-request
             и ДОЛЖНО игнорироваться при наличии.";
        }
        refine "voucher/last-renewal-date" {
          description
            "Поле last-renewal-date недействительно в voucher-request
             и ДОЛЖНО игнорироваться при наличии.";
        }
        refine "voucher/domain-cert-revocation-checks" {
          description
            "Поле domain-cert-revocation-checks недействительно в
             voucher-request и ДОЛЖНО игнорироваться при наличии.";
        }
        refine "voucher/assertion" {
          mandatory false;
          description
            "Агенту MASA СЛЕДУЕТ игнорировать любые assertion в
             voucher-request от регистратора.";
        }
        augment "voucher" {
          description
            "Добавляет листью, подходящие для запроса ваучеров.";
          leaf prior-signed-voucher-request {
            type binary;
            description
              "Если требуется сменить ваучер или заново подписать и
               переслать ваучер, который был представлен с протокольным
               путём, прежний ваучер СЛЕДУЕТ включить в это поле.

               Например, заявитель может подписать voucher-request с
               proximity-registrar-cer и регистратор включит его как
               prior-signed-voucher-request. Это простой механизм для
               связывания доверенных сторон для смены voucher-request
               с сохранением сведений прежней подписи.

               Регистратор и MASA МОГУТ проверить сведения подписанного
               ранее ваучера в соответствии с правилами. Например, эти
               сведения могут быть полезны MASA для проверки согласия
               заявителя и регистратора с заявлением близости. MASA
               СЛЕДУЕТ удалять сведения prior-signed-voucher-request
               при подписании ваучера для впечатывания с целью
               сокращения окончательного размера вайчера.";
          }
          leaf proximity-registrar-cert {
            type binary;
            description
              "Структура сертификата X.509 v3, заданная в разделе 4
               RFC 5280, закодированная по правилам ASN.1 DER, как
               указано в ITU X.690.
```

```
                Первый сертификат в последовательности certificate_list
                TLS-сервера регистратора (сертификат TLS конечного
                элемента, см. RFC 8446), представленной регистратором
                заявителю. Лист ДОЛЖЕН заполняться в voucher-request
                заявителя при запросе заявления близости.";
              reference
                "ITU X.690: Information Technology - ASN.1 encoding
                rules: Specification of Basic Encoding Rules (BER),
                Canonical Encoding Rules (CER) and Distinguished
                Encoding Rules (DER)
                RFC 5280: Internet X.509 Public Key Infrastructure
                Certificate and Certificate Revocation List (CRL)
                Profile
                RFC 8446: The Transport Layer Security (TLS)
                Protocol Version 1.3";
            }
          }
        }
      }
    }
  }
<CODE ENDS>
```

Рисунок 9. Модуль YANG для Voucher-Request.

4. Детали посредничества (заявитель - прокси - регистратор)

Этот раздел является нормативным для использования с ANIMA ACP, применение механизма GRASP является частью ACP. Другим пользователям BRSKI нужно определить эквивалентный механизм посредничества для настройки прокси. Роль прокси состоит в облегчении взаимодействия путём пересылки между заявителем и регистратором пакетов, предоставленных посреднику через полное обнаружение GRASP ACP.

В этом разделе определён механизм посредничества в учётом состояния, называемый circuit proxy, который является формой шлюза прикладного уровня (Application Level Gateway, см. параграф 2.9 в [RFC2663]). Посредник не прерывает согласование TLS, он просто передаёт потоки данных без проверки. Посреднику **недопустимо** предполагать какую-либо конкретную версию TLS. Детали инвариантов TLS описаны в параграфе 9.3 [RFC8446].

Регистратор может напрямую представлять описанные ниже анонсы прокси и в этом случае анонсируемый порт указывает напрямую сам регистратор. В этом варианте заявитель не знает об отсутствии посредника. Это полезно для регистраторов, которые обслуживают заявителей в напрямую подключённых сетях.

В результате процесса обнаружения прокси, описанного в параграфе 4.1.1, раскрываемому посредником номеру порта не требуется быть общеизвестным или выделенным IANA.

В процессе обнаружения регистратора посредником присоединения этот Join Proxy узнает также, какие механизмы посредничества доступны. Это позволит Join Proxy использовать механизм с наименьшим влиянием, общий для Join Proxy и регистратора.

Чтобы позволить реализацию функциональности посредника для максимально широкого класса устройств следует применять механизм с минимальным числом состояний ну устройстве-посреднике. Хотя многие устройства в целевом пространства ANIMA будут скорее большими маршрутизаторами, функции посредника вероятно будут реализованы в CPU плоскости управления такого устройства с возможностями proxy-функции близкими к устройствам IoT класса 2.

В [ANIMA-STATE] приведён более детальный анализ и основы других методов посредничества.

4.1. Обнаружение посредника заявителем

Результатом обнаружения является логическое соединение с регистратором через посредника, который прозрачен для заявителя. Связь между заявителем и Join Proxy происходит по адресам IPv6 link-local. Для обнаружения посредника заявитель выполняет указанные ниже действия.

1. **Требуется** получить локальный адрес с использованием методов IPv6, описанных в IPv6 Stateless Address Autoconfiguration [RFC4862]. Рекомендуется применять временные адреса [RFC8981]. Для ограничения всеобъемлющего отслеживания [RFC7258] новые временный адрес **может** применяться кратковременно (малое значение TEMP_PREFERRED_LIFETIME). Заявители обычно будут предпочитать адреса IPv6 link-local и искать посредника с помощью механизмов локального канала. Методы для IPv4 описаны в Приложении A.
2. **Должны** прослушиваться анонсы GRASP M_FLOOD [RFC8990] для цели AN_Proxy, более подробно описанной в параграфе 4.1.1. Анонсирование Proxy GRASP. Заявитель **может** одновременно прослушивать другие источники информации (Приложение B. Варианты mDNS и DNS-SD Proxy Discovery).

После обнаружения посредника заявитель взаимодействует с регистратором через прокси, используя протокол начальной загрузки, заданный в разделе 5. Детали протокола (заявитель - регистратор - MASA).

Хотя механизм GRASP M_FLOOD является пассивным для заявителя, другие ненормативные методы (mDNS и IPv4), описанные в Приложении B, являются активными. Заявителю **следует** запускать эти методы вместе с прослушиванием M_FLOOD. Активным методам **следует** удваивать время отсрочки (вплоть до 1 часа) для предотвращения перегрузки сети попытками обнаружения. При обнаружении смены состояния физического канала (например, несущей Ethernet) **следует** сбрасывать таймеры отсрочки.

Заявитель может обнаружить на данном физическом интерфейсе более одного посредника. Кроме того, у заявителя может быть несколько физических интерфейсов, а коммутатор Ethernet L2/3 может иметь сотни физических портов. Каждый предложенный прокси **следует** пытаться использовать, пока не будет получен пригодный ваучер. Имеется множество вариантов получения отказа, но успехом является лишь получение действительного ваучера.

Интервалы между попытками соединения через один прокси **следует** экспоненциально увеличивать вплоть до 1 часа для предотвращения перегрузки сетевой инфраструктуры. Таймеры отсрочки для каждого соединения **должны** быть независимыми. Попытки соединения **следует** предпринимать в параллель для предотвращения проблем в голове очереди, когда атакующий с фиктивным посредником или регистратором может преднамеренно замедлять операции протокола. Попытки соединения с разными прокси **следует** предпринимать с интервалом от 3 до 5 секунд. Заявителю **следует** продолжать прослушивание дополнительных сообщений GRASP M_FLOOD при попытках соединения. Для каждой попытки соединения через свой Join Proxy **должно** применяться уникальное значение nonce в voucher-request.

Когда соединение с регистратором установлено (например, создан сеансовый ключ TLS), ожидается более скорый отклик (5.2. Запрос заявителем ваучера у регистратора).

После того, как все обнаруженные службы использованы (безуспешно), устройство **должно** вернуться к прослушиванию GRASP M_FLOOD. Ему **следует** повторять любые заданные изготовителем механизмы. Заявитель **может** приоритизировать порядок выбора в соответствии с ожидаемой средой.

4.1.1. Анонсирование Proxy GRASP

Посредник использует для анонсирования себя механизм DULL GRASP M_FLOOD. Этот анонс может находиться в одном сообщении с анонсом ACP, описанным в [RFC8994]. Формальное определение CDDL[1] [RFC8610] имеет вид

```
<CODE BEGINS> file "proxygrasp.cddl"
flood-message = [M_FLOOD, session-id, initiator, ttl,
                 +[objective, (locator-option / [])]]

objective = ["AN_Proxy", objective-flags, loop-count,
                                     objective-value]

ttl             = 180000     ; 180000 мсек (3 минуты)
initiator = адрес ACP для контакта с регистратором
objective-flags   = sync-only  ; как в спецификации GRASP
sync-only         =  4         ; M_FLOOD требует лишь синхронизации
loop-count        =  1         ; только один интервал пересылки (hop)
objective-value   =  any       ; нет

locator-option    = [ O_IPv6_LOCATOR, ipv6-address,
                      transport-proto, port-number ]
ipv6-address      = адрес IPv6 LL для Proxy
$transport-proto /= IPPROTO_TCP  ; Это может быть любое значение из
                                 ; реестра протоколов IANA, как в
                                 ; параграфе 2.9.5.1 RFC 8990
                                 ; (примечание 3).
port-number      = выбирается Proxy
<CODE ENDS>
```

Рисунок 10. Определение CDDL для сообщения Proxy Discovery.

Ниже приведён пример анонса M_FLOOD для посредника с адресом fe80::1 на порту TCP 4443.

```
[M_FLOOD, 12340815, h'fe80000000000000000000000000001', 180000,
           [["AN_Proxy", 4, 1, ""],
            [O_IPv6_LOCATOR,
             h'fe80000000000000000000000000001', IPPROTO_TCP, 4443]]]
```

Рисунок 11. Пример сообщения Proxy Discovery.

В небольшой сети регистратор **может** включать анонсы GRASP M_FLOOD в локально подключённые сети. Поле $transport-proto выше указывает метод, который будет использовать pledge-proxy-registrar. Описанный здесь метод TCP является обязательным, а другие прокси-методы, такие как методы CoAP, не определённые в этом документе, не являются обязательными. Другие методы **недопустимо** включать, пока Join Registrar ASA не указывает их поддержку в своих анонсах.

4.2. Соединение CoAP с регистратором

Использование CoAP для соединения заявителя с регистратором выходит за рамки этого документа и рассматривается в [ANIMA-CONSTRAINED-VOUCHER].

4.3. Обнаружение посредника и связь с регистратором

Регистратору следует анонсировать себя, чтобы посредники могли найти его и определить, какой тип соединения может быть завершён. Регистратор анонсирует себя в сообщениях GRASP M_FLOOD с целью AN_join_registrar в экземпляре ACP. Регистратор может анонсировать любой подходящий номер порта, включая стандартный порт 443. Посредники ANI **должны** поддерживать обнаружение регистраторов GRASP. Формат M_FLOOD показан ниже.

```
[M_FLOOD, 51804321, h'fda379a6f6ee00000200000064000001', 180000,
           [["AN_join_registrar", 4, 255, "EST-TLS"],
            [O_IPv6_LOCATOR,
             h'fda379a6f6ee00000200000064000001', IPPROTO_TCP, 8443]]]
```

Рисунок 12. Пример сообщения Registrar Announcement.

Формальное определение CDDL приведено ниже.

```
<CODE BEGINS> file "jrcgrasp.cddl"
flood-message = [M_FLOOD, session-id, initiator, ttl,
                 +[objective, (locator-option / [])]]
```

[1]Concise Data Definition Language - краткий язык определения данных.

```
objective = ["AN_join_registrar", objective-flags, loop-count,
                                    objective-value]

initiator = адрес ACP для контактов с регистратором
objective-flags = sync-only  ; как в спецификации GRASP
sync-only         =  4       ; M_FLOOD требует лишь синхронизации
loop-count      = 255        ; обязательный максимум
objective-value = text       ; имя (или список имен) поддерживаемого
                             ; протокола (EST-TLS для RFC 7030).
<CODE ENDS>
```

Рисунок 13. Определение CDDL для сообщения Registrar Announcement.

Сообщения M_FLOOD **должны** передаваться периодически. По умолчанию **следует** устанавливать 60 секунд и значение **следует** делать настраиваемым оператором, но **не следует** устанавливать значения меньше 60 секунд. Частота отправки **должна** быть такой, что общее число периодических M_FLOOD от всех источников лавинной рассылки не вызывало значительного трафика через ACP.

Ниже приведены некоторые примеры локаторов для иллюстрации, из которых лишь первый ($transport-protocol = 6, TCP) определён в этом документе и обязателен для регистрации.

```
locator1  = [O_IPv6_LOCATOR, fd45:1345::6789, 6,  443]
locator2  = [O_IPv6_LOCATOR, fd45:1345::6789, 17, 5683]
locator3  = [O_IPv6_LOCATOR, fe80::1234, 41, nil]
```

Протокол 6 указывает, что желательно посредничество TCP на указанном порту.

Регистраторы **должны** анонсировать набор поддерживаемых протоколов и должны поддерживать трафик TCP.

Регистраторы **должны** воспринимать трафик HTTPS/EST на указанном порту TCP.

Регистраторы **должны** поддерживать ANI TLS Circuit Proxy и, следовательно, BRSKI через HTTPS/TLS в ACP.

В ANI **должен** использоваться защищённый ACP экземпляр GRASP [RFC8990] для обнаружения адресов ANI регистратора ACP и портов посредников ANI. Поэтому ветвь TCP прокси-соединения между посредником ANI и регистратором ANI также работает через ACP.

5. Детали протокола (заявитель - регистратор - MASA)

Заявитель **должен** инициировать BRSKI после загрузки, если это ещё не настроено. заявителю **недопустимо** автоматически инициировать BRSKI, если инфраструктура уже настроена или настраивается сейчас.

Инфраструктура BRSKI описана как расширение EST [RFC7030]. Цель этих расширений заключается в сокращении числа соединений TLS и криптоопераций, требуемых для заявителя. Регистратор реализует интерфейс BRSKI REST в дереве URI /.well-known/brski и реализует имеющиеся EST URI, как указано в параграфе 2.2.2 EST [RFC7030]. Канал связи между заявителем и регистратором обозначается как BRSKI-EST (Рисунок 1).

Каналом связи между регистратором и MASA является новый коммуникационный канал, аналогичный EST, внутри вновь зарегистрированного дерева /.well-known/brski. Для ясности этот канал обозначен как BRSKI-MASA (Рисунок 1).

MASA URI имеет вид https://" authority "/.well-known/brski.

BRSKI использует имеющиеся форматы сообщений CMS для существующих операций EST и JSON [RFC8259] для всех новых операций, определённых здесь для форматов ваучеров. Во всех случаях, когда двоичное значение должно быть передано в строке JSON, применяется формат base64 (раздел 4 в [RFC4648]) в соответствии с параграфом 6.6 [RFC7951].

Хотя EST (параграф 3.2 в [RFC7030]) не настаивает на использовании постоянных соединений HTTP (параграф 6.3 в [RFC7230]), соединениям BRSKI-EST **следует** применять их. Цель этого руководства состоит в том, чтобы гарантировать возникновение состояния готовности TLS лишь один раз и предотвратить MITM[1]-атаки на последующее распознавание состояния готовности во время критической фазы.

Если применяются непостоянные соединения, заявитель и регистратор **должны** помнить сертификаты, которые были просмотрены, а также переданы для первого соединения. Они **должны** проверять в каждом последующем соединении наличие тех же сертификатов и каждая сторона **должна** использовать одни и те же сертификаты. Это задаёт сложное ограничение для чередования сертификатов на регистраторе.

Обобщённые расширения автоматизации для потока BRSKI-EST перечислены ниже.

- Заявитель пытается установить одновременные соединения через все обнаруженные прокси или быстро наступает тайм-аут и соединения организуются последовательно, как указано в параграфе 5.1.
- Заявитель воспринимает сертификат регистратора при согласовании TLS, как указано в параграфе 5.1.
- Заявитель запрашивает ваучер, используя новые вызовы REST, описанные ниже. Ваучер затем проверяется.
- Заявитель завершает аутентификацию сертификата сервера, как указано в параграфе 5.6.1. Это выводит соединения BRSKI-EST TLS из условного (provisional) состояния.
- Обязательные этапы начальной загрузки завершаются телеметрией статуса ваучера (параграф 5.7).

Соединение BRSKI-EST TLS теперь может служить для зачисления EST. Расширения для регистратора (эквивалент сервера EST) указаны ниже.

- Аутентификация клиента автоматизируется использованием IDevID как при аутентификации клиента EST по на основе сертификатов. Кодирование DN поля субъекта **должно** включать атрибут serialNumber с уникальным серийным номером устройства, как описано в параграфе 2.3.1. Идентификация заявителя.

[1]Man-in-the-Middle - перехват и изменение данных с участием человека.

- Регистратор запрашивает ваучер у MASA и проверяет его.
- Регистратор пересылает ваучер заявителю по запросу.
- Регистратор проверяет журнал (log), как указано в параграфе 5.8.3, в дополнение к локальным проверкам полномочий, прежде чем воспринимать необязательные запросы заявителя на зачисление его устройства.

5.1. Организация BRSKI-EST TLS

Заявитель организует соединение TLS с регистратором через Circuit Proxy (см. раздел 4), но согласование TLS происходит с регистратором. Заявитель BRSKI-EST является клиентом TLS, а регистратор BRSKI-EST - сервером TLS. Все защищённые связи существуют между заявителем и регистратором независимо от операций посредника.

Рекомендуется применять TLS 1.3 (или новее), на стороне заявителя **требуется** TLS 1.2 или новее. На интерфейсе сервера и клиента у регистратора **следует** поддерживать TLS 1.3 (или новее), но **можно** применять TLS 1.2.[1] При использовании TLS 1.3 применять индикатор имени сервера (Server Name Indicator или SNI, [RFC6066]) не требуется в соответствии с параграфом 9.2 в RFC8446 и эта спецификация задаёт профиль приложения. Заявитель соединяется с регистратором, используя лишь адрес IP, и не имеет представления о корректном значении SNI. Это также предполагает, что интерфейс регистратора не может быть виртуальным, поддерживаемым с использованием SNI.

Организация соединения BRSKI-EST TLS происходит в соответствии с параграфом 4.1.1 Bootstrap Distribution of CA Certificates в [RFC7030], при этом клиент аутентифицируется по сертификату IDevID, а сервер EST (регистратор) аутентифицирован ранее по непроверенному сертификату сервера. Настройка и распространение базы данных о привязках доверия, служащей для проверки сертификата IDevID, выходит за рамки этой спецификации. Отметим, что привязки доверия в корне (/), исключённые из базы данных, будут влиять на выбор устройств изготовителя, воспринимаемых регистратором как заявители, и могут также служить для ограничения набора MASA, которым доверяют при зачислении.

Подпись в сертификате **должна** проверяться, даже если ключ подписи (еще) не может быть проверен. Сертификат (или цепочка) **должен** сохраняться для последующей проверки.

Приемлем самоподписанный сертификат регистратора, поскольку ваучер может подтвердить его при успешном зачислении.

Заявитель выполняет входную проверку всех данных, полученных, пока ваучер не проверен, как указано в параграфе 5.6.1 и соединение TLS не выйдет из представленного состояния. Пока эти операции не завершены, заявитель может обмениваться данными со злоумышленником.

В коде заявителя должна учитываться возможность передачи в этот момент всех данных неаутентифицированному партнёру, а данные, принятые в соединении TLS, **должны** считаться недоверенными. Это особенно важно для заголовков HTTP и структур CMS, из которых состоит ваучер.

Заявителю, способному одновременно соединиться с несколькими регистраторами, **следует** делать это. Некоторые устройства могут быть неспособны на это из-за нехватки потоков или проблем с ресурсами. Параллельные соединения препятствуют попыткам вредоносных посредников вызвать атаки, подобные TCP Slowloris [slowloris].

Заявитель, не способный поддерживать число соединений по числу подходящих посредников, должен будет чередовать варианты, прерывая «заторможенные» соединения. Если в течение 5 секунд соединение «не продвинулось», заявителю **следует** отбросить наиболее старое соединение и перейти к другому посреднику (тому, с которым связывались в последний раз). Если не было найдено других посредников, заявитель **может** продолжить ожидание, одновременно прослушивая анонсы новых посредников.

5.2. Запрос заявителем ваучера у регистратора

При начальной загрузке заявитель запрашивает ваучер у регистратора с использованием HTTPS POST с путём к операции /.well-known/brski/requestvoucher. Тип носителя (Content-Type) в voucher-request показан ниже.

application/voucher-cms+json
: [RFC8366] задаёт «определённый с использованием YANG документ JSON, подписанный структурой CMS», а voucher-request, описанный в разделе 3, создаётся таким же способом. Тип носителя совпадает с заданным в [RFC8366]. Это же применяется для voucher-request от заявителя. Заявитель **должен** подписать запрос, используя свидетельства, указанные в параграфе 2.3. Исходный идентификатор устройства.

Реализациям регистраторов **следует** предполагать будущие типы носителей, но могут, конечно, просто возвращать отказ, если тип ещё не известен.

Заявителю **следует** включать поле заголовка Accept (см. параграф 5.3.2 в [RFC7231]), указывающее приемлемый для отклика с ваучером тип носителя. Тип application/voucher-cms+json определён в [RFC8366], но в будущем ожидаются ограниченные форматы ваучеров. Предполагается гибкость регистраторов и MASA при их восприятии.

Заявитель заполняет поля voucher-request, указанные ниже.

created-on
: Заявителю, у которого есть часы, **рекомендуется** указывать в этом поле текущую дату и время в формате yang:date-and-time, дающие дополнительные сведения агенту MASA. Заявители без часов **могут** опускать это поле.

nonce
: Запрос ваучера от заявителя **должен** содержать криптографически строгое случайное или псевдослучайное значение nonce (параграф 6.2 в [RFC4086]). Поскольку nonce обычно создаётся в самом начале загрузки, имеются опасения, что одно значение nonce может создаваться при нескольких загрузках или после сброса к заводским установкам. Для каждой попытки начальной загрузки **должно** создаваться своё значение nonce при параллельных или последовательных загрузках. Свежесть nonce смягчает отсутствие часов, как указано в параграфе 2.6.1.

[1] В оригинале последующий текст этого абзаца отличается. См. https://www.rfc-editor.org/errata/eid6648. *Прим. перев.*

assertion
Заявитель указывает поддержку описанного в этом документе механизма, помещая значение proximity в voucher-request, и **должен** включать поле proximity-registrar-cert (см. ниже).

proximity-registrar-cert
В voucher-request заявителя это первый сертификат в последовательности сервера TLS certificate_list (параграф 4.4.2 в [RFC8446]), представленной регистратором заявителю. Это сертификат конечного элемента (end-entity), который **должен** указываться в voucher-request заявителя.

serial-number
Серийный номер заявителя включается в voucher-request от заявителя. Это значение включается лишь для проверки пригодности и не пересылается регистратором, как указано в параграфе 5.5.

Остальные поля **могут** быть опущены в voucher-request заявителя.

Пример JSON для voucher-request заявителя представлен в параграфе 3.3 (пример 1).

Регистратор подтверждает, что заявление является proximity и прикрепленный proximity-registrar-cert является сертификатом регистратора. Если эта проверка завершается отказом, это говорит о наличии злоумышленника в пути (MITM) и соединение **должно** быть закрыто после возврата кода HTTP 401.

5.3. Проверка полномочий заявителя регистратором

В полностью автоматизированной сети все устройства должны быть защищённо идентифицированы и их полномочия должны быть проверены до присоединения к домену.

Регистратор воспринимает или отклоняет запрос на присоединение к домену на основе представленного аутентифицированного отождествления. Для разных сетей примеры автоматизированного принятия запросов могут включать:

- любое устройство заданного типа (как указано X.509 IDevID);
- любое устройство определённого производителя (как указано X.509 IDevID);
- конкретное устройство производителя (как указано X.509 IDevID) из принятого в домене списка (механизм проверки по этому списку для нескольких регистраторов выходит за рамки документа).

Если проверка не проходит, регистратору **следует** отвечать кодом HTTP 404. Если voucher-request имеет неизвестный формат, лучше подходит код HTTP 406. В ситуациях, которые можно разрешить административными мерами (например, добавлением производителя в список принимаемых) **может** возвращаться отклик HTTP 403.

При успешной проверке полномочий регистратор получает ваучер от службы MASA (5.5. Запрос регистратором ваучера у MASA) и возвращает заявителю подписанный MASA ваучер, как указано в параграфе 5.6.

5.4. Организация BRSKI-MASA TLS

Соединение BRSKI-MASA TLS является «обычным» соединением TLS, подходящим для интерфейсов HTTPS REST. Регистратор инициирует соединения и использует идентификатор MASA URL, полученный, как указано в параграфе 2.8. Механизмы [RFC6125] **следует** применять при аутентификации MASA с использованием DNS-ID, соответствующего найденному в IDevID. Регистраторы **могут** включать механизм переопределения MASA URL для каждого изготовителя и при таком переопределении целесообразно предоставлять альтернативные привязки. Обычно это применяется некоторыми производителями для создания явных (или приватных) привязок доверия для проверки их MASA, которые являются частью интеграции с каналами продаж.

При недоступности TLS 1.3 **требуется** TLS 1.2 [RFC5246] с поддержкой SNI [RFC6066]. Индикатор имени сервера (SNI) требуется, когда регистратор взаимодействует с MASA, чтобы разместить агент MASA в инфраструктуре TLS с множеством арендаторов.[1]

Как описано в [RFC7030], MASA и регистраторам **следует** быть готовыми к поддержке аутентификации клиентского сертификата TLS и/или использованию механизмов HTTP Basic, Digest или Salted Challenge Response Authentication Mechanism. В соединении аутентификация клиента **может** не использоваться.

Регистраторам **следует** разрешать предварительную настройку привязок доверия по производителям (MASA). Регистраторам **следует** иметь возможность настройки TLS Client Certificate по MASA или не использовать сертификаты клиентов, а также **следует** поддерживать настройку аутентификации HTTP Basic и Digest.

Аутентификация соединения BRSKI-MASA не меняет процесс запроса ваучера, поскольку voucher-request уже подписан регистратором. Аутентификация обеспечивает контроль доступа к журналу аудита, как описано в параграфе 5.8. Запрос журнала аудита у регистратора.

Разработчикам рекомендуется обеспечивать возможность при обращении к MASA создавать защищённое API-соединение с web-службой и учитывать, что в отрасли поддерживается несколько моделей аутентификации. Регистраторам **рекомендуется** корректно завершать работу и создавать полезные административные уведомления или записи в журнале (log) при получении неожиданных откликов HTTP 401 (Unauthorized) от MASA.

5.4.1. Аутентификация агентом MASA регистратора клиента

Предоставление опций по клиентам требует однозначной идентификации клиентских регистраторов. Это можно сделать любым методом без сохранения состояний, который поддерживает HTTPS (например, аутентификация HTTP Basic или Digest, т. е. использование паролей), но **рекомендуется** применять TLS Client Certificate.

Методы с поддержкой состояний, включающие маркеры API или HTTP Cookie, не рекомендуются.

Предполагается, что установка и настройка Client Certificate выполняется как часть процесса продажи.

[1] В оригинале это предложение указано иначе. См. https://www.rfc-editor.org/errata/eid6642. *Прим. перев.*

Использование сертификатов конечных объектов из общедоступных PKI (т. е. WebPKI) для идентификации регистратора является разумным и при повсеместном применении это позволит MASA идентифицировать клиентские регистраторы просто по полным доменным именами (Fully Qualified Domain Name или FQDN).

Использование записей DANE в подписанных DNSSEC зонах также позволяет применять FQDN для идентификации клиентских регистраторов.

Третьим (самым простым, но наименее гибким) механизмом является просто хранение в MASA сертификатов регистраторов, закреплённых в базе данных.

MASA без интеграции с цепочкой поставок может просто воспринимать регистраторы без аутентификации или вслепую на основе TOFU, как описано в параграфе 7.4.2. Доверие при первом использовании.

Этот документ не даёт конкретных рекомендаций по способу аутентификации регистраторов агентом MASA, поскольку возможны различные компромиссы в разных средах для различной продукции. Даже в сфере применимости ANIMA ACP имеются существенные различия между логистикой цепочек поставок простых устройств CPE за $100 и маршрутизаторов ядра стоимостью $100000 core.

5.5. Запрос регистратором ваучера у MASA

При получении регистратором voucher-request от заявителя он представляет свой voucher-request службе MASA через интерфейс HTTPS [RFC7231] с помощью HTTP POST с путём к операции /.well-known/brski/requestvoucher. Тип носителя для ваучера application/voucher-cms+json определён в [RFC8366] и применяется также в запросе ваучера регистратором. Это документ JSON, подписанный с использованием структуры CMS. Регистратор **должен** подписывать свой voucher-request.

Реализациям MASA **следует** предвидеть будущие типы носителей, но они, конечно, могут просто возвращать отказ при получении запроса с ещё неизвестным типом.

Объект CMS в voucher-request включает некоторое число сертификатов, которые служат вводом для MASA, поскольку агент заполняет pinned-domain-cert. [RFC8366] обеспечивает достаточную гибкость в части содержимого pinned-domain-cert, поэтому агенту MASA требуется некий сигнал о типе сертификата, подходящем для заполнения поля. Это может варьироваться от сертификата конечного объекта, используемого регистратором, до всего частного сертификата Enterprise CA. Более конкретные сертификаты усиливают привязку ваучера к домену, а менее конкретные повышают гибкость представления домена сертификатами.

Регистратор, ищущий ваучер без nonce для последующего автономного (offline) применения, выигрывает от менее конкретного сертификата, поскольку он позволяет прикреплённому CA определить фактическую пару ключей, применяемую будущим регистратором.

В некоторых случаях менее конкретный сертификат, такой как WebPKI CA, может быть слишком открытым и может позволять стать владельцем устройства, для которого закреплён ваучер, любому объекту с сертификатом от этого органа. В будущих работах может быть представлено решение для закрепления сертификата и имени, что снизит риск обмана при заявлении прав владения.

Регистратору **следует** запрашивать наиболее конкретный ваучер, соответствующий режиму работы. Для этого при подготовке структуры CMS для подписанного voucher-request регистратору **следует** включать лишь сертификаты, являющиеся частью цепочки, желаемой для закрепления агентом MASA. Это **может** оказаться лишь одним сертификатом конечного объекта (с id-kp-cmcRA), используемым как сертификат сервера TLS, или целой цепочкой, включающей CA домена.

Регистратору **следует** включать поле заголовка Accept (параграф 5.3.2 в [RFC7231]) указывающее приемлемые типы носителя. Этому списку **следует** быть полным списком, представленным регистратору в исходном запросе заявителя (5.2. Запрос заявителем ваучера у регистратора), но **может** указываться лишь часть. Предполагается, что агент MASA достаточно гибок при восприятии.

Регистратор заполняет поля voucher-request, как указано ниже.

created-on
: Регистратору **следует** указывать в этом поле дату и время формирования voucher-request. Поле обеспечивает дополнительные сведения для MASA.

nonce
: Это значение копируется из voucher-request заявителя (при наличии) и **может** быть опущено в voucher-request регистратора (см. 3.1. Ваучеры без Nonce).

serial-number
: Серийный номер заявителя, для которого регистратор хочет получить ваучер. Регистратор определяет это значение путём анализа аутентифицированного сертификата IDevID от заявителя (см. 2.3. Исходный идентификатор устройства). Регистратор **должен** убедиться, что найденное поле serial-number соответствует полю serial-number, представленному заявителем в его voucher-request. Это обеспечивает проверку пригодности, полезную для обнаружения ошибок и ведения журнала. Регистратору **недопустимо** просто копировать поле serial-number из voucher-request заявителя, поскольку оно не заверяется.

idevid-issuer
: Значение Issuer из сертификата заявителя IDevID включается для однозначной интерпретации serial-number. В случае voucher-request без nonce требуется задать подходящее значения из того внешнего (out-of-band) же источника, что и serial-number.

prior-signed-voucher-request
: **Следует** включать подписанный запрос voucher-request от заявителя в voucher-request регистратора. Подписанная CMS структура с кодированием base64 включается целиком для передачи в структуре JSON.

Агенту MASA **можно** представлять запрос voucher-request от регистратора без nonce. Это позволяет регистратору запросить ваучер отключённого (offline) заявителя или в предположении отсутствия связи с MASA во время развёртывания заявителя. В некоторых случаях регистратора должен знать соответствующие IDevID serialNumber и поле заголовка Accept из маркировки физического устройства или канала продаж (выходит за рамки документа).

Остальные поля voucher-request от регистратора **могут** быть опущены.

Поле proximity-registrar-cert **недопустимо** включать в voucher-request регистратора.

Примеры JSON для voucher-request от регистратора приведены в параграфе 3.3 (примеры 2 - 4).

MASA проверяет внутреннюю согласованность voucher-request от регистратора, но не обязательно аутентифицирует сертификат регистратора, поскольку регистратор **может** быть неизвестен агенту MASA заранее. Перед выдачей ваучера MASA выполняет действия и проверки, описанные в последующих параграфах.

5.5.1. Обновление агентом MASA просроченных ваучеров

Как описано в [RFC8366], ваучеры обычно имеют короткий срок действия, чтобы избежать проблем с отзывом. Если запрос для прежнего (просроченного) ваучера передан тем же регистратором (регистратором с тем же CA домена), запросу обновлённого ваучера **следует** предоставлять полномочия автоматически. Агент MASA имеет достаточно сведений для определения этого путём проверки запроса на предмет аутентификации регистратора и имеющегося журнала аудита (audit-log). Выпуск обновлённого ваучера записывается в журнал (5.6. MASA и отклик с ваучером регистратора).

Для информирования MASA о том, что имеющиеся ваучеры не будут обновляться можно обновить или отозвать свидетельства регистратора, служащие для предоставления полномочий запросу (параграфы 5.5.4 и 5.5.3). Более гибкие методы будут, скорей всего, включать интеграцию с каналами продаж и предоставлением полномочий, но это выходит за рамки данного документа.

5.5.2. Закрепление регистратора агентом MASA

Цепочка сертификатов извлекается из подписанного CMS контейнера регистратора. Эта цепочка может содержать от одного сертификата конечного объекта до всей цепочки сертификатов регистратора, включая сертификат доменного CA, как указано в параграфе 5.5. Запрос регистратором ваучера у MASA.

Если CA домена неизвестен MASA, он считается временной привязкой доверия для остальных шагов, рассматриваемых в этом параграфе. Намерение состоит не в аутентификации сообщения как полученного из полностью проверенного источника, а в установке согласованности PKI домена.

MASA **может** использовать сертификат из цепочки, наиболее удалённый от сертификата конечного объекта для регистратора, в соответствии с политикой MASA. Для MASA **может** быть задана локальная политика, в которой закрепляются только сертификаты конечных элементов. Это согласуется с [RFC8366]. Детали политики обычно зависят от уровня интеграции с цепочкой поставок и механизма, используемого регистратором для аутентификации. Политика будет также определять реакцию MASA на запросы ваучеров без nonce.

5.5.3. Проверка агентом MASA подписи в Voucher-Request

Как указано в параграфе 5.5.2, MASA определяет CA домена регистратора, используемый для проверки подписи CMS [RFC5652] в voucher-request.

При проверке подписи voucher-request предполагается обычная проверка отзывов PKIX. Сертификат CA **может** иметь точки распространения списков отозванных сертификатов (Certificate Revocation List или CRL) или сведения протокола состояния сертификатов (Online Certificate Status Protocol или OCSP) [RFC6960]. При наличии таких данных агент MASA **должен** иметь доступ к соответствующим серверам, относящимся к CA домена регистратора для проверки отзыва.

Предпочтительно использовать OCSP Stapling.

5.5.4. Проверка агентом MASA регистратора домена

Агент MASA **должен** проверить, что voucher-request от регистратора подписан тем. Это подтверждается проверкой наличия поля расширенного использования ключа id-kp-cmcRA (см. параграф 3.6.1 в EST [RFC7030]) в сертификате объекта, подписавшего voucher-request от регистратора. Это проверяет лишь то, что неаутентифицированный CA домена для стороны, подписавшей voucher-request, является регистратором. Выполнение такой проверки повышает ценность PKI домена, гарантируя администратору домена, что служба MASA будет учитывать лишь требования уполномоченных органов регистрации в домене.

Даже при аутентификации CA домена агентом MASA и надёжной интеграции с каналом продаж для понимания, кто является законным владельцем, указанная выше проверка id-kp-cmcRA предотвращает наличие ваучеров, выпущенных с произвольными сертификатами конечных элементов (такими как LDevID).

Значение id-kp-cmcRA является атрибутом расширенного использования ключа (Extended Key Usage или EKU). При установке какого-либо атрибута EKU сертификат **должен** иметь все связанные с этим атрибуты. Это значит, что сертификат регистратора **должен** иметь id-kp-clientAuth (для MASA) и id-kp-serverAuth (для заявителя).[1]

Другие случаи неправомерного выпуска ваучеров обнаруживает проверка журнала аудита (audit-log).

При представлении voucher-request без сертификата агент MASA **должен** аутентифицировать регистратор, как описано в EST (параграфы 3.2.3 и 3.3.2 в [RFC7030]), или путём проверки сертификата регистратора, использованного для подписи voucher-request, с применением настроенной привязки доверия. Любой из этих методов снижает риск атак DDoS и представляет аутентифицированное отождествление на вход интеграции с каналом продаж и проверки полномочий (детали этого выходят за рамки документа).

При использовании nonce проверку отождествления регистратора (через TLS Client Certificate или аутентификацию HTTP) **можно** опустить, если MASA знает, что политика устройства задаёт восприятие ваучеров лишь для аудита.

[1] В оригинале этот абзац отсутствует. См. https://www.rfc-editor.org/errata/eid6649. *Прим. перев.*

5.5.5. Проверка агентом MASA prior-signed-voucher-request от заявителя

MASA **может** проверить, что voucher-request от регистратора включает поле prior-signed-voucher-request. При наличии этого поля оно **должно** включать сертификат proximity-registrar-cert, соответствующий сертификату, использованному для voucher-request от регистратора. Кроме того, лист serial-number в voucher-request **должен** соответствовать serial-number от заявителя, который MASA извлекает из подписанного сертификата в prior-signed-voucher-request. Описанная выше проверка соответствия включает проверку наличия в proximity-registrar-cert отпечатка открытого ключа субъекта (Subject Public Key Info или SPKI) в цепочке сертификатов подписи CMS для voucher-request от регистратора. Это практически совпадает с проверкой закрепления (pin), описанной в параграфе 2.6 [RFC7469].

После успешной проверки MASA обновляет листья ваучера и утверждения audit-log утверждением proximity, как указано в параграфе 5.3 [RFC8366].

5.5.6. Обработка MASA Nonce

MASA не проверяет значение nonce. Если voucher-request от регистратора включает nonce и имеется prior-signed-voucher-request, агент MASA **должен** проверить согласованность nonce (напомним, что voucher-request может не включать nonce, см. 5.5. Запрос регистратором ваучера у MASA и 5.5.4. Проверка агентом MASA регистратора домена).

MASA помещает в audit-log проверенное значение nonce. Если ваучер не включает nonce, в audit-log помещается JSON-значение null.

5.6. MASA и отклик с ваучером регистратора

Отклик MASA с ваучером для регистратора пересылается без изменений заявителю, поэтому данный параграф применим для MASA и регистратора. Описанная сигнализация HTTP применима к откликам MASA и регистратора.

Когда voucher-request поступает к регистратору и у того имеется кэшированный отклик MASA для соответствующего voucher-request от регистратора, этот отклик можно использовать в соответствии с локальной политикой. В иных случаях регистратор создаёт новый запрос voucher-request и передаёт его агенту MASA.

Оценка ваучера регистратором выполняется лишь в целях прозрачности и аудита для последующей проверки журнала (5.8.3. Проверка журнала аудита у регистратора), поэтому регистратор может принимать будущие форматы ваучера, непонятные ему.

При успехе запроса voucher-request отклик сервера (агент MASA, отвечающий регистратору, или регистратор, отвечающий заявителю) **должен** содержать код HTTP 200. Сервер **должен** отвечать подходящим кодом HTTP 4xx или 5xx [RFC7230] при возникновении проблем. В этом случае данные отклика от MASA **должны** быть понятным человеку текстом (UTF-8) сообщения об ошибке, разъясняющим причины отклонения запроса.

Регистратор **может** отвечать кодом HTTP 202 (запрос принят для обработки, но она ещё не завершена), как описано в параграфе 4.2.3 EST [RFC7030], а клиент «**должен** ждать в течение по меньшей мере retry-after перед повторением запроса» (см. параграф 6.6.4 в [RFC7231]). Заявителю **рекомендуется** предоставлять локальный отклик (мигание светодиода и т. п.) в процессе ожидания, если для этого имеется механизм. Чтобы злонамеренный регистратор не мог существенно задержать начальную загрузку, заявитель **должен** ограничивать Retry-After значением 60 секунд. В идеале заявитель будет отслеживать значения поля Retry-After для любого числа остающихся регистраторов, но это требует наличия у него таблицы состояний. Вместо этого заявитель **может** игнорировать точные значения Retry-After в пользу одного жёстко заданного значения (регистратор, не способный завершить транзакцию за первые 60 секунд, получает другой шанс через 1 минуту). Заявителю **следует** быть готовым поддерживать состояние отклика 202 до 4 дней, что превышает длинные выходные, после чего передаётся отказ в попытке зачисления и заявитель возвращается в состояние Discovery. Это даёт время на передачу отклика оператору (человеку), который может принять решение о продолжении попыток или отказе от зачисления. Заявитель, повторяющий запрос после получения кода 202, **должен** передать тот же voucher-request. **Недопустимо** каждый раз подписывать новый voucher-request, в частности, **недопустимо** менять значение nonce.

Для предотвращения бесконечных циклов перенаправления, которые орнанизует вредоносный регистратор с целью помешать заявителю найти корректного регистратора, заявителю **недопустимо** следовать более, чем одному перенаправлею (код 3xx) на другой web-источник. EST поддерживает перенаправление, но требует ввода от пользователя, это изменение позволяет заявитель использовать 1 перенаправление без участия пользователя.

Отклик 403 (Forbidden - запрещено) подходит при некорректной подписи voucher-request, его устаревании или наличия у заявителя другого ваучера, который не может быть переопределён. Отклик 404 (Not Found - не найдено) подходит при запросе для устройства, неизвестного MASA. Отклик 406 (Not Acceptable - неприемлемо) подходит, если ваучер желаемого типа или использующий желаемый алгоритм (как указано в поле заголовка Accept и алгоритме, примененном в подписи) не может быть выдан, поскольку MASA знает, что заявитель не может обработать этот тип. Регистратору **следует** применять такой отклик, если он видит, что заявитель не приемлем для контроля учёта, MASA audit-log или по иным причина. Отклик 415 (Unsupported Media Type - неподдерживаемый тип носителя) подходит для запроса с непонятным voucher-request или значением Accept.

Формат отклика с ваучером указан в полях представленного заголовка Accept или основывается на предварительном знании агентом MASA подходящего для этого заявителя формата. В настоящее время определён лишь носитель application/voucher-cms+json [RFC8366]. Синтаксис ваучера подробно описан в [RFC8366]. Пример содержимого ваучера представлен на рисунке 14.

```
{
  "ietf-voucher:voucher": {
    "nonce": "62a2e7693d82fcda2624de58fb6722e5",
    "assertion": "logged",
    "pinned-domain-cert": "base64encodedvalue==",
    "serial-number": "JADA123456789"
  }
}
```

Рисунок 14. Пример ваучера.

Агент MASA заполняет поля ваучера, указанные ниже.

nonce
: Значение nonce от заявителя при его доступности (см. 5.5.6. Обработка MASA Nonce).

assertion
: Метод проверки взаимоотношений заявителя и регистратора (см. 5.5.5. Проверка агентом MASA prior-signed-voucher-request от заявителя).

pinned-domain-cert
: Сертификат (см. 5.5.2. Закрепление регистратора агентом MASA). Пример приведён в Приложении C.2. Примеры процессов, где применяется сертификат конечного объекта.

serial-number
: Серийный номер, представленный в voucher-request (см. также 5.5.5. Проверка агентом MASA prior-signed-voucher-request от заявителя).

domain-cert-revocation-checks
: Устанавливается в соответствии с возможностями заявителя и [RFC8366]. MASA **может** установить в этом поле значение false, поскольку установка true потребует доступности сведений об отзыве заявителю, а этот документ не задаёт нормативных требования для интеграции с [RFC6961], параграфом 4.4.2.1 в [RFC8446] или эквивалентами.

expires-on
: Устанавливается для ваучеров без nonce. MASA гарантирует, что срок действия ваучера согласуется с отзывом и проверками согласованности pinned-domain-cert, которые заявитель может выполнять (см. параграф 2.6.1). Нужно учитывать три момента: (a) настроенный срок действия ваучера в MASA, (b) срок завершения сертификата регистратора, (c) срок действия CRL. Значению поля expires-on **следует** быть раньше каждого из этих значений. Обычно (b) указывает достаточно далеко в будущее, но значение (c) обычно невелико (неделя или меньше). **Рекомендуемый** период для (a) составляет около 20 минут, поэтому он обычно определяет срок действия полученного ваучера. 20 минут достаточно для достижения заявителем состояния post-provisional, когда между ним и регистратором уже установлены отношения доверия. Последующие операции могут продолжать столько, сколько им требуется. Срок действия ваучера не влияет на срок действия отношений владения.

При каждой выдаче ваучера агент MASA должен обновлять журнал аудита, чтобы можно было создать отклик, как описано в параграфе 5.8.1. Требования к внутреннему состоянию для поддержки audit-log выходят за рамки документа.

5.6.1. Проверка ваучера заявителем

Заявитель **должен** проверять подпись ваучера, используя заданные изготовителем привязки доверия, связанные с MASA изготовителя (вероятно, они включены в прошивку заявителя). Управления заданными изготовителем привязками доверия выходит за рамки этого документа, а протокол не обновляет такие привязки.

Заявитель должен убедиться, что поле serial-number в ваучере соответствует серийному номеру заявителя.

Заявитель должен проверить данные nonce в ваучере. При наличии nonce в ваучере значение должно совпадать с nonce, представленным регистратору. Приемлемы ваучеры без nonce (в зависимости от локальной политики, см. 7.2. Снижение защиты заявителем).

Заявитель **должен** быть готов к синтаксическому анализу и отказу от отклика с ваучером, не содержащим поля pinned-domain-cert. Это указывает отказ при зачислении в домен и заявитель **должен** попытаться присоединиться через другой доступный Join Proxy.

Заявитель **должен** быть готов игнорировать непонятные ему дополнительные поля.

5.6.2. Аутентификация заявителем представленного соединения TLS

В соответствии с процессом из [RFC8366] заявителю следует считать открытый ключ из pinned-domain-cert единственной временной привязкой доверия. Затем заявитель оценивает цепочку сертификатов сервера TLS, полученную при организации соединения TLS с использованием этой привязки доверия. Возможно совпадение открытого ключа из pinned-domain-cert с открытым ключом в сертификате конечного объекта, представленном сервером TLS.

Если свидетельства регистратора не удаётся проверить с использованием привязки доверия pinned-domain-cert из ваучера, соединение TLS отбрасывается и заявитель отказывается от попыток начальной загрузки с найденным регистратором. Заявителю **следует** передать телеметрию статуса ваучера (см. ниже) до закрытия соединения TLS. Хаявитель **должен** попытаться зарегистрироваться с использованием других найденных посредников. Ему следует вернуться к тому же посреднику после неудачных попыток с другими proxy. Попытки **следует** предпринимать с использованием описанного выше таймера отсрочки. Попытки **следует** повторять, поскольку причиной отказа может быть временная несогласованность (несогласованный ключ регистратора или другая ошибка в конфигурации). Несогласованность может быть также результатом активной MITM-атаки на соединение EST.

Регистратор **должен** использовать сертификат, связанный с pinned-domain-cert, в качестве сертификата сервера TLS.

Проверка заявителем пути PKIX для срока действия сертификата регистратора описана в параграфе 2.6.1. После успешной проверки пути PKIX соединение TLS перестаёт быть временным.

Сертификат pinned-domain-cert **может** устанавливаться как привязка доверия для будущих операций, таких как зачисление (например, как рекомендовано в [RFC7030]), управление привязками доверия или raw-протоколы, которым не нужно полное управление ключами на основе PKI. Его можно применять для аутентификации любого динамически найденного сервера EST с расширением использования ключей id-kp-cmcRA, заданным в EST (параграф 3.6.1 в [RFC7030]), но для упрощения заявителю **следует** избегать дополнительных операций обнаружения. Взамен заявителю **следует** напрямую взаимодействовать с регистратором в качестве сервера EST. сертификат pinned-domain-cert не является откликом полного распространения сертификата CA, как описано в параграфе 4.1.3 [RFC7030], что служит дополнительным основанием продолжить операции управления ключами EST. После получения полного отклика с сертификатом CA этот сертификат будет более полномочным для домена, нежели ограниченный отклик pinned-domain-cert.

5.7. Телеметрия статуса заявителя BRSKI

Предполагается, что домен предоставит системным администраторам индикацию состояния жизненного цикла устройства. Для этого нужны сведения телеметрии статуса устройства.

Заявитель **должен** указывать свой статус в отношении ваучера, передавая регистратору сообщение о состоянии с использованием носителя типа application/json

Клиент передаёт серверу HTTP POST с URI .well-known/brski/voucher_status. Формат и семантика описаны ниже для версии 1. Поле версии включено для возможности внесения в будущем существенных изменений. Регистратору, получившему сообщение о состоянии с номером версии больше известного ему, **следует** сделать запись в журнале и подать сигнал человеку. Поле статуса указывает, приемлем ли ваучер. Допустимы логические значения, true указывает, что ваучер был воспринят.

Если ваучер не подходит, строка Reason указывает причину этого. В случае отказа сообщение может передаваться неаутентифицированному и, возможно, враждебному регистратору, поэтому в строку Reason **не следует** включать сведения, которые могут дать преимущества злоумышленнику. Эксплуатационная польза от таких сообщений уравновешивается операционными издержками, связанными с тем, что не вносится запись об игнорировании ваучера клиентом, который по мнению регистратора планировал продолжить присоединение к домену.

Атрибут reason-context представляет собой произвольный объект JSON (буквальное значение или хэш) с дополнительными сведениями, относящимися к этому заявителю. Содержимое поля не стандартизуется.

Поля version и status **должны** присутствовать. Поле Reason **следует** включать, когда поле состояния имеет значение false. Поле Reason-Context является необязательным. При успехе (SUCCESS) поле Reason **можно** не включать.

В ключах для этого объекта JSON учитывается регистр символов и они **должны** использовать нижний регистр. На рисунке 16 представлен пример JSON.

```
<CODE BEGINS> file "voucherstatus.cddl"
voucherstatus-post = {
    "version": uint,
    "status": bool,
    ? "reason": text,
    ? "reason-context" : { $$arbitrary-map }
  }
}
<CODE ENDS>
```

Рисунок 15. CDDL для Voucher Status POST.

```
{
    "version": 1,
    "status":false,
    "reason":"Informative human-readable message",
    "reason-context": { "additional" : "JSON" }
}
```

Рисунок 16. Пример Status Telemetry.

Серверу **следует** отвечать с кодом HTTP 200, но **можно** просто отказать с кодом HTTP 404. Клиент игнорирует отклик. Серверу **следует** записывать данные телеметрии в системные журналы.

В запрос POST **могут** быть добавлены другие стандартные поля JSON (см. 8.5. Телеметрия состояния BRSKI у заявителя). Серверу, встретившему с неизвестное полем, следует записать это в журнал, игнорируя поле в остальном.

5.8. Запрос журнала аудита у регистратора

После получения телеметрии статуса заявителя (параграф 5.7) регистратору **следует** запросить MASA audit-log у сервера MASA с помощью HTTP POST с путём к операции /.well-known/brski/requestauditlog.

Регистратору **следует** включать в HTTP POST свой voucher-request, использованный при запросе ваучера (с тем же Content-Type) и передавать его с URI /requestauditlog. Поля idevid-issuer и serial-number указывают MASA запрашиваемый журнал, чтобы он мог быть подготовлен для отклика. Использование того же типа носителя и сообщения минимизирует операции с криптографией и сообщением, хотя создаёт дополнительный сетевой трафик. Надёжная реализация MASA **может** использовать внутреннее состояние для связывания этого запроса с исходным, который уже проверен, чтобы избежать дополнительной криптопроверки.

Регистратор **может** запросить журнал в будущем. Если регистратор генерирует новый запрос, MASA придётся выполнить дополнительные криптографические операции для его проверки.

Агент MASA, получивший запрос для несуществующего устройства или для которого запрашивающий владелец никогда не был таковым, возвращает код HTTP 404 (Not found).

С точки зрения регистратора разумно считать, что MASA не считает себя текущим владельцем, чтобы запросить журнал аудита. Для этого могут быть причины, которые трудно знать заранее. Например, такой регистратор может не знать, что устройство было перепродано, возможно устройство перепродано неправомерно и именно так прежний владелец узнает о случившемся. Возможно также легитимное использование устройства в двух разных сетях.

Вместо возврата журнала аудита в ответ на POST (с кодом 200) MASA **может** вернуть отклик 201 (Created - создано) ([RFC7231], параграф 6.3.2 и раздел 7.1) и URL для подготовленного (идемпотентного, поэтому кэшируемого) отклика с аудитом в поле заголовка Location.

Чтобы избежать перечисления журналов аудита устройств, агенту MASA, возвращающему URL, **следует** позаботиться о том, чтобы возвращаемый URL нельзя было угадать. В [W3C.capability-urls] приведены очень хорошие рекомендации для этого. Например, вместо возврата URL с номером базы данных, вроде https://example.com/auditlog/1234, или расширенным уникальным идентификатором (Extended Unique Identifier или EUI) устройства, таким как

https://example.com/auditlog/10-00-00-11-22-33, MASA **следует** возвращать случайно сгенерированное значение (slug в терминологии web). Это значение сложит для нахождения соответствующей записи базы данных. Агент MASA, возвращающий код 200, **может** включать заголовок Location для дальнейшего использования регистратором.

5.8.1. Отклик MASA Audit-Log

Возвращается файл данных из журнала, состоящий из всех записей, связанных с устройством, заданным значением IDevID в запросе. Журнал аудита можно сократить путём удаления старых или повторяющихся записей, как указано ниже. Возвращаемые данные представлены в формате JSON [RFC8259], для Content-Type **следует** применять application/json.

Приведённый ниже код CDDL [RFC8610] показывает структуру формата JSON с откликом из журнала аудита.

```
<CODE BEGINS> file "auditlog.cddl"
audit-log-response = {
  "version": uint,
  "events": [ + event ]
  "truncation": {
    ? "nonced duplicates": uint,
    ? "nonceless duplicates": uint,
    ? "arbitrary": uint,
  }
}

event = {
  "date": text,
  "domainID": text,
  "nonce": text / null,
  "assertion": "verified" / "logged" / "proximity",
  ? "truncated": uint,
}
<CODE ENDS>
```

Рисунок 17. CDDL для отклика Audit-Log.

Ниже представлен пример отклика.

```
{
  "version":"1",
  "events":[
    {
        "date":"2019-05-15T17:25:55.644-04:00",
        "domainID":"BduJhdHPpfhQLyponf48JzXSGZ8=",
        "nonce":"VOUFT-WwrEv0NuAQEHoV7Q",
        "assertion":"proximity",
        "truncated":"0"
    },
    {
        "date":"2017-05-15T17:25:55.644-04:00",
        "domainID":"BduJhdHPpfhQLyponf48JzXSGZ8=",
        "nonce":"f4G6Vi1t8nKo/FieCVgpBg==",
        "assertion":"proximity"
    }
  ],
    "truncation": {
        "nonced duplicates": "0",
        "nonceless duplicates": "1",
        "arbitrary": "2"
     }
}
```

Рисунок 18. Пример отклика Audit-Log.

Поле domainID указывает двоичное значение SubjectKeyIdentifier, рассчитанное в соответствии с параграфом 5.8.2. Оно однократно кодируется в формате base64 для транспортировки в контейнере JSON. Значение даты (date) указывается в формате [RFC3339], соответствующем типичному использования JSON в JavaScript. Структуру отсечки **можно** не включать, если все значения равны 0. Предполагается, что все пропущенные значения счётчиков в этой структуре имеют значение 0. Значение nonce является строкой, представленной в voucher-request, и применяется в ваучере. Если в результирующем ваучере нет nonce, **следует** использовать значение null, а не пропускать строку. Хотя nonce часто создаётся как последовательность случайных байтов в кодировке base64, этого не следует предполагать.

Распространение большого журнала далеко не идеально. Структуру можно оптимизировать - записи с nonce или без nonce для одного domainID **можно** сократить, оставляя лишь самую свежую запись с nonce или без nonce для этого domainID. В случае отсечки значению event в ней **следует** указывать число пропущенных событий для данного domainID. **Не следует** дополнительно сокращать журнал, но могут быть рабочие ситуации, где поддержка полного журнала невозможна. В таких случаях размер журнала **можно** сократить произвольно, а для числа удалённых записей указать значение arbitrary (произвольное). Если счётчик отсечки превышает 1024, MASA **может** использовать это значение без дальнейшего инкрементирования.

Журнал, где дубликаты записей для одного домена были опущены (nonced duplicates и/или nonceless duplicates), может оставаться приемлемым для обоснованных решений. Журнал с произвольной (arbitrary) отсечкой менее приемлем, но лучше указать изготовителя, чем спрятать отсечки.

Регистратор, который видит значение версии больше 1, указывает формат журнала, который может содержать дополнительные сведения. Никакая информация не будет удаляться в новых версиях, а при несовместимых изменениях будет использоваться новая конечная точка.

Этот документ задаёт простой формат журнала, предоставляемого службой MASA регистратору, который можно улучшить с помощью технологий распределенного согласования (consensus), объединяющих ваучеры с такими технологиями, как block-chain, хэш-деревья, оптимизированное ведение журналов. Это вопрос выходит за рамки документа, но остаётся желаемым для будущей работы. Поэтому регистраторам **следует** предвидеть новые типы откликов и **следует** давать операторам элементы управления для указания способов обработки неизвестных откликов.

5.8.2. Расчёт domainID

Поле domainID - это двоичное значение (BIT STRING), однозначно указывающие регистратор по pinned-domain-cert. Если сертификат pinned-domain-cert включает SubjectKeyIdentifier (параграф 4.2.1.2 в [RFC5280]), это значение используется в качестве domainID, в противном случае применяется SPKI Fingerprint, как указано в параграфе 2.4 [RFC7469]. Это значение должны рассчитывать как MASA (для audit-log), так и регистратор (для распознавания себя в audit-log). Параграф 4.2.1.2 [RFC5280] не требует присутствия расширения SubjectKeyIdentifier в сертификатах не от CA. В (самоподписанный) сертификат регистратора **рекомендуется** всегда включать SubjectKeyIdentifier для использования в качестве domainID.

Значение domainID определяется из цепочки сертификатов, связанной с pinned-domain-cert, и применяется для обновления audit-log.

5.8.3. Проверка журнала аудита у регистратора

При каждом выпуске ваучера MASA добавляет в конце сведения о назначении во внутренний журнал аудита для этого устройства, который обрабатывается при откликах на запросы, как описано в параграфе 5.8. Содержимое журнала может указывать разные уровни доверия и в этом параграфе описано, какое доверие может получить регистратор из записи.

Хотя audit-log предоставляет список выпущенных MASA ваучеров, они выдаются в ответ на voucher-requests и именно содержимое voucher-request определяет значимость записей в журнале аудита.

Регистратору **следует** использовать сведения из журнала для принятия обоснованного решения вопроса о продолжении начальной загрузки заявителя. Точные правила этого выходят за рамки документа, поскольку они зависят от требований безопасности в домене регистратора. Можно предполагать, что у купленного оборудования уже были другие владельцы и ниже приведены сведения, которые могут помочь разобраться с записями журнала.

date
: Поле даты позволяет регистратору выделять в журнале записи по известным событиям, таким как дата покупки. В зависимости от доступного регистратору или администратору контекста, события до или после определённой даты может существенно различаться по важности. Например, для считающегося новым (без истории) оборудования будет сюрпризом наличие предшествующих событий.

domainID
: Если журнал включает неожиданное значение domainID, заявитель может быть «впечатан» в другой домен. Можно предполагать у регистратора наличие различных методов обнаружения «неожиданностей» от списков прежних доменов до обнаружения аномалий (например, устройство было привязано к домену, отличающемуся от домена других развёрнутых устройств). Записи можно также сравнивать с локальными журналами истории для поиска несоответствий (например, это устройство уже было внедрено внутри несколько раз, но журнал аудита показывает внешнее развёртывание, о котором во внутренних журналах нет сведений).

nonce
: Записи без nonce говорят, что указанный в журнале domainID теоретически может инициировать сброс заявителя, а затем принять на себя управления, используя имеющийся ваучер без nonce.

assertion
: Лист подтверждения в ваучере и журнале аудита указывает причину выдачи ваучера агентом MASA. Запись verified (подтверждено) указывает, что агент MASA выдал ваучер по положительному результату проверки владения. Однако эта запись не показывает, был ли заявитель развернут в прежнем домене. Запись logged (зарегистрировано) говорит регистратору, что прежние ваучеры были выданы с минимальной проверкой. Запись proximity (близость) гарантирует регистратору, что заявитель действительно взаимодействовал с прежним доменом, что подтверждает внедрение заявителя в прежнем домене.

Относительно простая политика состоит в том, чтобы принять список известных (внутренних или внешних) domainID и требовать наличия nonce во всех ваучерах. Другой вариант требует, чтобы все ваучеры без nonce относились в некому подмножеству domainID (например, только внутренние). При нарушении правил достаточно просто отозвать все локально выданные свидетельства для соответствующего заявителя или отклонить пересылку ваучера. После этого регистратор **должен** отклонять любые действия EST и **следует** также уведомить человека через журнал. Регистратор **можно** настроить на игнорирование (т. е. переопределение указанного выше правила) истории устройства, но **рекомендуется** делать это лишь при наличии аппаратной поддержки, т. е. модуля доверенной платформы (Trusted Platform Module[1] или TPM) оценки конечных точек сети (Network Endpoint Assessment или NEA) [RFC5209].

5.9. Интеграция EST с начальной загрузкой

Заявителю **следует** выполнить операции BRSKI с операциями зачисления EST, включая запросы сертификата CA, атрибутов CSR, сертификата клиента, генерации ключей на сервере и т. п. Это сравнительно бесшовная интеграция, поскольку вызовы BRSKI API обеспечивают автоматизированный вариант начальной загрузки, описанной в [RFC7030]. Как отмечено выше, использование сохраняющегося соединения HTTP упрощает конечный автомат заявителя.

Хотя EST позволяет клиентам получать несколько сертификатов, отправляя запросы на подписание сертификата (Certificate Signing Request или CSR), BRSKI не поддерживает этот механизм напрямую. Это связано с тем, что заявители BRSKI **должны** использовать запрос атрибутов CSR (параграф 4.5 в [RFC7030]). Регистратор **должен** проверить CSR на соответствие ожидаемым атрибутам. Это означает, что запросы клиента «будут выглядеть одинаково» и, следовательно, приведут к выпуску одного логического сертификата даже при нескольких запросах от клиента. Регистраторы **могут** применять более сложную логику, но это выходит за рамки документа. BRSKI не указывает каких либо расширений или ограничений для этой возможности.

[1] В оригинале ошибочно сказано Transport Performance Metrics. См. https://www.rfc-editor.org/errata/eid6716. Прим. перев.

5.9.1. Распространение EST для сертификатов CA

Заявителю **следует** запрашивать полное распространение EST для сообщений с сертификатами CA, см. параграф 4.1 в [RFC7030]. Это обеспечивает наличие у заявителя полного набора текущих сертификатов CA в дополнение к pinned-domain-cert (ограничения, присущие использованию одного сертификата вместо полного отклика с сертификатами CA, рассмотрены в параграфе 5.6.2). Хотя эти ограничения допустимы при исходной начальной загрузке, они не подходят для текущей проверки сертификата конечного объекта PKIX.

5.9.2. Атрибуты CSR для EST

Автоматизированная начальная загрузка происходит без настройки заявителя локальным администратором. В некоторых развёртываниях вполне вероятно создание заявителем запроса, содержащего лишь известное ему отождествление (по сути, X.509 IDevID) и получение в конечном итоге сертификата со специфическими для домена сведениями об отождествлении. Концептуально CA имеет полный контроль над всеми полями, выдаваемыми в сертификате конечного объекта, но на деле это сложно с точки зрения текущего состояния внедрения PKI CA, где CSR представляется в CA через ряд нестандартных протоколов. Даже при использовании стандартизованных протоколов может быть проблематично ожидать, что зависящие от службы поля сертификата могут быть созданы CA, который вероятно управляется группой, не имеющей представления об используемых сетевых протоколах и службах. Например, CA может поддерживаться сторонней организацией.

Для снижения этих эксплуатационных трудностей заявитель **должен** запросить атрибуты CSR для EST у сервера EST, а тот должен иметь возможность ответить атрибутами, требуемыми для использования сертификата в предполагаемых протоколах (службах). Такой подход допускает минимальную интеграцию с CA, а локальная инфраструктура (сервер EST) сообщает заявителю нужные поля для включения в генерируемый запрос CSR (такие как rfc822Name). Это позволяет автоматизировать начальную загрузку в самых разных средах.

В средах, использующих зачисленный BRSKI сертификат для аутентификации ACP, атрибуты CSR для EST **должны** включать поля сведений о домена ACP, заданные в параграфе 6.2.2 [RFC8994].

Регистратор **должен** также подтвердить, что формат результирующего CSR соответствует указанному, до отправки запроса CA. Если регистратор взаимодействует с CA по протоколу, такому как управление сертификатами через CMS (Certificate Management over CMS или CMC), который предоставляет механизмы переопределения атрибутов CSR, эти механизмы **могут** использоваться, если клиент игнорирует рекомендации для атрибутов CSR.

5.9.3. Запрос сертификата клиента EST

Заявитель **должен** запрашивать новый сертификат клиента (см. параграф 4.2 в [RFC7030]).

5.9.4. Телеметрия статуса зачисления

Для автоматизированной начальной загрузки устройств административные элементы, обеспечивающие начальную загрузку, предоставляют системным администраторам индикацию о статусе жизненного цикла устройства. Это может включать сведения о попытках начальной загрузки, видимые клиенту. MASA предоставляет журналы и статус зачисления свидетельств. Поскольку в соответствии с [RFC7030] предполагается конечный пользователь, финальная индикация успеха на включается. Этого недостаточно для автоматизированных вариантов применения.

Клиент **должен** передать регистратору сведения о статусе своего зачисления. Это делается с использованием в HTTP POST словаря JSON с атрибутами, описанными ниже для новой конечной точки EST, по ссылке /.well-known/brski/enrollstatus.

При индикации успешного зачисления клиенту **следует** сначала заново организовать сессию EST TLS, используя вновь полученные свидетельства. TLS 1.3 поддерживает это по основному каналы (in-band), а TLS 1.2 не поддерживает. Поэтому клиенту **следует** закрыть имеющееся соединение TLS и начать новое с тем же Join Proxy. При отказе в зачислении клиент **должен** передать сведения телеметрии через соединение TLS, которое применялось при попытке зачисления, со строкой Reason, указывающей причину отказа при последней попытке зачисления (для таких случаев соединение TLS является наиболее надёжным способом сопоставить сведения сервера с клиентскими).

Поля version и status **должны** присутствовать, поле Reason **следует** включать при значении false в поле status. При успехе (SUCCESS) поле Reason **можно** не включать.

Атрибут reason-context является произвольным объектом JSON (буквальное значение или хэш) с дополнительными сведениями об отказе при зачислении этого заявителя. Содержимое этого поля не стандартизуется. Это представлено групповым сокетом $$arbitrary-map в CDDL.

```
<CODE BEGINS> file "enrollstatus.cddl"
enrollstatus-post = {
    "version": uint,
    "status": bool,
    ? "reason": text,
    ? "reason-context" : { $$arbitrary-map }
  }
}
<CODE ENDS>
```

Рисунок 19. CDDL для Enrollment Status POST

An example status report can be seen below. It is sent with the media type: application/json

```
{
    "version": 1,
    "status":true,
    "reason":"Informative human readable message",
    "reason-context": { "additional" : "JSON" }
}
```

Рисунок 20. Пример Enrollment Status POST.

Серверу **следует** отвечать кодом HTTP 200, но **можно** просто отказать с кодом HTTP 404. В своих журналах сервер **должен** оставить запись, если это сообщение было получено в сессии TLS с соответствующим сертификатом клиента.

5.9.5. Множество сертификатов

Заявители, которым нужно несколько сертификатов, могут организовать прямые соединения EST с регистратором.

5.9.6. EST через CoAP

Этот документ описывает расширение EST для инфраструктуры удалённых ключей при начальной загрузке. Начальная загрузка актуальная также для обсуждения зачисления CoAP. Определение EST и BRSKI через CoAP не рассматривается в этом документе, кроме обеспечения proxy-поддержки для операций CoAP. Предполагается, что определение сопоставлений CoAP будет дано в последующих документах, таких как [ACE-COAP-EST], а отображения CoAP для BRSKI будут описана там же или в будущих работах.

6. Разъяснение по транспортному кодированию

[RFC7030] определяет конечные точки для включения заголовка Content-Transfer-Encoding, а для данных (payload) используется представление DER с кодированием base64 [RFC4648].

При использовании с BRSKI исходные конечные точки EST сохраняют кодирование base64 [RFC7030] (как разъяснено в [RFC8951]), но новые конечные точки BRSKI, передающие и принимающие двоичные артефакты (в частности, /.well-known/brski/requestvoucher), остаются бинарными, т. е. кодирование не применяется.

В контексте BRSKI поле заголовка EST Content-Transfer-Encoding следует игнорировать, если оно имеется. Включать это поле не требуется.

7. Режимы работы с пониженной защитой

Общим требованием начальной загрузки является поддержка менее защищённых режимов работы для конкретных случаев. В этом разделе предлагается ряд механизмов, меняющих гарантии защиты BRSKI для разных архитектур внедрения и смягчения проблем управления жизненным циклом, отмеченных в разделе 10. Они представлены здесь в качестве ненормативного руководства для будущей стандартизации. А разделе 9 приведены заявления о применимости стандартизации для ANIMA ACP. Другие пользователи могут ожидать, что подмножества этих механизмов могут быть профилированы в заявлениях о применимости, подобных приведённым в разделе 9.

Этот раздел считается ненормативным в рамках протокола. Использование предложенных здесь механизмов **должно** быть детализовано в конкретных профилях BRSKI, как в разделе 9. Применимость к ACP.

7.1. Модель доверия

В этом параграфе объяснены отношения доверия, подробно описанные в параграфе 2.4. Поток протокола.

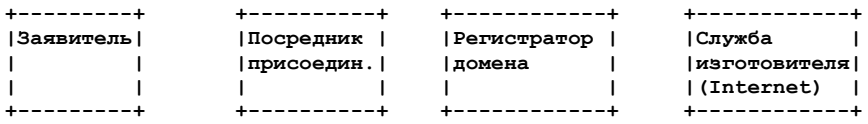

Рисунок 21. Элементы модели доверия BRSKI.

Pledge - заявитель

Заявитель может быть скомпрометирован или на него может быть нацелена атака вредоносного ПО. Объект считается доверенным лишь для впечатывания (imprint) с использованием защищённых методов, описанных в этом документе. Рекомендуются дополнительные методы оценки конечных точек, но это выходит за рамки документа.

Join Proxy - посредник присоединения

Обеспечивает функции посредника, но не участвует в вопросах безопасности.

Registrar - регистратор

Регистратор принимает все решения при взаимодействии с MASA. Для ваучеров аудита владения (Ownership Audit Voucher, [RFC8366] регистратору предоставляется возможность воспринимать решения MASA.

Vendor Service, MASA - служба производителя, агент MASA

Этой форме службы изготовителя доверяют аккуратное ведение журнала всех попыток подачи заявлений и предоставление регистраторам достоверных сведений из журнала. MASA не знает о связи устройств с доменами. Заявления могут дополнительно защищаться криптографическими методами ведения журнала, обеспечивающими возможность лишь дописывания в журнал, криптографическое подтверждение, и открытую проверку.

Vendor Service, Ownership Validation - служба изготовителя, проверка владения

Этой форме службы изготовителя доверяют точные сведения о принадлежности устройств к доменам.

7.2. Снижение защиты заявителем

Ниже приведён список вариантов поведения, которые можно запрограммировать для заявителя. Эти варианты не исключают друг друга и не зависят один от другого. Некоторые из этих вариантов разрешают автономное и аварийное (с привлечением оператора) развёртывание. Используется язык требования, поскольку эти варианты поведения упоминаются в последующих разделах.

1. Заявитель **должен** воспринимать ваучеры без nonce. Это позволяет использовать вариант, где регистратор не может связаться с MASA во время развёртывания. Ведение журнала и сроки действия учитывают соображения безопасности, связанные с поддержкой этих вариантов.

2. Многие устройства уже поддерживают доверие при первом использовании для физических интерфейсов, такие как консольные порты. Данный документ не меняет этого. Устройствам, поддерживающим этот протокол, **недопустимо** поддерживать доверие при первом использовании на сетевых интерфейсах. Это обусловлено тем, что доверие при первом использовании на сетевом интерфейсе подорвёт средства защиты, основанные на записи в журнал, предусмотренной этой спецификацией.

3. У заявителя **может** быть режим работы, в котором он пропускает проверку ваучера один раз, например, при нажатии (специальной) кнопки в процессе начальной загрузки. Это может быть полезно при недоступности службы изготовителя. Такое поведение **следует** делать доступным через локальную конфигурацию или методы физического присутствия (такие, как использование последовательной консоли), чтобы обеспечить возможность внедрения новых объектов даже при отказе автономных методов. Это позволяет получить незащищённый отпечаток (imprint).

4. Последовательная консоль может включать такие команды, как est-enroll [2001:db8:0:1]:443, которые запускают процесс EST с момента после проверки ваучера. Этому процессу **следует** включать проверку сертификата сервера с использованием экранного отпечатка (on-screen fingerprint).

Рекомендуется разрешать доверие при первом использовании или иной метод пропуска проверки ваучера (включая использование последовательной консоли) лишь при поддержке аппаратной оценки конечных точек (NEA) [RFC5209]. Эта рекомендация гарантирует, что мониторинг сети домена сможет обнаружить использованием процедур автономного или аварийного развёртывания, когда основанная на ваучерах начальная загрузка не применяется.

7.3. Снижение защиты регистратором

Регистратор может воспринимать устройства, используя менее защищённые методы, но **недопустимо** разрешать это по умолчанию. Такие методы могут быть приемлемы в ситуациях, когда модель угроз указывает, что снижение защиты допустимо. Это включает ситуации, где решения по безопасности принимает локальны администратор.

1. Регистратор **может** воспринимать все устройства или устройства определённого типа. Администратор может делать такой выбор в случаях, когда сложно настроить регистратор на работу с индивидуальными идентификаторами, ожидаемыми от каждого нового устройства.

2. Регистратор **может** воспринимать устройства, которые заявили уникальный идентификатор без проверки подлинности заявленного отождествления. Это может происходить, когда у заявителя нет установленного на производстве свидетельства X.509 IDevID. Новые объекты без свидетельства X.509 IDevID **могут** формировать запрос в соответствии с параграфом 5.2, используя формат из параграфа 5.5, чтобы убедиться, что сведения о серийном номере представлены регистратору (включая значение IDevID AuthorityKeyIdentifier, которое статически настраивается у заявителя). Заявитель **может** отказаться от предъявления клиентского сертификата TLS (поскольку тот недоступен). Заявителю **следует** поддерживать аутентификацию на основе HTTP или TLS без сертификата, как описано в параграфе 3.3.2 EST [RFC7030]. Регистратору **недопустимо** воспринимать новые неаутентифицированные объекты, пока это на задал администратор, который подтвердил, что лишь ожидаемые проверенные объекты могут взаимодействовать с регистратором (предположительно через физически защищённый периметр).

3. Регистратор **может** представить службе MASA voucher-request без nonce. Полученные ваучеры могут сохраняться регистратором, пока они нужны для операций начальной загрузке. Это применяется в случаях, когда целевая сеть защищена воздушным зазором (изолирована) и поэтому не может взаимодействовать со службой MASA в процессе развёртывания заявителя.

4. Регистратор **может** игнорировать нераспознанные записи журнала без nonce. Это может происходить при использовании оборудования, купленного с действительной историей развёртывания в изолированных сетях, которые требуют автономных (offline) ваучеров.

5. Регистратор **может** воспринимать форматы ваучеров будущих типов, которые он не может анализировать. В таких случаях регистратор не может увидеть всего содержимого ваучера, но не меняет операций протокола.

7.4. Снижение защиты MASA

Режимы пониженной защиты, выбранные службой MASA, влияют на развёртывание всех устройств, если пониженный уровень защиты не связан с отождествлениями конкретных устройств. Описанные ниже режимы могут применяться к конкретным устройствам на основе сведений об их продаже. Они могут также связываться с конкретными клиентами (независимо от отождествления устройства) аутентифицирующим регистратором клиента.

7.4.1. Выпуск ваучеров без nonce

MASA может не включать nonce в ваучер и/или не требовать его наличия в voucher-request. Это ведёт к распространению ваучеров, которые могут иметь неограниченный срок действия, и по сути делает указанный домен доверенным объектом для заявителя при всех последующих попытках начальной загрузки. Журнальные записи фиксируют выпуск ваучеров без nonce, что позволяет регистратору принять соответствующее решение по защите при зачислении заявителя в домен. Ваучеры без nonce полезны для поддержки вариантов использования, где регистратор может быть недоступен в процессе фактической начальной загрузки.

Хотя ваучер без nonce может включать дату завершения срока действия, типичное применение таких ваучеров предполагает их долгосрочными. Если можно быть уверенным в точности часов устройства (MASA будет знать), тогда ваучер без nonce **можно** выпускать с ограниченным сроком действия.

Более типичным для ваучера без nonce является использование при автономном (offline) включении, когда невозможно передать новый voucher-request агенту MASA. Использование долгосрочного ваучера также избавляет от необходимости заботиться о доступности MASA на много дней в будущее. Поэтому ваучеры без nonce не имеют срока действия.

Долгосрочный ваучер не требует подтверждения наличия устройства в сети. Выдача такого ваучера допускается лишь случаях, когда регистратор аутентифицирован агентом MASA, а тот уполномочен предоставлять свои функции данному клиенту. Агенту MASA **рекомендуется** использовать эту функциональность лишь в сочетании с расширенным уровнем отслеживания владения, детали которого выходят за рамки этого документа.

Если известно, что устройство заявителя имеет часы, установленные на заводе, **рекомендуется** использовать срок действия ваучера.

7.4.2. Доверие при первом использовании

MASA может не проверять владение перед отправкой отклика с ваучером. Предполагается, что это будет обычная рабочая модель, поскольку это позволяет изготовителям, предоставляющим услуги MASA, обойтись без отслеживания владения в процессе поставки и через цепочки поставок, а также обойтись очень малыми издержками, связанными с услугами MASA. Регистратор использует сведения из журнала аудита как стратегию всесторонней защиты, чтобы гарантировать, что это не произойдёт неожиданно (например, при покупке нового оборудования регистратор выдаст ошибку, если не будут получены сведения из журнала аудита). Агенту MASA **следует** проверять сведения prior-signed-voucher-request для заявителей, которые поддерживают такую функциональность. Это обеспечивает проверку подтверждения близости, которая снижает потребность в проверке владения. Подтверждение близости основано на допущении размещения заявителя и посредника присоединения на одном локальном канале.

Агент MASA, использующий TOFU для отождествления регистратора, может захотеть аннотировать источник соединения по адресу IP или сетевому блоку (netblock) и ограничить будущее использование этого отождествления из других мест. Агенту MASA, который делает это, **следует** позаботиться о том, чтобы не создавать для себя неприятные ситуации, когда клиент имеет несколько регистраторов или использует исходящие соединения IPv4-to-IPv4 NAT (NAT44), которые часто меняются.

7.4.3. Обновление или расширение точек доверия ваучера

В этом параграфе рассматриваются две проблемы - недоступность MASA из-за проблем в бизнесе и отказ MASA взаимодействовать со вторичной продажей.

Изготовитель может предложить механизм управления, позволяющий расширить список привязок доверия для проверки ваучеров. В [YANG-KEYSTORE] описан интерфейс, который можно реализовать с использованием YANG. Практически любой механизм настройки, используемый сегодня, можно расширить для обеспечения требуемого обновления. Производитель может даже установить привязки доверия доменного CA, полученные на этапе EST cacerts в качестве привязок для проверки сертификатов. Потребуются некоторые дополнительные сигналы для чёткой идентификации ключей с полномочиями проверки ваучеров среди подписанных доменным CA. Это дело будущего.

При таком изменении списка привязок доверия ваучеры может выпускать другой агент MASA, которым может быть предыдущий владелец (продавец) или иное доверенное лицо, посредничающее при продаже. Если это третья сторона, продавец должен предпринять действия по внедрению конфигурации этой стороны на устройство до отсоединения. Третья сторона (например, оптовый продавец использованного оборудования) может использовать механизм, описанный в параграфе 7.2, для получения контроля над устройством после его физического получения. Это позволит третьей стороне действовать как MASA при будущих подключениях (onboarding). Сертификат IDevID заменить нельзя, поэтому регистратор нового владельца должен будет поддерживать переопределение MASA URL.

Это будет полезно при перепродаже или другой смене владельца в одной из двух указанных ситуаций. В самом простом варианте устройство не сбрасывается к принятым по умолчанию (заводским) настройкам перед новым подключением. Для инициирования устройства нужен некий защищённый физический сигнал. Это лучше всего подходит для повторного развёртывания устройства на том же предприятии. Это влечёт сохранение в системе предыдущей конфигурации, пока новый владелец не заменит её полностью, поэтому возникает некоторый риск.

В другом случае потребуется два уровня сброса к заводским установкам. Один уровень полностью восстанавливает заводское состояние, включая удаление добавленных привязок доверия, а другой (более частый) просто восстанавливает конфигурацию до известного состояния, принятого по умолчанию, без удаления привязок доверия. Этот более мягкий сброс может сохранить на устройстве значимые свидетельства, что может быть неприемлемо для некоторых владельцев.

В качестве третьего варианта привязки доверия от изготовителя могут быть полностью заменены локальными и возврат к заводским установкам не восстановит этих привязок. Этот вариант даёт большие возможности, но требует большей ответственности - при потере доступа к приватной части новых привязок производитель не сможет помочь.

8. Взаимодействие с IANA

В соответствии с этим документом агентство IANA выполнило указанные ниже действия.

8.1. Реестр IETF XML

Этот документ вносит URI в реестр IETF XML Registry [RFC3688].

```
URI:  urn:ietf:params:xml:ns:yang:ietf-voucher-request
Registrant Contact:  The ANIMA WG of the IETF.
XML:  запрошенный URI является пространством имён XML.
```

8.2. Реестр YANG Module Names

Этот документ регистрирует модуль YANG в реестре YANG Module Names [RFC6020].

```
Name:  ietf-voucher-request
Namespace:  urn:ietf:params:xml:ns:yang:ietf-voucher-request
Prefix:  vch
Reference:  RFC 8995
```

8.3. Общеизвестные BRSKI

8.3.1. Регистрация BRSKI .well-known

Этот документ регистрирует в реестре Well-Known URIs (https://www.iana.org/assignments/well-known-uris/) имя brski с шаблоном из [RFC8615]

```
URI Suffix:  brski
Change Controller:  IETF
```

Агентство IANA изменило регистрацию est, которая сейчас включает лишь [RFC7030] и не указывает этот документ. В предшествующих черновых версиях этого документа использовалось /.well-known/est вместо /.well-known/brski.

8.3.2. Реестр BRSKI .well-known

Агентство IANA создало реестр BRSKI Well-Known URIs. Записи реестра состоят из 3 полей: URI, Description (описание), Reference (документ). Новые записи добавляются в реестр по процедуре Specification Required [RFC8126]. Исходное содержимое реестра представлено в таблице 1.

Таблица 1. Общеизвестные BRSKI URI.

URI	*Описание*	*Документ*
requestvoucher	От заявителя к регистратору и от регистратора к MASA	RFC 8995
voucher_status	От заявителя к регистратору	RFC 8995
requestauditlog	От регистратора к MASA	RFC 8995
enrollstatus	От заявителя к регистратору	RFC 8995

8.4. Реестр PKIX

Агентство IANA зарегистрировало значение id-mod-MASAURLExtn2016(96) из реестра pkix(7) id-mod(0).

Агентство IANA выделило реестра id-pe (Structure of Management Information (SMI) Security for PKIX Certificate Extension) значение 32 для id-pe-masa-url, приводящее к OID 1.3.6.1.5.5.7.1.32.

8.5. Телеметрия состояния BRSKI у заявителя

Агентство IANA создало реестр BRSKI Parameters с таблицей Pledge BRSKI Status Telemetry Attributes. Новые элементы добавляются в реестр по процедуре Specification Required. Исходная таблица в соответствии с этим документом (5.7. Телеметрия статуса заявителя BRSKI) содержит поля:

- version;
- status;
- reason;
- reason-context.

8.6. Имена DNS для служб

Агентство IANA зарегистрировало указанные ниже имена служб.

```
Service Name:  brski-proxy
Transport Protocol(s):  tcp
Assignee:  IESG <iesg@ietf.org>
Contact:  IESG <iesg@ietf.org>
Description: The Bootstrapping Remote Secure Key Infrastructure Proxy
Reference:  RFC 8995

Service Name:  brski-registrar
Transport Protocol(s):  tcp
Assignee:  IESG <iesg@ietf.org>
Contact:  IESG <iesg@ietf.org>
Description: The Bootstrapping Remote Secure Key Infrastructure
   Registrar
Reference:  RFC 8995
```

8.7. Имена целей GRASP

Агентство IANA зарегистрировало значение AN_Proxy в таблице GRASP Objective Names реестра GRASP Parameter. Спецификация для этого значения приведена в параграфе 4.1.1. Анонсирование Proxy GRASP. Зарегистрировано также значение AN_join_registrar в таблице GRASP Objective Names реестра GRASP Parameter. Спецификация значения приведена в параграфе 4.3. Обнаружение посредника и связь с регистратором.

9. Применимость к ACP

Этот документ предлагает решение для защищённой начальной загрузки как определено в Using an Autonomic Control Plane for Stable Connectivity of Network Operations, Administration, and Maintenance (OAM) [RFC8368], A Reference Model for Autonomic Networking [RFC8993], а также в параграфах 3.2 (Secure Bootstrap over an Unconfigured Network) и 6.2 (ACP Domain, Certificate, and Network) документа An Autonomic Control Plane (ACP) [RFC8994].

Описанный в этом документе протокол подходит для множества вариантов применения без ANIMA. Такое использование протокола будет реализовано в других средах с другими компромиссами в части приватности, безопасности, надёжности и автономности от изготовителей. Поэтому для такого применения потребуются свои заявления о применимости и нужно решить уникальные вопросы приватности и безопасности для сред применения.

Плоскость управления ACP, загружаемая протоколом BRSKI, обычно применяется у средних и крупных провайдеров Internet. Аналогичные предприятия, имеющие значительные возможности связности L3, также получат существенные преимущества, особенно при наличии множества сайтов. Не исключаются и сети, построенные в основном на соединениях L2, но связи, создаваемые и поддерживаемые ACP в этом случае не будут отражать топологию, пока все устройства не участвуют в ACP. В ACP посредник присоединения (Join Proxy) является близким, поскольку взаимодействие между заявителем и Join Proxy осуществляется исключительно по адресам IPv6 link-local. Близость Join Proxy к регистратору последний проверяет с использованием адресов ANI ACP IPv6 ULA. Эти адреса не маршрутизируются через Internet, поэтому при корректной работе Join Proxy заявление о близости соблюдается. Для других применений BRSKI потребуется аналогичный анализ, если они применяют заявления о близости.

Как указано в уставе ANIMA, эта работа «сосредоточена на профессионально управляемых сетях». В таких сетях имеются операторы, которые могут устанавливать, настраивать и применять функции регистратора. Оператор принимает решения о покупке и знает о изготовителях, чьё оборудования он ожидает увидеть в сети. Такой оператор также может выполнять начальную загрузку устройств с использованием последовательной консоли. Автоматический (zero-touch) механизм, описанный здесь и в документе ACP [RFC8994] обеспечивает высокую эффективность, в частности снижает необходимость отправки квалифицированных специалистов для настройки оборудования.

По мере развития технологии становилось ясно, что не всегда удаётся выполнить действия автоматически и иной раз требуется привлекать человека, для чего в параграфе 7.2 представлены некоторые механизмы. Изготовитель должен предоставить хотя бы один из механизмов настройки «в одно касание» (one-touch), позволяющих выполнить зачисление при недоступности какого-либо сервера изготовителя (такого как MASA).

Протокол BRSKI внедряется в средах, где уже имеется достаточное число фирменных систем управления от производителей. Предполагается, что такие системы исчезнут нескоро и будут использовать защищённые свидетельства, предоставляемые BRSKI. Требования связности с такими системами управления выполняются ACP.

9.1. Эксплуатационные требования

В этом параграфе собраны рабочие требования на основе трёх ролей в BRSKI - MASA, владелец (домена) и устройство. Следует отметить, что изготовитель может играть две роли, он создаёт программы/микрокод для устройства, а также может быть оператором MASA.

Требования в этом разделе представлены с использованием уровней BCP 14 [RFC2119] [RFC8174]. Здесь нет новых нормативных требований, просто приведён обзор некоторых аспектов с учётом ролей. Это применимо также к варианту применения ANIMA ACP. Для иных применения могут быть похожие, но отличающиеся требования.

9.1.1. Операционные требования к MASA

Изготовитель **должен** обеспечить доступность сетевой (online) службы, называемой MASA. Служба **должна** быть доступна по URL, закодированному в расширении сертификата IDevID, описанного в параграфе 2.3.2. Расширение MASA URI. Эта служба **должна** иметь доступ к секретному ключу, которым подписаны артефакты ваучеров [RFC8366]. Открытый ключ, сертификат или цепочка сертификатов **должны** встраиваться в устройство как часть прошивки (микрокода). Изготовителю **рекомендуется** организовать депонирование ключей подписи в соответствии обычной практикой депонирования исходного кода программ [softwareescrow].

MASA воспринимает voucher-request от владельцев доменов в соответствии с операционной практикой, подходящей для устройства. Это может варьироваться от восприятия запросов от любого владельца домена (обслуживание в порядке поступления запросов в стиле TOFU) до полной интеграции с каналами продаж, когда владелец домена должен быть точно идентифицирован закреплённым сертификатом TLS или процессом аутентификации HTTP. MASA создаёт артефакты подписанных ваучеров в соответствии со своими внутренними правилами.

Агент MASA **должен** вести журнал аудита для доступных устройств. Журнал позволяет простое кэширование и MASA **может** счесть полезным поместить его содержимое в сеть CDN[1].

9.1.2. Операционные требования к владельцу домена

Владелец домена **должен** поддерживать сервер EST [RFC7030] с описанными в этом документе расширениями (JRC или регистратор). Этот сервер JRC/EST **должен** анонсировать себя, используя GRASP в ACP. Сервер EST обычно размещается в сетевом операционном центре (Network Operations Center) организации. Владелец домена **может** поддерживать внутренний CA отдельно от EST или **может** объединить все действия в одном устройстве. Выбор архитектуры зависит от масштаба организации и требований к отказоустойчивости. **Можно** анонсировать в ACP несколько экземпляров JRC из разных мест для обеспечения требуемого резервирования.

Чтобы узнать, какие устройства (и каких изготовителей) принимаются в сеть владельца домена, владельцу **следует** поддерживать список разрешённых изготовителей. Это **можно** связать с отделами закупки, чтобы знать серийные номера устройств.

Владельцу домена **следует** использовать полученную в результате наложенную сеть ACP для управления устройствами взамен унаследованных автономных (out-of-band) механизмов.

Владельцу домена изготовителей **следует** поддерживать один или несколько серверов EST, которые могут сложить для обновления сертификатов домена (LDevID), размещённых на устройствах. Это **могут** быть те же серверы, что и JRC, а **можно** использовать отдельный набор устройств в зависимости от требований к отказоустойчивости.

Организация **должна** принять подходящие меры предосторожности против потери доступа к секретному ключу CA. Для этого подходят аппаратные модули защиты и/или разделение секретов.

9.1.3. Операционные требования к устройству

Устройства **должны** поставляться со встроенными привязками доверия, позволяющими проверить ваучеры от MASA. Устройства **должны** поставляться с (уникальными) сертификатами IDevID, включающими серийный номер и расширение MASA URL.

Предполагается что устройства находят посредников присоединения с использованием GRASP и затем соединяются с JRC, используя протокол, описанный в этом документе.

Предполагается, что после проверки владельца домена по ваучеру устройства будут зачисляться в домен с использованием EST. Ожидается, что после этого устройства сформируют ACP, используя IPsec с адресами IPv6 link-local,как описано в [RFC8994]. После зачисления устройства ему **следует** прослушивать адрес JRC, применяя GRASP, а также **следует** включить себя как Join Proxy и анонсировать это на всех каналах (интерфейсах) через GRASP DULL. Предполагается, что устройства обновят свои сертификаты до завершения срока их действия.

10. Вопросы приватности

10.1. Журнал аудита MASA

Журнал аудита MASA включает domainID каждого домена, для для которого был выдан ваучер. Эти сведения тесно связаны с фактическим отождествлением домена. MASA может потребоваться дополнительная защита от атак на

[1]Content Delivery Network - сеть доставки содержимого.

службы (Denial-of-Service, параграф 11.1), которая может включать сбор не указанных здесь сведений. Это может дать службе MASA сведения для подробного представления о имеющихся в домене устройствах.

Имеется много решений для смягчения рисков. Домен может поддерживать некоторую приватность, поскольку он может быть не аутентифицирован и не связан полномочно с цепочкой поставок.

Кроме того, domainID фиксирует лишь неаутентифицированный идентификатор ключа субъекта для домена. Чувствительный к приватности домен теоретически может создавать новое значение domainID для каждого внедряемого устройства. Точно так же для него скорее всего будут приобретаться устройства, поддерживающие заявление близости от производителя, не требующее интеграции с каналами продаж. Это обеспечит должный уровень защиты приватности при сохранении характеристик защиты, обеспечиваемых проверкой журнала аудита на основе регистратора.

10.2. Что раскрывает BRSKI-EST

На предварительном этапе соединения BRSKI-EST между заявителем и регистратором каждая сторона раскрывает другой свои сертификаты. Для заявителя это включает атрибут serialNumber, MASA URL, и отождествление, подписанное сертификатом IDevID.

TLS 1.2 раскрывает отождествления сертификатов находящимся в пути наблюдателям, включая Join Proxy. TLS 1.3 раскрывает отождествления сертификатов лишь конечным элементам, но соединение является предварительным и злоумышленник на пути (MITM) может видеть сертификаты. Это видят также регистраторы, которые видны заявителю, но не входят в предполагаемый домен.

Сертификат регистратора достаточно условен с точки зрения протокола BRSKI. Поскольку проверка [RFC6125] не предполагается, для содержимого легко организовать псевдоминимизацию. Любое устройство, способное видеть Join Proxy, способно соединиться с регистратором и получить отождествление соответствующей сети. Даже при псевдоминимизации сертификата можно сопоставить различные соединения в разных местах, относящиеся к одному объекту. Возникновение существенных проблем с приватностью маловероятно в ANIMA ACP с BRSKI, но может вызывать беспокойство у других пользователей BRSKI.

Сертификат заявителя может раскрываться враждебному Join Proxy, организовавшему MITM-атаку на предварительное соединение TLS. Такой злоумышленник может раскрыть отождествление заявителя третьей стороне.

Было бы полезно исследовать механизм многоэтапного согласования ключей с многосторонней аутентификацией, включающий какое-либо нераскрываемое (zero-knowledge) подтверждение. В идеале такой механизм позволит избежать раскрытия отождествлений, пока заявитель, регистратор и MASA не согласуют транзакцию. Такой механизм должен обнаруживать местоположение MASA, не имея отождествления заявителя и MASA. Эта часть проблемы может оказаться неразрешимой.

10.3. Что BRSKI-MASA раскрывает изготовителю

В устройствах, ориентированных на потребителя механизм «звонков домой» (call-home) для IoT вызывает серьёзные проблемы приватности (см. примеры в [livingwithIoT] и [IoTstrangeThings]). Применение BRSKI ACP нацелено не на индивидуальное использование устройств IoT, а на создание сетей предприятий и ISP, работающих без участия человека (zero-touch), где call-home создаёт новый класс вопросов приватности и управления жизненным циклом.

Нужно повторить, что механизм BRSKI-MASA срабатывает лишь 1 раз в процессе ввода устройства в эксплуатацию. Механизм чётко определён и, хотя и применяет шифрование TLS, теоретически может быть доступен для аудита. Соединения не создаётся при включении устройства или при обычном перезапуске (возможен, но крайне маловероятен, сброс устройства к заводским установкам при исключительной ситуации в процессе обновления прошивки, после которого потребуется заново выполнять зачисление устройства с организацией нового соединения).

В BRSKI механизм звонков домой осуществляется через регистратор владельца и передаваемая информация может быть напрямую проверена (аудит) владельцем устройства. Это существенно отличается от многих протоколов call-home, где устройство звонит самостоятельно, используя недокументированный протокол.

При этом содержимое подписанной части voucher-request от заявителя не может быть изменено, но не шифруется регистратором. Возможность аудита сообщений владельцем сети обеспечивает защиту от кражи данных вредоносным (nefarious) заявителем. Отметим ещё раз, что передаваемые сообщения защищаются с помощью TLS.

Обмен BRSKI-MASA раскрывает изготовителю указанные ниже сведения.

- Отождествление зачисляемого устройства раскрывается передачей подписанного voucher-request с серийным номером. Изготовитель обычно может связать серийный номер с моделью устройства.
- Отождествление владельца домена через привязки доверия. Однако это не глобальное имя, связанное с PKI в WebPKI, и может быть псевдонимом. При интеграции с каналами продаж MASA аутентифицируется владельцем домена через прикрепленный сертификат или иной метод аутентификации HTTP (параграф 5.5.4).
- Время активации устройства.
- IP-адрес регистратора владельца домена. Для ISP и предприятий адрес IP чётко указывает геолокацию владельца. Никакие расширения приватности адресов IP [RFC8981] не могут решить эту проблему, ведь поиск с помощью whois чётко указывает ISP или предприятие по старшим битам адреса. Пассивный атакующий, наблюдающий за соединением, может понять, что данное предприятие или ISP является клиентом конкретного производителя оборудования. Модели зачисляемых в домен устройств не раскрываются.

На основе указанных выше сведений изготовитель может отслеживать для конкретного устройства от одного псевдонима домена до следующего. При интеграции с каналами продажи отождествления не будут псевдонимами.

Изготовитель знает IP-адрес регистратора, но не может видеть IP-адрес самого устройства. Изготовитель не может отследить для устройства конкретное физическое или сетевое местоположение и знает лишь размещение регистратора (вероятно в центральном офисе предприятия или ISP).

Описанную выше ситуацию следует отличать от регистрации у изготовителя устройства отдельным человеком. У одного человека обычно нет нескольких офисов и регистратор, скорей всего, размещается в одной сети с устройством. Изготовителю, продающему продукцию для маршрутизации или коммутации организациям, вряд ли следует удивляться дополнительным закупкам таких устройств. Однако отклонения от сложившейся тенденции или установленного базового уровня будут заметны.

Ситуация не улучшается, если предприятие или /ISP использует службу анонимизации, такую как Tor [Dingledine], поскольку соединение TLS 1.2 раскроет применяемый ClientCertificate, чётко указывающий предприятие или ISP. TLS 1.3 в этом отношении лучше, но активный атакующий всё-таки может раскрыть участников, организовав MITM-атаку на первую попытку (прерывание путём сброса TCP - RST), а затем пропустив последующее соединение через себя.

Изготовитель может смешивать трафик BRSKI-MASA с остальным трафиком на своём сайте, разместив MASA за тем же балансировщиком нагрузки (или их набором), что и обычный маркетинговый сайт. Это имеет смысл с точки зрения прямого планирования пропускной способности, поскольку может применяться общий набор услуг (и средств смягчения DDoS-атак). Поскольку соединения BRSKI-MASA включают обмен TLS ClientCertificate, это легко отслеживается в TLS 1.2, а анализ трафика позволяет раскрыть сведения даже в TLS 1.3. Однако это не делает такой подход неактуальным. Могут быть и другие организационные причины отделять маркетинговый сайт (который обычно часто обновляется, отдаётся сторонним организациям и т. п.) от службы MASA, которая должна безотказно работать десятилетиями.

10.4. Изготовители и использованное или украденное оборудование

Как указано выше, изготовитель получает сведения всякий раз, когда устройство с заводскими настройками по умолчанию выполняет автоматическую (zero-touch) начальную загрузку и пытается зачислиться у регистратора владельца домена. Поэтому изготовитель может отказать в выдаче сертификата, если он видит, что новый владелец отличается от прежнего.

1. Это можно рассматривать как признак кражи устройства. Если законный владелец сообщил изготовителю о краже, при включении новым владельцем устройства с механизмом zero-touch будет раскрыто отождествление и местоположение этого владельца.
2. При легитимной смене владельца исходный хозяин может сообщить изготовителю о продаже или изготовитель может просто разрешать перепродажу, если не оговорено иное. В этом случае просто меняется владелец.
3. Изготовитель может отказаться выдавать новый ваучер при смена владельца (как в случае кражи).
4. Имеется возможность защиты изготовителем устройств от кражи, когда он принимает ответственность за защиту легитимного владельца от мошеннических заявлений о краже устройств. Без такой защиты претензия приведёт к отказу изготовителя в выдаче нового ваучера и если устройство будет сброшено к заводским настройкам (например, при замене основных компонентов), начальная загрузка станет невозможной.
5. Если изготовитель решил прекратить поддержку устройств (end-of-line) или владелец не оплатил поддержку, производитель может отклонить выдачу новых ваучеров. Это не является новым в отрасли и развёрнуто уже много систем лицензирования со значительно более жёсткими последствиями.

В этом параграфе описано 5 ситуаций, когда изготовитель может использовать систему ваучеров для соблюдения условий лицензии. Изготовитель, пытающийся обеспечить соблюдение лицензий через ваучеры, может счесть это малоэффективным, поскольку условия применяются лишь при зачислении устройств, а не ежедневно или ежемесячно.

10.5. Изготовители и «серое» оборудование

Изготовители устройств часто продают разную продукцию на рынках разных регионов. Доступность конкретной продукции на определённом рынке может зависеть от разницы в ценах, вопросов поддержки (некоторые рынки требуют документацию и поддержку на местных языках) и государственного регулирования экспорта (например, возможность поставки строгой криптографии на конкретный рынок). При получении владельцем домена устройства (возможно, нового) для другого рынка это называется приобретением на «сером рынке» (Grey Market).

Изготовитель может отказаться выдавать ваучер предприятию или ISP на основе его местоположения. Имеется много способов определить место - по геолокации регистратора, сведениям о клиенте в каналах продаж, доступности продукции на данном рынке. Если устройство имеет модуль GPS, координаты могут включаться в расширение ваучера.

Указанные выше действия не являются новыми или незаконными. Многие изготовители поставляли (экспортные) варианты устройств со слабой криптографией в прошивке в течение десятилетий. Первой задачей предприятия или ISP всегда был вход (login) в систему изготовителя, подтверждение своих «прав» (сведения о стране и оплате поддержки) и получение обновлённой прошивки или лицензионного ключа для активации нужной прошивки.

BRSKI позволяет автоматизировать указанные выше процессы и может таким способом поощрять дифференцирование за счёт снижения расходов на её реализацию. Проблема, с которой изготовители столкнутся в указанном выше автоматизированном процессе, состоит в том, что устройство поставляется в одну страну с одним набором правил (законов или разрешений), а реестр доменов размещается в другой. Выбор применяемых правил требует проработки - изготовитель может предполагать, что устройство получено через серый рынок, а на деле клиент может просто быть транснациональным предприятием.

10.6. Некоторые способы защиты от вмешательства изготовителей

Наиболее очевидным является отказ от покупки продукции. Следует выбирать изготовителей, которые открыто заявляют о своей политике и не меняют её без причины.

В параграфе 7.4.3 описаны некоторые способы, которыми изготовитель может предоставить механизм для управления привязками доверия и встроенными сертификатами (IDevID) в качестве расширения. Имеется много механизмов и некоторые могут требовать большой работы для получения корректного результата. Эти механизмы не меняют поток описанного здесь протокола, а скорее позволяют поменять исходные представления о доверии и являются направлением будущих работ по стандартизации.

Замена привязок проверки ваучеров (обычно указывающих на исходный MASA изготовителя) привязками нового владельца позволяет тому выпускать ваучеры для последующих владельцев. Это можно сделать, если продающий (прежний) владелец запустит свой агент MASA.

Протокол BRSKI зависит от привязок доверия и отождествления на устройстве. Управление этими элементами облегчает несколько новых режимов работы без внесения изменений в протокол BRSKI. Это включают автономный режим, где владелец домена поддерживает внутренний агент MASA для всех устройств, режим перепродажи, когда первый владелец домена становится MASA для следующего, и службы, где агрегатор покупает большое число устройств и может служить псевдонимом MASA для устройств разных изготовителей.

Хотя замена IDevID не требуется для указанных выше режимов, изготовитель может поддерживать это. Некоторые могут захотеть сменить IDevID в качестве индикации прекращения гарантий, для других требования приватности в некоторых средах могут считать это стандартной рабочей практикой.

Как отмечено в конце параграфа 5.8.1, можно было бы выполнить новую работу по использованию технологии распределенного согласия для журнала аудита. Это сохранило бы полезность журнала даже при использовании цепочки MASA в результате смены владельца.

10.7. Прекращение деятельности изготовителя

Общее опасение связано с возможность прекращения деятельности изготовителя, в результате чего владельцы устройств не смогут получить ваучеры для имеющейся продукции. Это может возникнуть при развёртывании устройств со склада или при необходимости заново инициализировать уже развёрнутое устройство.

MASA означает уполномоченный изготовителем орган подписания, чтобы подчеркнуть, что это не обязательно сам изготовитель. Ожидается, что появятся специализированные поставщики услуг, которые будут выпускать ваучеры, подобно компаниям, предоставляющим услуги электронной почты, рекламы или уборки. Кроме того, ожидается, что в рамках соглашений об обслуживании изготовители организуют депонирование секретных ключей, чтобы услуги MASA могла предоставить сторонняя организация. Для исходного кода это делается давно.

11. Вопросы безопасности

В этом документе подробно описан протокол начальной загрузки, балансирующий рабочие задачи и вопросы безопасности. Как отмечено во введении и повторено в разделе 7, протокол поддерживает режимы, позволяющие локальному администратору и владельцу получить дополнительный контроль над операциями за счёт некоторого снижения уровня безопасности. В этом разделе более подробно рассматриваются конкретные аспекты.

Для облегчения ведения журнала и административного надзора заявитель сообщает регистратору статус анализа ваучера в дополнение к проверке регистратором журналов MASA. В случае отказа эти сведения информируют о возможно вредоносном регистраторе. Это обязательно во всех случаях, поскольку информирует администратора о ситуациях, где отказ указывает проблему. Регистратору **рекомендуется** проверять журналы MASA, если не получен статус телеметрии ваучера.

Для содействия работе клиентов EST с ограничениями клиент **должен** поддерживать модель аутентификации (параграф 3.3.2 в [RFC7030]). В разделе 7 эти требования изменены указанием, что регистратор **может** воспринимать устройства при отказе криптографической аутентификации. Это отражает текущую (слабую) практику поставки устройств без криптографического отождествления, которая **не рекомендуется**.

На предварительном этапе соединения заявитель **должен** считать заголовки и содержимое HTTP недоверенными. Библиотеки HTTP регулярно раскрывают незащищённый трафик HTTP, чего у надёжных библиотек быть не должно.

Заявитель может параллельно выполнять операции с несколькими найденными регистраторами. Как отмечено выше, при этом используются разные значения nonce, но в результате может быть получено от MASA несколько ваучеров, если все регистраторы пытаются заявить устройство. Это не отказ и заявитель выбирает ваучер на основе своей логики. Регистраторы, проверяющие журнал, увидят несколько записей и учтут это при своём анализе.

11.1. DoS-атаки на MASA

В некоторых случаях агент MASA может быть недоступен или откажется сотрудничать с регистратором. Это включает активные DoS-атаки, разделение сети (partition), смену правил MASA или иные ситуации, когда правила MASA отклоняют заявку. Это создаёт риск для владельца регистратора, поскольку поведение MASA может ограничивать возможность начальной загрузки устройства заявителя. Регистратор может снизить риск, запрашивая и сохраняя долгосрочные копии ваучеров без nonce. Таким способом они гарантируют начальную загрузку своих устройств.

Ваучеры без nonce сами создают проблему безопасности. Если регистратор прежнего домена может перехватить обмен данными по протоколу, он сможет воспользоваться прежним ваучером без nonce для контроля над устройством заявителя даже после его продажи. Этот риск снижается регистрацией выдачи таких ваучеров в журнале аудита MASA, который проверяется следующим регистратором, и начальная загрузка происходит лишь при сбросе к заводским установкам. Это обеспечивает баланс между независимостью MASA при будущей начальной загрузке и защитой самой начальной загрузки. Контроль регистратором запроса и аудита ваучеров без nonce обеспечивает владельцам выбор подходящего баланса.

Агент MASA открыт для DoS-атак, в которых злоумышленник заявляет неограниченное число устройств. Проблему смягчает возможность регистратора представлять действительного клиента изготовителя даже без проверки владения конкретными устройствами заявителя. Подпись заявителя в его voucher-request, пересылаемая регистратором в поле prior-signed-voucher-request его voucher-request, существенно снижает риск, гарантируя MASA возможность подтвердить близость между заявителем и подающим запрос регистратором. Интеграция с цепочкой продаж (знай своих клиентов) является дополнительным средством снижения рисков для провайдеров MASA и производителей.

11.2. Устойчивость DomainID к атакам Second-Preimage

Идентификатор domainID служит ссылкой на домен в журнале аудита. Ожидается, что domainID рассчитывается с помощью хэш-функции, устойчивой к атакам second-preimage, позволяющим регистратору создавать ложные записи.

11.3. Доступность качественных случайных значений

Значения nonce, применяемые заявителем в voucher-request, следует генерировать строгой криптографической последовательностью (параграф 6.2 в [RFC4086]). Аналогичное требование применяется к TLS. В частности, реализациям следует принимать в внимание усовершенствования из раздела 3 и параграфа 3.4 в [RFC4086]. Случайная затравка (seed), используемая при загрузке, должна быть уникальной у каждого устройства при каждой начальной загрузке. Сброс устройства к заводским установкам не отменяет это требование.

11.4. Свежесть Voucher-Request

Высказывались предложения по включению в voucher-request заявителя некого содержимого (a nonce), предоставляемого регистратором и/или MASA, чтобы те могли проверить свежесть запроса ваучера заявителем. Есть ряд операционных проблем с получением от MASA nonce для заявителя. Несколько проще получить случайное значение от регистратора, но тот ещё не подтверждён, поэтому nonce от него имеет мало смысла. При решении этих оперативных вопросов возникают проблемы с приватностью и логистикой, поэтому для рассмотрения вопроса нужно разобраться с ценностью такого подхода. В этом параграфе рассматриваются последствия отсутствия свежего voucher-request от заявителя.

Поскольку регистратор аутентифицирует заявителя, полная MITM-атака невозможна, несмотря на предварительную аутентификацию TLS (см. раздел). Вместо этого рассматривается случай с фальшивым регистратором (Rm), который взаимодействует с заявителем параллельно или в непосредственной близости от предусмотренного регистратора (этот вариант поддерживается намеренно, как описано в параграфе 4.1). Фальшивый регистратор Rm может получить подписанный MASA ваучер напрямую или через произвольных посредников. Предположим, что MASA воспринимает voucher-request от регистратора (поскольку Rm взаимодействует с легитимным регистратором в соответствии с данными цепочки поставок или MASA лишь ведёт журнал аудита), тогда выдаётся ваучер, связывающий заявителя с Rm. Такой ваучер при его передаче заявителю будет связывать того с Rm и позволит заявителю завершить временное состояние. Заявитель будет доверять Rm и при наличии уязвимостей, которыми может воспользоваться Rm для получения полного административного контроля, можно предположить угрозу для предусмотренного регистратора.

Это смягчается проверкой предусмотренным регистратором журналов аудита, доступных у MASA, как описано в параграфе 5.8. Rm может принять voucher-request и подождать, пока предусмотренный регистратор завершит процесс предоставления полномочий, а затем передать voucher-request. Этот запрос ваучера от заявителя будет «устаревшим» в том смысле, что он больше не соответствует внутреннему состоянию заявителя. Для использования полученного ваучера Rm потребуется удалить устаревшее значение nonce или угадать будущее значение nonce от заявителя. Для снижения вероятности этого требуется генерировать строгие случайные или псевдослучайные значения nonce.

Для использования полученного ваучера Rm потребуется атака на заявителя для возврата того в состояние начальной загрузки. Это требует стирания заявителем текущей конфигурации и инициирования его начальной загрузки. Это не проще прямого захвата управления заявителем, но с учётом такой возможности **рекомендуется** выполнить в целевой сети указанные ниже действия.

- Постоянный мониторинг сети на предмет неожиданных попыток начальной загрузки заявителей.
- Извлечение и проверка сведений из журнала MASA при возникновении непредвиденных событий. Rm будет указан в журналах вместе с nonce.

11.5. Доверие к изготовителю

Расширения BRSKI для EST позволяет полностью настроить привязки доверия для домена у заявителя. Связь заданных на производстве привязок доверия с привязками в домене задаётся подписанным артефактом ваучера.

При отсутствии должной проверки ключа подписи IDevID изготовителя возникает риск восприятия сетью заявителей, которым не следует входить в сеть. Поскольку адрес MASA изготовителя включается в IDevID с использованием расширения из параграфа 2.3, заявитель-злоумышленник не будет иметь проблем при взаимодействии с MASA для создания действительного ваучера. Однако BRSKI не меняет основ модели доверия владельца домена изготовителю. Предполагая, что заявитель использует свой IDevID с EST [RFC7030] и BRSKI, домен (регистратор) должен доверять изготовителю. Организация такого доверия между доменом и изготовителем выходит за рамки BRSKI. Имеется ряд механизмов, которые подходят для этого, включая указанные ниже.

- Настройка привязок доверия для каждого изготовителя вручную.
- Механизм TOFU. При первом появлении привязки доверия выдаётся запрос человеку, а затем привязка помещается в доверенное хранилище. При этом возникает риск, связанный с похожими именами (например, dem0.example и demO.example) даже при проверке сопоставления ключа с именем, такой как WebPKI.
- Сканирование привязки доверия по QR-коду на упаковке (на деле это ручной вариант механизма TOFU).
- Работа с каналом продаже, где привязки доверия предоставляются как часть процесса продажи и возможно включаются в упаковочную накладную или счёт-фактуру.
- Участие в консорциуме со всеми изготовителями данной категории оборудования (например, кабельных модемов), все члены которой специально для этого подписаны CA. Сегодня так делает CableLabs, выполняя аутентификацию и проверку полномочий как часть [docsisroot] и [TR069].

WebPKI обеспечивает пригодную привязку открытого ключа к имени изготовителя и аутентифицирует ключ. Однако это не обеспечивает разумной проверки полномочий изготовителя, поэтому не может применяться напрямую.

11.6. Поддержка привязок доверия изготовителем

BRSKI зависит от изготовителя при организации привязок доверия для устройства заявителя. Артефакт ваучера, подписанный MASA, проверяется заявителем с использованием таких привязок. Это предполагает, что изготовителю нужно поддерживать доступ к ключу подписи, который заявитель может проверить. Заявителю нужно поддерживать возможность создания подписей, которые можно проверить в течение всего срока службы устройства. Связь между

продолжительностью службы устройства и возможным временем его подключения (onboarding) зависит от способа использования устройства. Инвентаризация хранящихся на складе устройств может не проводиться десятки лет.

У изготовителей имеются веские криптографические проблемы не поддерживать доступ к секретному ключу десятилетиями. Изготовитель в такой ситуации может воспользоваться встроенной в заявитель долгосрочной привязкой CA, а затем может быть использована цепочка с обычным набором сертификатов CMS. Это может увеличить размер артефактов ваучеров, но это не создаёт проблем в средах без ограничений. Имеется ещё несколько вариантов, которые изготовители могут рассмотреть. Например, нет причин устанавливать на каждое устройство один и тот же набор привязок доверия. Устройства, выпущенные на разных заводах или в разное время, могут иметь различные встроенные привязки доверия и запись о принадлежности устройства к определённой партии сохраняется в архивной базе данных, из которой изготовитель может узнать, для каких привязок доверия подписывать артефакт.

Помимо беспокойства о долгосрочном доступе к секретным ключам, основным фактором, ограничивающим срок годности многих устройств являются используемые криптоалгоритмы. Устройство, выпущенное в 2019 г. будет способно аппаратными или программными средствами проверить алгоритмы, принятые в этом году, но не будет иметь защиты от атак (как квантовых, так и фон Неймана), которые ещё не придуманы. Эта проблема ортогональна проблеме доступа к секретным ключам, но, вероятно, является доминирующей и ограничивает срок службы устройств, хранящихся на складе. Если было бы возможно обновить прошивку для поддержки новых криптографических механизмов (пока устройство находится на складе), одновременно можно было бы обновить и привязки доверия.

Набор стандартный рабочих процедур для поддержки долгосрочных секретных ключей хорошо документирован. Например, WebPKI обеспечивает ряд вариантов аудита [cabforumaudit], а корневые операции DNSSEC хорошо описаны в [dnssecroot].

Неясно, примут ли изготовители такой уровень предосторожностей и насколько сильны экономические стимулы поддержки надлежащего уровня защиты.

В следующем параграфе оцениваются риски, связанные с компрометацией ключа подписи IDevID у изготовителя, а затем рассматриваются риски при компрометации ключа MASA. Заключительный параграф рассматривает ситуацию, когда сам web-сервер MASA находится под контролем злоумышленника, но ключ подписи MASA безопасно хранится в аппаратном модуле, не подключённом напрямую.

11.6.1. Компрометация ключей подписи IDevID у изготовителя
Злоумышленник с доступом к ключу, который изготовитель применяет для подписи сертификатов IDevID, пожет создавать контрафактные устройства, которые могут представляться продукцией конкретного изготовителя, на деле отличаясь от таковой и являясь, по сути, троянскими конями. Поскольку злоумышленник контролирует MASA URL в сертификате, регистратора можно убедить обратиться к MASA злоумышленника. Регистратору при этом не нужно быть в каком-либо неразборчивом режиме. Помимо создания поддельных устройств злоумышленник может отзывать выпущенные сертификаты, если процесс сертификации IDevID полагается на распространяемые списки отзыва (CRL).

В остальном компрометация не представляется опасной для устройств, которые уже развёрнуты или находятся в коробках, ожидая внедрения (локальное хранение для замены). Однако операторы не смогут доверять устройствам, которые находились на неконтролируемом складе, поскольку они не могут знать, настоящие ли это устройства.

11.6.2. Компрометация ключей подписи MASA
Здесь нужно учитывать два момента времени - попадание ключа к злоумышленнику и признание взлома агент MASA.

11.6.2.1. Возможности атакующего при компрометации ключей MASA
Злоумышленник с доступом к ключу подписи MASA может создавать ваучеры для развёрнутых и хранящихся на складе устройств. Для использования таких ваучеров нужно выполнить два условия - устройство должно пройти полный цикл загрузки с заводскими настройками, а регистратор должен связаться с MASA злоумышленника.

Если злоумышленник контролирует видимый устройству регистратор, проблем с доставкой фиктивного ваучера не возникает. Практическим примером может служить атака на ЦОД в точке обмена трафиком (peering) ISP (приватной или общедоступной - IX). В такой ситуации к оборудованию уже подключены кабели, ведущие к другим устройствам (партнёрам по IX), через которые можно доставить фальшивый ваучер. Сложнее будет сбросить устройство к заводским настройкам, но это можно сделать путём обмана (социальная психология) инженерного персонала ЦОД. В большинстве закрытых ячеек имеются вентиляционные отверстия и с помощью длинной «скрепки» можно дотянуться до кнопки сброса устройства. Как только нужная часть оборудования ISP скомпрометирована, его можно использовать для компрометации соединённых с ним устройств (даже через длинные соединения - long haul), при условии их сброса к заводским настройкам.

Приведённый пример представляется маловероятным, поскольку требует физического доступа, но при наличии эксплойта, вызывающего не прямое нарушения, а скорее отказ, обеспечивающий сброс к заводским настройкам, операции можно выполнить удалённо.

Сказанное выше относится к использованию BRSKI с ANI. В случае применения IEEE 802.11 или 802.15.4 не возникает необходимости подключаться к устройству напрямую, но сброс к заводским настройкам все равно нужен. Физическое владение устройством не требуется, как отмечено выше, при наличии способа удалённого сброса к заводским настройкам. Для некоторых потребительских устройствах со слабой реализацией конечные пользователи могут знать о необходимости регулярного сброса настроек.

Авторы не смогли придумать вариант атаки, где скомпрометированная подпись ваучера позволяет злоумышленнику внедрить скомпрометированного заявителя в сеть оператора. Это связано с наличием контроля оператора за взаимодействием между регистратором и MASA, не позволяющего внедрить фальшивый ваучер.

11.6.2.2. Риски после компрометации ключей
Когда оператор MASA осознает компрометацию ключа подписи ваучеров, он должен выполнить несклько действий, описанных ниже.

Во-первых, он **должен** выпустить обновление прошивки для всех устройств, использовавших этот ключ как привязку доверия, чтобы они перестали доверять ваучерам с этим ключом. Это повлияет на развёрнутые устройства, но ини не выполняют операций по включению, поэтому исправление (patch) не будет критическим.

Устройства, находящиеся в коробках (на складе) останутся уязвимыми, пока их не обновят. Оператору разумно распаковать устройства, включить их в безопасной среде и обновить прошивки. Это может делать не только конечный оператор, но и дистрибьютор, хранящий запасные устройства (части). Для дорогостоящих устройств рекомендуется проверять работоспособность регулярно (обычно это не занимает более 4 часов).

Если в процессе включения выполняется аттестация версий прошивки, в этом процессе оператору будет рекомендовано обновить программу перед запуском устройства в работу. К сожалению, это не поможет при использовании злоумышленником своего регистратора (как указано выше).

Необходимость использования краткосрочных ваучеров разъяснена в параграфе 6.1 [RFC8366]. Применение nonce обеспечивает свежесть, а краткосрочность ваучеров означает, что временные рамки доставки фиктивного ваучера очень малы. Злоумышленнику останутся лишь хранящиеся на складе устройства с долгоживущими ваучерами без nonce, поскольку их уязвимость сохранится.

Основной операционной рекомендацией для изготовителя является подписывание долгосрочных ваучеров без nonce не тем ключом, который применяется для подписи краткосрочных ваучеров. Если оба ключа приходят из одной привязки доверия (CA изготовителя), компрометация ключа этого CA приведёт к компрометации обоих ключей и, вероятно, других ключей, упомянутых в этом разделе.

11.6.3. Компрометация Web-службы MASA

Злоумышленник, захвативший web-службу MASA может организовать ряд атак. Самым очевидным путём является извлечение из базы данных списка клиентов и устройств для продажи этих сведений другим злоумышленникам, которые знают, как найти потенциально уязвимые устройства. Другим распространенным вариантом является установка (kill) службы или нарушение её работы с целью разочаровать клиентов. Это может серьёзно повлиять на возможность развёртывания клиентами новых услуг и станет большой проблемой при аварийном восстановлении.

Хотя компрометация web-службы MASA может вести к компрометации ключа подписи ваучеров у MASA, если подпись создаётся вовне (offboard), например в аппаратном модуле подписей (hardware signing module или HSM), ключи сохраняются в безопасности, но контроль над ними остаётся у злоумышленника. Такой злоумышленник может выпускать ваучеры для любого эксплуатируемого устройства, хотя это устройство ещё потребуется убедить в необходимости сброса к заводским настройкам перед атакой.

Если у злоумышленника есть доступ к ключу, которому доверяют долгосрочные ваучеры без nonce, он может выпустить ваучеры для устройств, которые ещё не введены в эксплуатацию. Такую атаку очень сложно заметить, поскольку потребуется проверить все хранящиеся на складе устройства (дорого и долго). Изготовитель может не пойти на это.

11.7. Вопросы безопасности для модуля YANG

Как указано в параграфе 7.4 (Вопросы безопасности) [RFC8366], заданный в этом документе модуль YANG определяет схему для данных, которые затем инкапсулируются в тип носителя с подписью CMS, как описано в разделе 5 [RFC5652]. Поэтому все данные модели YANG защищены от изменения.

Применение YANG для задания структур данных оператором yang-data является сравнительно новым и отличается от традиционного использования YANG для определения API с доступом по протоколам сетевого управления, таким как NETCONF [RFC6241] и RESTCONF [RFC8040]. Поэтому эти рекомендации не соответствуют шаблону, приведённому в параграфе 3.7 [RFC8407].

12. Литература

12.1. Нормативные документы

[IDevID] IEEE, "IEEE Standard for Local and metropolitan area networks - Secure Device Identity", IEEE 802.1AR, <https://1.ieee802.org/security/802-1ar>.

[ITU.X690] ITU-T, "Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, ISO/IEC 8825-1:2015, August 2015, <https://www.itu.int/rec/T-REC-X.690>.

[REST] Fielding, R.F., "Architectural Styles and the Design of Network-based Software Architectures", 2000, <http://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <https://www.rfc-editor.org/info/rfc3339>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[RFC3927] Cheshire, S., Aboba, B., and E. Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses", RFC 3927, DOI 10.17487/RFC3927, May 2005, <https://www.rfc-editor.org/info/rfc3927>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/info/rfc4086>.

[RFC4519] Sciberras, A., Ed., "Lightweight Directory Access Protocol (LDAP): Schema for User Applications", RFC 4519, DOI 10.17487/RFC4519, June 2006, <https://www.rfc-editor.org/info/rfc4519>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <https://www.rfc-editor.org/info/rfc4862>.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, DOI 10.17487/RFC5272, June 2008, <https://www.rfc-editor.org/info/rfc5272>.

[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, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[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, DOI 10.17487/RFC6125, March 2011, <https://www.rfc-editor.org/info/rfc6125>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <https://www.rfc-editor.org/info/rfc6762>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <https://www.rfc-editor.org/info/rfc6763>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <https://www.rfc-editor.org/info/rfc7030>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[RFC7469] Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning Extension for HTTP", RFC 7469, DOI 10.17487/RFC7469, April 2015, <https://www.rfc-editor.org/info/rfc7469>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/info/rfc7951>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC8366] Watsen, K., Richardson, M., Pritikin, M., and T. Eckert, "A Voucher Artifact for Bootstrapping Protocols", RFC 8366, DOI 10.17487/RFC8366, May 2018, <https://www.rfc-editor.org/info/rfc8366>.

[RFC8368] Eckert, T., Ed. and M. Behringer, "Using an Autonomic Control Plane for Stable Connectivity of Network Operations, Administration, and Maintenance (OAM)", RFC 8368, DOI 10.17487/RFC8368, May 2018, <https://www.rfc-editor.org/info/rfc8368>.

[RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/info/rfc8407>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", RFC 8610, DOI 10.17487/RFC8610, June 2019, <https://www.rfc-editor.org/info/rfc8610>.

[RFC8951] Richardson, M., Werner, T., and W. Pan, "Clarification of Enrollment over Secure Transport (EST): Transfer Encodings and ASN.1", RFC 8951, DOI 10.17487/RFC8951, November 2020, <https://www.rfc-editor.org/info/rfc8951>.

[RFC8981] Gont, F., Krishnan, S., Narten, T., and R. Draves, "Temporary Address Extensions for Stateless Address Autoconfiguration in IPv6", RFC 8981, DOI 10.17487/RFC8981, February 2021, <https://www.rfc-editor.org/info/rfc8981>.

[RFC8990] Bormann, C., Carpenter, B., Ed., and B. Liu, Ed., "GeneRic Autonomic Signaling Protocol (GRASP)", RFC 8990, DOI 10.17487/RFC8990, May 2021, <https://www.rfc-editor.org/rfc/rfc8990>.

[RFC8994] Eckert, T., Ed., Behringer, M., Ed., and S. Bjarnason, "An Autonomic Control Plane (ACP)", RFC 8994, DOI 10.17487/RFC8994, May 2021, <https://www.rfc-editor.org/rfc/rfc8994>.

12.2. Дополнительная литература

[ACE-COAP-EST] van der Stok, P., Kampanakis, P., Richardson, M., and S. Raza, "EST over secure CoAP (EST-coaps)", Work in Progress[1], Internet-Draft, draft-ietf-ace-coap-est-18, 6 January 2020, <https://tools.ietf.org/html/draft-ietf-ace-coap-est-18>.

[ANIMA-CONSTRAINED-VOUCHER] Richardson, M., van der Stok, P., Kampanakis, P., and E. Dijk, "Constrained Voucher Artifacts for Bootstrapping Protocols", Work in Progress, Internet-Draft, draft-ietf-anima-constrained-voucher-10, 21 February 2021, <https://tools.ietf.org/html/draft-ietf-anima-constrained-voucher-10>.

[ANIMA-STATE] Richardson, M., "Considerations for stateful vs stateless join router in ANIMA bootstrap", Work in Progress, Internet-Draft, draft-richardson-anima-state-for-joinrouter-03, 22 September 2020, <https://tools.ietf.org/html/draft-richardson-anima-state-for-joinrouter-03>.

[brewski] Urban Dictionary, "brewski", March 2003, <https://www.urbandictionary.com/define.php?term=brewski>.

[cabforumaudit] CA/Browser Forum, "Information for Auditors and Assessors", August 2019, <https://cabforum.org/information-for-auditors-and-assessors/>.

[Dingledine] Dingledine, R., Mathewson, N., and P. Syverson, "Tor: The Second-Generation Onion Router", August 2004, <https://svn-archive.torproject.org/svn/projects/design-paper/tor-design.pdf>.

[dnssecroot] "DNSSEC Practice Statement for the Root Zone ZSK Operator", December 2017, <https://www.iana.org/dnssec/procedures/zsk-operator/dps-zsk-operator-v2.1.pdf>.

[docsisroot] "CableLabs Digital Certificate Issuance Service", February 2018, <https://www.cablelabs.com/resources/digital-certificate-issuance-service/>.

[imprinting] Wikipedia, "Imprinting (psychology)", January 2021, <https://en.wikipedia.org/w/index.php?title=Imprinting_(psychology)&=999211441>.

[IoTstrangeThings] ESET, "IoT of toys stranger than fiction: Cybersecurity and data privacy update", March 2017, <https://www.welivesecurity.com/2017/03/03/internet-of-things-security-privacy-iot-update/>.

[livingwithIoT] Silicon Republic, "What is it actually like to live in a house filled with IoT devices?", February 2018, <https://www.siliconrepublic.com/machines/iot-smart-devices-reality>.

[minerva] Richardson, M., "Minerva reference implementation for BRSKI", 2020, <https://minerva.sandelman.ca/>.

[minervagithub] "ANIMA Minerva toolkit", <https://github.com/ANIMAgus-minerva>.

[openssl] OpenSSL, "OpenSSL X509 Utility", September 2019, <https://www.openssl.org/docs/man1.1.1/man1/openssl-x509.html/>.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <https://www.rfc-editor.org/info/rfc2131>.

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, DOI 10.17487/RFC2663, August 1999, <https://www.rfc-editor.org/info/rfc2663>.

[RFC5209] Sangster, P., Khosravi, H., Mani, M., Narayan, K., and J. Tardo, "Network Endpoint Assessment (NEA): Overview and Requirements", RFC 5209, DOI 10.17487/RFC5209, June 2008, <https://www.rfc-editor.org/info/rfc5209>.

[RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, DOI 10.17487/RFC6960, June 2013, <https://www.rfc-editor.org/info/rfc6960>.

[RFC6961] Pettersen, Y., "The Transport Layer Security (TLS) Multiple Certificate Status Request Extension", RFC 6961, DOI 10.17487/RFC6961, June 2013, <https://www.rfc-editor.org/info/rfc6961>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <https://www.rfc-editor.org/info/rfc7228>.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <https://www.rfc-editor.org/info/rfc7258>.

[RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <https://www.rfc-editor.org/info/rfc7435>.

[1] Опубликовано в RFC 9148. *Прим. перев.*

[RFC7575] Behringer, M., Pritikin, M., Bjarnason, S., Clemm, A., Carpenter, B., Jiang, S., and L. Ciavaglia, "Autonomic Networking: Definitions and Design Goals", RFC 7575, DOI 10.17487/RFC7575, June 2015, <https://www.rfc-editor.org/info/rfc7575>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

[RFC8993] Behringer, M., Ed., Carpenter, B., Eckert, T., Ciavaglia, L., and J. Nobre, "A Reference Model for Autonomic Networking", RFC 8993, DOI 10.17487/RFC8993, May 2021, <https://www.rfc-editor.org/info/rfc8993>.

[slowloris] Wikipedia, "Slowloris (computer security)", January 2021, <https://en.wikipedia.org/w/index.php?title=Slowloris_(computer_security)&oldid=1001473290/>.

[softwareescrow] Wikipedia, "Source code escrow", March 2020, <https://en.wikipedia.org/w/index.php?title=Source_code_escrow&oldid=948073074>.

[Stajano99theresurrecting] Stajano, F. and R. Anderson, "The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks", 1999, <https://www.cl.cam.ac.uk/~fms27/papers/1999-StajanoAnd-duckling.pdf>.

[TR069] Broadband Forum, "CPE WAN Management Protocol", TR-069, Issue 1, Amendment 6, March 2018, <https://www.broadband-forum.org/download/TR-069_Amendment-6.pdf>.

[W3C.capability-urls] Tennison, J., "Good Practices for Capability URLs", W3C First Public Working Draft, World Wide Web Consortium WD WD-capability-urls-20140218, February 2014, <https://www.w3.org/TR/2014/WD-capability-urls>.

[YANG-KEYSTORE] Watsen, K., "A YANG Data Model for a Keystore", Work in Progress, Internet-Draft, draft-ietf-netconf-keystore-22, 18 May 2021, <https://tools.ietf.org/html/draft-ietf-netconf-keystore-22>.

Приложение A. IPv4 и операции без ANI

Спецификация BRSKI в разделе 4 намеренно рассматривает лишь механизмы для заявителей с адресами IPv6 link-local. В этом приложении даны ненормативные расширения, которые могут быть применены в других средах.

A.1. Адреса IPv4 Link-Local

Вместо адреса IPv6 link-local можно создать адрес IPv4 в соответствии с Dynamic Configuration of IPv4 Link-Local Addresses [RFC3927]. В случае формирования адреса IPv4 link-local начальная загрузка будет продолжаться как для IPv6 с поиском посредника.

A.2. Использование DHCPv4

Заявитель **может** получить IP-адрес по протоколу DHCP ([RFC2131]. Предоставленные DHCP параметры для системы доменных имён (DNS) можно использовать для операций DNS при отказе попыток локального обнаружения.

Приложение B. Варианты mDNS и DNS-SD Proxy Discovery

Обнаружение посредника заявителем (параграф 4.1) **может** выполняться через DNS-based Service Discovery [RFC6763] по протоколу Multicast DNS [RFC6762] для поиска посредника в _brski-proxy._tcp.local. Обнаружение посредником регистратора (параграф 4.3) **может** выполняться поиском служб через по протоколу Multicast DNS для обнаружения _brski-registrar._tcp.local.

Для предотвращения неприемлемого уровня сетевого трафика при использовании mDNS **должны** применяться механизмы, заданные в разделе 7 [RFC6762]. Заявителю **следует** прослушивать незапрошенные широковещательные отклики, как описано в [RFC6762]. Это позволит устройствам избежать анонсирования своего присутствия через широковещание mDNS и просто присоединяться к сети, отслеживая периодические незапрошенные широковещательные отклики.

Обнаружить регистратор **можно** также с помощью поиска служб через DNS по _brski-registrar._tcp.example.com. В этом случае домен example.com обнаруживается, как указано в разделе 11 [RFC6763] (в Приложении A.2 предложено использование параметров DHCP).

Если локальный посредник или регистратор не найден с помощью механизмов the GRASP или описанного выше обнаружения служб через DNS, заявитель **может** обратиться к общеизвестным серверам начальной загрузки изготовителей через поиск в DNS общеизвестных URI, таких как brski-registrar.manufacturer.example.com. Детали URI зависят от изготовителя. Изготовители, применяющие такой метод у заявителя, отвечают за предоставление службы регистратора (см. также параграф 2.7).

Службы DNS, возвращённые при каждом запросе, поддерживаются до завершения начальной загрузки. В случае отказа при загрузке и возврата заявителя в состояние Discovery,он продолжает с того места, где остановился, пытаясь выполнить начальную загрузку. Например, если первый отклик Multicast DNS _bootstrapks._tcp.local не работает, применяется второй, затем третий. Если это не помогает, заявитель переходит к обычному поиску служб через DNS.

Приложение C. Примеры ваучеров

В использование ваучеров вовлечены три сущности - MASA выпускает (подписывает) ваучеры, в ваучере упоминается открытый ключ регистратора, а заявитель проверяет ваучер. Для представления воспроизводимых примеров сначала указываются секретные и открытые ключи для примеров MASA и регистратора.

Ключи взяты из открытой реализации BRSKI, называемый Minerva [minerva], доступной на GitHub [minervagithub]. Представленные здесь ключи используются модульных и интеграционных тестах. Код MASA называется highway, код регистратора - fountain, а пример клиента называется reach.

Компоненты открытых ключей представлены как сертификаты в кодировке base64 и декодированы утилитой openssl x509, чтобы можно было видеть расширения. Использовалась версия 1.1.1c библиотеки и утили [openssl].

C.1. Вовлечённые ключи

Изготовитель имеет CA, где подписываются IDevID заявителя. В дополнение к этому агент подписей изготовителя (MASA) подписывает ваучеры и его сертификат должен распространяться устройствам в процессе производства, чтобы ваучеры можно было проверить.

C.1.1. Орган сертификации изготовителя для подписей IDevID

Приведённый ниже секретный ключ применяется CA для подписания сертификатов IDevID.

```
<CODE BEGINS> file "vendor.key"
-----BEGIN EC PRIVATE KEY-----
MIGkAgEBBDCAYkoLW1IEA5SKKhMMdkTK7sJxk5ybKqYq9Yr5aR7tNwqXyLGS7z8G
8S4w/UJ58BqgBwYFK4EEACKhZANiAAQu5/yktJbFLjMC87h7b+yTreFuF8GwewKH
L4mS0r0dVAQubqDUQcTrjvpXrXCpTojiLCzgp8fzkcUDkZ9LD/M90LDipiLNIOkP
juF8QkoAbT8pMrY83MS8y76wZ7AalNQ=
-----END EC PRIVATE KEY-----
<CODE ENDS>
```

Приведённый ниже открытый ключ служит для проверки сертификатов IDevID. Файл examples/vendor.key.

```
<CODE BEGINS> file "vendor.cert"
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1216069925 (0x487bc125)
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: CN = highway-test.example.com CA
        Validity
            Not Before: Apr 13 20:34:24 2021 GMT
            Not After : Apr 13 20:34:24 2023 GMT
        Subject: CN = highway-test.example.com CA
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (384 bit)
                pub:
                    04:2e:e7:fc:a4:b4:96:c5:2e:33:02:f3:b8:7b:6f:
                    ec:93:ad:e1:6e:17:c1:b0:7b:02:87:2f:89:92:d2:
                    bd:1d:54:04:2e:6e:a0:d4:41:c4:eb:8e:fa:57:ad:
                    70:a9:4e:88:e2:2c:2c:e0:a7:c7:f3:91:c5:03:91:
                    9f:4b:0f:f3:3d:d0:b0:e2:a6:22:cd:20:e9:0f:8e:
                    e1:7c:42:4a:00:6d:3f:29:32:b6:3c:dc:c4:bc:cb:
                    be:b0:67:b0:1a:94:d4
                ASN1 OID: secp384r1
                NIST CURVE: P-384
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
            X509v3 Subject Key Identifier:
                5E:0C:A9:52:5A:8C:DF:A9:0F:03:14:E9:96:F1:80:76:
                8C:53:8A:08
            X509v3 Authority Key Identifier:
                keyid:5E:0C:A9:52:5A:8C:DF:A9:0F:03:14:E9:96:F1:
                80:76:8C:53:8A:08

    Signature Algorithm: ecdsa-with-SHA256
         30:64:02:30:60:37:a0:66:89:80:27:e1:0d:e5:43:9a:62:f1:
         02:bc:0f:72:6d:a9:e9:cb:84:a5:c6:44:d3:41:9e:5d:ce:7d:
         46:16:6e:15:de:f7:cc:e8:3e:61:f9:03:7c:20:c4:b7:02:30:
         7f:e9:f3:12:bb:06:c6:24:00:2b:41:aa:21:6b:d8:25:ed:81:
         07:11:ef:66:8f:06:bf:c8:be:f0:58:74:24:45:39:4d:04:fc:
         31:69:6f:cf:db:fe:61:7b:c3:24:31:ff
-----BEGIN CERTIFICATE-----
MIIB3TCCAWSgAwIBAgIESHvBJTAKBggqhkjOPQQDAjAmMSQwIgYDVQQDDBtoaWdo
d2F5LXRlc3QuZXhhbXBsZS5jb20gQ0EwHhcNMjEwNDEzMjAzNDI0WhcNMjMwNDEz
MjAzNDI0WjAmMSQwIgYDVQQDDBtoaWdod2F5LXRlc3QuZXhhbXBsZS5jb20gQ0Ew
djAQBgcqhkjOPQIBBgUrgQQAIgNiAAQu5/yktJbFLjMC87h7b+yTreFuF8GwewKH
L4mS0r0dVAQubqDUQcTrjvpXrXCpTojiLCzgp8fzkcUDkZ9LD/M90LDipiLNIOkP
juF8QkoAbT8pMrY83MS8y76wZ7AalNSjYzBhMA8GA1UdEwEB/wQFMAMBAf8wDgYD
VR0PAQH/BAQDAgEGMB0GA1UdDgQWBBReDKlSWozfqQ8DFOmW8YB2jFOKCDAfBgNV
HSMEGDAWgBReDKlSWozfqQ8DFOmW8YB2jFOKCDAKBggqhkjOPQQDAgNnADBkAjBg
N6BmiYAn4Q3lQ5pi8QK8D3JtqenLhKXGRNNBnl3OfUYWbhXe98zoPmH5A3wgxLcC
```

```
MH/p8xK7BsYkACtBqiFr2CXtgQcR72aPBr/IvvBYdCRFOU0E/DFpb8/b/mF7wyQx
/w==
-----END CERTIFICATE-----
<CODE ENDS>
```

C.1.2. Пара ключей MASA для подписи ваучеров

MASA - это уполномоченный изготовителем агент подписания (Manufacturer Authorized Signing Authority). Приведённые ниже ключи служат для подписывания ваучеров. Пример сертификата TLS (5.4. Организация BRSKI-MASA TLS) для аутентификации HTTP не представлен, поскольку тот имеет базовую форму. Секретный ключ для подписи ваучеров показан ниже.

```
<CODE BEGINS> file "masa.key"
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIFhdd0eDdzip67kXx72K+KHGJQYJHNy8pkiLJ6CcvxMGoAoGCCqGSM49
AwEHoUQDQgAEqgQVo0S54kT4yfkbBxumdHOcHrpsqbOpMKmiMln3oB1HAW25MJV+
gqi4tMFfSJ0iEwt8kszfWXK4rLgJS2mnpQ==
-----END EC PRIVATE KEY-----
<CODE ENDS>
```

Открытый ключ служит для проверки ваучеров и подписывается показанным выше ключом CA. Файл examples/masa.key.

```
<CODE BEGINS> file "masa.cert"
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 193399345 (0xb870a31)
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: CN = highway-test.example.com CA
        Validity
            Not Before: Apr 13 21:40:16 2021 GMT
            Not After : Apr 13 21:40:16 2023 GMT
        Subject: CN = highway-test.example.com MASA
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:aa:04:15:a3:44:b9:e2:44:f8:c9:f9:1b:07:1b:
                    a6:74:73:9c:1e:ba:6c:a9:b3:a9:30:a9:a2:32:59:
                    f7:a0:1d:47:01:6d:b9:30:95:7e:82:a8:b8:b4:c1:
                    5f:48:9d:22:13:0b:7c:92:cc:df:59:72:b8:ac:b8:
                    09:4b:69:a7:a5
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:FALSE
    Signature Algorithm: ecdsa-with-SHA256
         30:66:02:31:00:ae:cb:61:2d:d4:5c:8d:6e:86:aa:0b:06:1d:
         c6:d3:60:ba:32:73:36:25:d3:23:85:49:87:1c:ce:94:23:79:
         1a:9e:41:55:24:1d:15:22:a1:48:bb:0a:c0:ab:3c:13:73:02:
         31:00:86:3c:67:b3:95:a2:e2:e5:f9:ad:f9:1d:9c:c1:34:32:
         78:f5:cf:ea:d5:47:03:9f:00:bf:d0:59:cb:51:c2:98:04:81:
         24:8a:51:13:50:b1:75:b2:2f:9d:a8:b4:f4:b9
-----BEGIN CERTIFICATE-----
MIIBcDCB9qADAgECAgQLhwoxMAoGCCqGSM49BAMCMCYxJDAiBgNVBAMMG2hpZ2h3
YXktdGVzdC5leGFtcGxlLmNvbSBDQTAeFw0yMTA0MTMyMTQwMTZaFw0yMzA0MTMy
MTQwMTZaMCgxJjAkBgNVBAMMHWhpZ2h3YXktdGVzdC5leGFtcGxlLmNvbSBNQVNB
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEqgQVo0S54kT4yfkbBxumdHOcHrps
qbOpMKmiMln3oB1HAW25MJV+gqi4tMFfSJ0iEwt8kszfWXK4rLgJS2mnpaMQMA4w
DAYDVR0TAQH/BAIwADAKBggqhkjOPQQDAgNpADBmAjEArsthLdRcjW6GqgsGHcbT
YLoyczYl0yOFSYcczpQjeRqeQVUkHRUioUi7CsCrPBNzAjEAhjxns5Wi4uX5rfkd
nME0Mnj1z+rVRwOfAL/QWctRwpgEgSSKURNQsXWyL52otPS5
-----END CERTIFICATE-----
<CODE ENDS>
```

C.1.3. Орган сертификации регистратора

Этот CA зачисляет заявителя, если тот уполномочен, а также подписывает сертификат регистратора.

```
<CODE BEGINS> file "ownerca_secp384r1.key"
-----BEGIN EC PRIVATE KEY-----
MIGkAgEBBDCHnLI0MSOLf8XndiZqoZdqblcPR5YSoPGhPOuFxWy1gFi9HbWv8b/R
EGdRgGEVSjKgBwYFK4EEACKhZANiAAQbf1m6F8MavGaNjGzgw/oxcQ9l9iKRvbdW
gAfb37h6pUVNeYpGlxlZljGxj2l9Mr48yD5bY7VG9qjVb5v5wPPTuRQ/ckdRpHbd
0vC/9cqPMAF/+MJf0/UgA0SLi/IHbLQ=
-----END EC PRIVATE KEY-----
<CODE ENDS>
```

Ниже приведён открытый ключ, указываемый в voucher-request регистратора для индикации близости. Файл examples/ownerca_secp384r1.key

```
<CODE BEGINS> file "ownerca_secp384r1.cert"
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 694879833 (0x296b0659)
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: DC = ca, DC = sandelman,
```

```
          CN = fountain-test.example.com Unstrung Fountain Root CA
        Validity
            Not Before: Feb 25 21:31:45 2020 GMT
            Not After : Feb 24 21:31:45 2022 GMT
        Subject: DC = ca, DC = sandelman,
         CN = fountain-test.example.com Unstrung Fountain Root CA
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (384 bit)
                pub:
                    04:1b:7f:59:ba:17:c3:1a:bc:66:8d:8c:6c:e0:c3:
                    fa:31:71:0f:65:f6:22:91:bd:b7:56:80:07:db:df:
                    b8:7a:a5:45:4d:79:8a:46:97:19:59:96:31:b1:8f:
                    69:7d:32:be:3c:c8:3e:5b:63:b5:46:f6:a8:d5:6f:
                    9b:f9:c0:f3:d3:b9:14:3f:72:47:51:a4:76:dd:d2:
                    f0:bf:f5:ca:8f:30:01:7f:f8:c2:5f:d3:f5:20:03:
                    44:8b:8b:f2:07:6c:b4
                ASN1 OID: secp384r1
                NIST CURVE: P-384
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
            X509v3 Subject Key Identifier:
                B9:A5:F6:CB:11:E1:07:A4:49:2C:A7:08:C6:7C:10:BC:
                87:B3:74:26
            X509v3 Authority Key Identifier:
                keyid:B9:A5:F6:CB:11:E1:07:A4:49:2C:A7:08:C6:7C:
                10:BC:87:B3:74:26

    Signature Algorithm: ecdsa-with-SHA256
         30:64:02:30:20:83:06:ce:8d:98:a4:54:7a:66:4c:4a:3a:70:
         c2:52:36:5a:52:8d:59:7d:20:9b:2a:69:14:58:87:38:d8:55:
         79:dd:fd:29:38:95:1e:91:93:76:b4:f5:66:29:44:b4:02:30:
         6f:38:f9:af:12:ed:30:d5:85:29:7c:b1:16:58:bd:67:91:43:
         c4:0d:30:f9:d8:1c:ac:2f:06:dd:bc:d5:06:42:2c:84:a2:04:
         ea:02:a4:5f:17:51:26:fb:d9:2f:d2:5c
-----BEGIN CERTIFICATE-----
MIICazCCAfKgAwIBAgIEKWsGWTAKBggqhkjOPQQDAjBtMRIwEAYKCZImiZPyLGQB
GRYCY2ExGTAXBgoJkiaJk/IsZAEZFglzYW5kZWxtYW4xPDA6BgNVBAMMM2ZvdW50
YWluLXRlc3QuZXhhbXBsZS5jb20gVW5zdHJ1bmcgRm91bnRhaW4gUm9vdCBDQTAe
Fw0yMDAyMjUyMTMxNDVaFw0yMjAyMjQyMTMxNDVaMG0xEjAQBgoJkiaJk/IsZAEZ
FgJjYTEZMBcGCgmSJomT8ixkARkWCXNhbmRlbG1hbjE8MDoGA1UEAwwzZm91bnRh
aW4tdGVzdC5leGFtcGxlLmNvbSBVbnN0cnVuZyBGb3VudGFpbiBSb290IENBMHYw
EAYHKoZIzj0CAQYFK4EEACIDYgAEG39ZuhfDGrxmjYxs4MP6MXEPZfYikb23VoAH
29+4eqVFTXmKRpcZWZYxsY9pfTK+PMg+W2O1Rvao1W+b+cDz07kUP3JHUaR23dLw
v/XKjzABf/jCX9P1IANEi4vyB2y0o2MwYTAPBgNVHRMBAf8EBTADAQH/MA4GA1Ud
DwEB/wQEAwIBBjAdBgNVHQ4EFgQUuaX2yxHhB6RJLKcIxnwQvIezdCYwHwYDVR0j
BBgwFoAUuaX2yxHhB6RJLKcIxnwQvIezdCYwCgYIKoZIzj0EAwIDZwAwZAIwIIMG
zo2YpFR6ZkxKOnDCUjZaUo1ZfSCbKmkUWIc42FV53f0pOJUekZN2tPVmKUS0AjBv
OPmvEu0w1YUpfLEWWL1nkUPEDTD52BysLwbdvNUGQiyEogTqAqRfF1Em+9kv0lw=
-----END CERTIFICATE-----
<CODE ENDS>
```

C.1.4. Пара ключей регистратора

Регистратор является представителем владельца домена. Приведённый ниже ключ подписывает voucher-requests от регистратора и завершает соединение TLS от заявителя.

```
<CODE BEGINS> file "jrc_prime256v1.key"
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIFZodk+PC5Mu24+ra0sbOjKzan+dW5rvDAR7YuJUOC1YoAoGCCqGSM49
AwEHoUQDQgAElmVQcjS6n+Xd5l/28IFv6UiegQwSBztGj5dkK2MAjQIPV8l8lH+E
jLIOYdbJiI0VtEIf1/Jqt+TOBfinTNOLOg==
-----END EC PRIVATE KEY-----
<CODE ENDS>
```

Открытый ключ указывается в voucher-request от заявителя для индикации близости.

```
<CODE BEGINS> file "jrc_prime256v1.cert"
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1066965842 (0x3f989b52)
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: DC = ca, DC = sandelman,
         CN = fountain-test.example.com Unstrung Fountain Root CA
        Validity
            Not Before: Feb 25 21:31:54 2020 GMT
            Not After : Feb 24 21:31:54 2022 GMT
        Subject: DC = ca, DC = sandelman,
         CN = fountain-test.example.com
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:96:65:50:72:34:ba:9f:e5:dd:e6:5f:f6:f0:81:
```

```
                    6f:e9:48:9e:81:0c:12:07:3b:46:8f:97:64:2b:63:
                    00:8d:02:0f:57:c9:7c:94:7f:84:8c:b2:0e:61:d6:
                    c9:88:8d:15:b4:42:1f:d7:f2:6a:b7:e4:ce:05:f8:
                    a7:4c:d3:8b:3a
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Extended Key Usage: critical
                CMC Registration Authority
            X509v3 Key Usage: critical
                Digital Signature
    Signature Algorithm: ecdsa-with-SHA256
         30:65:02:30:66:4f:60:4c:55:48:1e:96:07:f8:dd:1f:b9:c8:
         12:8d:45:36:87:9b:23:c0:bc:bb:f1:cb:3d:26:15:56:6f:5f:
         1f:bf:d5:1c:0e:6a:09:af:1b:76:97:99:19:23:fd:7e:02:31:
         00:bc:ac:c3:41:b0:ba:0d:af:52:f9:9c:6e:7a:7f:00:1d:23:
         c8:62:01:61:bc:4b:c5:c0:47:99:35:0a:0c:77:61:44:01:4a:
         07:52:70:57:00:75:ff:be:07:0e:98:cb:e5
-----BEGIN CERTIFICATE-----
MIIB/DCCAYKgAwIBAgIEP5ibUjAKBggqhkjOPQQDAjBtMRIwEAYKCZImiZPyLGQB
GRYCY2ExGTAXBgoJkiaJk/IsZAEZFglzYW5kZWxtYW4xPDA6BgNVBAMMM2ZvdW50
YWluLXRlc3QuZXhhbXBsZS5jb20gVW5zdHJ1bmcgRm91bnRhaW4gUm9vdCBDQTAe
Fw0yMDAyMjUyMTMxNTRaFw0yMjAyMjQyMTMxNTRaMFMxEjAQBgoJkiaJk/IsZAEZ
FgJjYTEZMBcGCgmSJomT8ixkARkWCXNhbmRlbG1hbjEiMCAGA1UEAwwZZm91bnRh
aW4tdGVzdC5leGFtcGxlLmNvbTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABJZl
UHI0up/l3eZf9vCBb+lInoEMEgc7Ro+XZCtjAI0CD1fJfJR/hIyyDmHWyYiNFbRC
H9fyarfkzgX4p0zTizqjKjAoMBYGA1UdJQEB/wQMMAoGCCsGAQUFBwMcMA4GA1Ud
DwEB/wQEAwIHgDAKBggqhkjOPQQDAgNoADBlAjBmT2BMVUgelgf43R+5yBKNRTaH
myPAvLvxyz0mFVZvXx+/1RwOagmvG3aXmRkj/X4CMQC8rMNBsLoNr1L5nG56fwAd
I8hiAWG8S8XAR5k1Cgx3YUQBSgdScFcAdf++Bw6Yy+U=
-----END CERTIFICATE-----
<CODE ENDS>
```

C.1.5. Пара ключей заявителя

Заявитель имеет пару ключей IDevID, создаваемую при производстве.

```
<CODE BEGINS> file "idevid_00-D0-E5-F2-00-02.key"
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIBHNh6r8QRevRuo+tEmBJeFjQKf6bpFA/9NGoltv+9sNoAoGCCqGSM49
AwEHoUQDQgAEA6N1Q4ezfMAKmoecrfb0OBMc1AyEH+BATkF58FsTSyBxs0SbSWLx
FjDOuwB9gLGn2TsTUJumJ6VPw5Z/TP4hJw==
-----END EC PRIVATE KEY-----
<CODE ENDS>
```

Сертификат, применяемый регистратором для поиска MASA представлен ниже.

```
<CODE BEGINS> file "idevid_00-D0-E5-F2-00-02.cert"
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 521731815 (0x1f18fee7)
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: CN = highway-test.example.com CA
        Validity
            Not Before: Apr 27 18:29:30 2021 GMT
            Not After : Dec 31 00:00:00 2999 GMT
        Subject: serialNumber = 00-D0-E5-F2-00-02
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:03:a3:75:43:87:b3:7c:c0:0a:9a:87:9c:ad:f6:
                    f4:38:13:1c:d4:0c:84:1f:e0:40:4e:41:79:f0:5b:
                    13:4b:20:71:b3:44:9b:49:62:f1:16:30:ce:bb:00:
                    7d:80:b1:a7:d9:3b:13:50:9b:a6:27:a5:4f:c3:96:
                    7f:4c:fe:21:27
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                45:88:CC:96:96:00:64:37:B0:BA:23:65:64:64:54:08:
                06:6C:56:AD
            X509v3 Basic Constraints:
                CA:FALSE
            1.3.6.1.5.5.7.1.32:
                ..highway-test.example.com:9443
    Signature Algorithm: ecdsa-with-SHA256
         30:65:02:30:62:2a:db:be:34:f7:1b:cb:85:de:26:8e:43:00:
         f9:0d:88:c8:77:a8:dd:3c:08:40:54:bc:ec:3d:b6:dc:70:2b:
         c3:7f:ca:19:21:9a:a0:ab:c5:51:8e:aa:df:36:de:8b:02:31:
         00:b2:5d:59:f8:47:c7:ed:03:97:a8:c0:c7:a8:81:fa:a8:86:
         ed:67:64:37:51:7a:6e:9c:a3:82:4d:6d:ad:bc:f3:35:9e:9d:
         6a:a2:6d:7f:7f:25:1c:03:ef:f0:ba:9b:71
-----BEGIN CERTIFICATE-----
MIIBrzCCATWgAwIBAgIEHxj+5zAKBggqhkjOPQQDAjAmMSQwIgYDVQQDDBtoaWdo
d2F5LXRlc3QuZXhhbXBsZS5jb20gQ0EwIBcNMjEwNDI3MTgyOTMwWhgPMjk5OTEy
MzEwMDAwMDBaMBwxGjAYBgNVBAUTETAwLUQwLUU1LUYyLTAwLTAyMFkwEwYHKoZI
```

```
zj0CAQYIKoZIzj0DAQcDQgAEA6N1Q4ezfMAKmoecrfb0OBMc1AyEH+BATkF58FsT
SyBxs0SbSWLxFjDOuwB9gLGn2TsTUJumJ6VPw5Z/TP4hJ6NZMFcwHQYDVR0OBBYE
FEWIzJaWAGQ3sLojZWRkVAgGbFatMAkGA1UdEwQCMAAwKwYIKwYBBQUHASAEHxYd
aGlnaHdheS10ZXN0LmV4YW1wbGUuY29tOjk0NDMwCgYIKoZIzj0EAwIDaAAwZQIw
YirbvjT3G8uF3iaOQwD5DYjId6jdPAhAVLzsPbbccCvDf8oZIZqgq8VRjqrfNt6L
AjEAsl1Z+EfH7QOXqMDHqIH6qIbtZ2Q3UXpunKOCTW2tvPM1np1qom1/fyUcA+/w
uptx
-----END CERTIFICATE-----
<CODE ENDS>
```

C.2. Примеры процессов

Приведённые ниже примеры JSON используют строки размером 60 символов, что ведёт к включению символов перевода строки, неприемлемых для JSON. Поэтому перед обработкой строки нужно объединить.

Для удобства вывод asn1parse был ограничен 68 без перевода строк.

C.2.1. От заявителя к регистратору

Как описано в параграфе 5.2, заявитель подписывает свой voucher-request, содержащий открытый ключ регистратора в поле proximity-registrar-cert. Кодировка base64 ниже для удобства представлена с размером строк 60 символов.

```
<CODE BEGINS> file "vr_00-D0-E5-F2-00-02.b64"
MIIGcAYJKoZIhvcNAQcCoIIGYTCCBl0CAQExDTALBglghkgBZQMEAgEwggOJBgkqhkiG
9w0BBwGgggN6BIIDdnsiaWV0Zi12b3VjaGVyLXJlcXVlc3Q6dm91Y2hlciI6eyJhc3Nl
cnRpb24iOiJwcm94aW1pdHkiLCJjcmVhdGVkLW9uIjoiMjAyMS0wNC0xM1QxNzo0Mzoy
My43NDctMDQ6MDAiLCJzZXJpYWwtbnVtYmVyIjoiMDAtRDAtRTUtRjItMDAtMDIiLCJu
b25jZSI6Ii1fWEU5eks5cThMbDFxeWxNdExLZWciLCJwcm94aW1pdHktcmVnaXN0cmFy
LWNlcnQiOiJNSUlCL0RDQ0FZS2dBd0lCQWdJRVA1aWJVakFLQmdncWhrak9QUVFEQWpC
dE1SSXdFQVlLQ1pJbWlaUHlMR1FCR1JZQ1kyRXhHVEFYQmdvSmtpYUprL0lzWkFFWkZn
bHpZVzVrWld4dFlXNHhQREE2QmdOVkJBTU1NMlp2ZFc1MFlXbHVMWFJsYzNRdVpYaGhi
WEJzWlM1amIyMGdWVzV6ZEhKMWJtY2dSbTkxYm5SaGFXNGdVbTl2ZENCRFFUQWVGdzB5
TURBeU1qVXlNVE14TlRSYUZ3MHlNakF5TWpReU1UTXhOVFJhTUZNeEVqQVFCZ29Ka2lh
SmsvSXNaQUVaRmdKallURVpNQmNHQ2dtU0pvbVQ4aXhrQVJrV0NYTmhibVJsYkcxaGJq
RWlNQ0FHQTFVRUF3d1pabTkxYm5SaGFXNHRkR1Z6ZEM1bGVHRnRjR3hsTG1OdmJUQlpN
Qk1HQnlxR1NNNDlBZ0VHQ0NxR1NNNDlBd0VIQTBJQUJKWmxVSEkwdXAvbDNlWmY5dkNC
YitsSW5vRU1FZ2M3Um8rWFpDdGpBSTBDRDFmSmZKUi9oSXl5RG1IV31ZaU5GYlJDSDlm
eWFyZmt6Z1g0cDB6VGl6cWpLakFvTUJZR0ExVWRKUUVCL3dRTU1Bb0dDQ3NHQVFVRkJ3
TWNNQTRHQTFVZER3RUIvd1FFQXdJSGdEQUtCZ2dxaGtqT1BRUURBZ05vQURCbEFqQm1U
MkJNVlVnZWxnZjQzUis1eUJLTlJUYUhteVBBdkx2eHl6MG1GVlp2WHgrLzFSd09hZ212
RzNhWG1Sa2ovWDRDTVFDOHJNTkJzTG9OcjFMNW5HNTZmd0FkSThoaUFXRzhTOFhBUjVr
MUNneDNZVVFCU2dkU2NGY0FkZisrQnc2WXkrVT0ifX2gggGyMIIBrjCCATWgAwIBAgIE
DYOv2TAKBggqhkjOPQQDAjAmMSQwIgYDVQQDDBtoaWdod2F5LXRlc3QuZXhhbXBsZS5j
b20gQ0EwIBcNMjEwNDEzMjAzNzM5WhgPMjk5OTEyMzEwMDAwMDBaMBwxGjAYBgNVBAUM
ETAwLUQwLUU1LUYyLTAwLTAyMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEA6N1Q4ez
fMAKmoecrfb0OBMc1AyEH+BATkF58FsTSyBxs0SbSWLxFjDOuwB9gLGn2TsTUJumJ6VP
w5Z/TP4hJ6NZMFcwHQYDVR0OBBYEFEWIzJaWAGQ3sLojZWRkVAgGbFatMAkGA1UdEwQC
MAAwKwYIKwYBBQUHASAEHxYdaGlnaHdheS10ZXN0LmV4YW1wbGUuY29tOjk0NDMwCgYI
KoZIzj0EAwIDZwAwZAIwTmlG8sXkKGNbwbKQcYMapFbmSbnHHURFUoFuRqvbgYX7FlXp
BczfwF2kllNuujigAjAow1kc4r55EmiH+OMEXjBNlWlBSZC5QuJjEf0Jsmxssc+pucjO
J4ShqnexMEy7bjAxggEEMIIBAAIBATAuMCYxJDAiBgNVBAMMG2hpZ2h3YXktdGVzdC51
eGFtcGxlLmNvbSBDQQIEDYOv2TALBglghkgBZQMEAgGgaTAYBgkqhkiG9w0BCQMxCwYJ
KoZIhvcNAQcBMBwGCSqGSIb3DQEJBTEPFw0yMTA0MTMyMTQzMjNaMC8GCSqGSIb3DQEJ
BDEiBCBJwhyYibIjeqeR3bOaLURzMlGrc3F2X+kvJ1errtoCtTAKBggqhkjOPQQDAgRH
MEUCIQCmYuCE61HFQXH/E16GDOCsVquDtgr+Q/6/Du/9QkzA7gIgf7MFhAIPW2PNwRa2
vZFQAKXUbimkiHKzXBA8md0VHbU=
<CODE ENDS>
```

Декодирование ASN1 артефакта из файла examples/vr_00-D0-E5-F2-00-02.b64 имеет вид

```
  0:d=0  hl=4 l=1648 cons: SEQUENCE
  4:d=1  hl=2 l=   9 prim: OBJECT            :pkcs7-signedData
 15:d=1  hl=4 l=1633 cons: cont [ 0 ]
 19:d=2  hl=4 l=1629 cons: SEQUENCE
 23:d=3  hl=2 l=   1 prim: INTEGER           :01
 26:d=3  hl=2 l=  13 cons: SET
 28:d=4  hl=2 l=  11 cons: SEQUENCE
 30:d=5  hl=2 l=   9 prim: OBJECT            :sha256
 41:d=3  hl=4 l= 905 cons: SEQUENCE
 45:d=4  hl=2 l=   9 prim: OBJECT            :pkcs7-data
 56:d=4  hl=4 l= 890 cons: cont [ 0 ]
 60:d=5  hl=4 l= 886 prim: OCTET STRING      :{"ietf-voucher-request:v
950:d=3  hl=4 l= 434 cons: cont [ 0 ]
954:d=4  hl=4 l= 430 cons: SEQUENCE
958:d=5  hl=4 l= 309 cons: SEQUENCE
962:d=6  hl=2 l=   3 cons: cont [ 0 ]
964:d=7  hl=2 l=   1 prim: INTEGER           :02
967:d=6  hl=2 l=   4 prim: INTEGER           :0D83AFD9
973:d=6  hl=2 l=  10 cons: SEQUENCE
975:d=7  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
985:d=6  hl=2 l=  38 cons: SEQUENCE
987:d=7  hl=2 l=  36 cons: SET
989:d=8  hl=2 l=  34 cons: SEQUENCE
991:d=9  hl=2 l=   3 prim: OBJECT            :commonName
996:d=9  hl=2 l=  27 prim: UTF8STRING        :highway-test.example.com
1025:d=6  hl=2 l=  32 cons: SEQUENCE
1027:d=7  hl=2 l=  13 prim: UTCTIME           :210413203739Z
1042:d=7  hl=2 l=  15 prim: GENERALIZEDTIME   :29991231000000Z
```

```
1059:d=6  hl=2 l=  28 cons: SEQUENCE
1061:d=7  hl=2 l=  26 cons: SET
1063:d=8  hl=2 l=  24 cons: SEQUENCE
1065:d=9  hl=2 l=   3 prim: OBJECT            :serialNumber
1070:d=9  hl=2 l=  17 prim: UTF8STRING        :00-D0-E5-F2-00-02
1089:d=6  hl=2 l=  89 cons: SEQUENCE
1091:d=7  hl=2 l=  19 cons: SEQUENCE
1093:d=8  hl=2 l=   7 prim: OBJECT            :id-ecPublicKey
1102:d=8  hl=2 l=   8 prim: OBJECT            :prime256v1
1112:d=7  hl=2 l=  66 prim: BIT STRING
1180:d=6  hl=2 l=  89 cons: cont [ 3 ]
1182:d=7  hl=2 l=  87 cons: SEQUENCE
1184:d=8  hl=2 l=  29 cons: SEQUENCE
1186:d=9  hl=2 l=   3 prim: OBJECT            :X509v3 Subject Key Ident
1191:d=9  hl=2 l=  22 prim: OCTET STRING      [HEX DUMP]:04144588CC9696
1215:d=8  hl=2 l=   9 cons: SEQUENCE
1217:d=9  hl=2 l=   3 prim: OBJECT            :X509v3 Basic Constraints
1222:d=9  hl=2 l=   2 prim: OCTET STRING      [HEX DUMP]:3000
1226:d=8  hl=2 l=  43 cons: SEQUENCE
1228:d=9  hl=2 l=   8 prim: OBJECT            :1.3.6.1.5.5.7.1.32
1238:d=9  hl=2 l=  31 prim: OCTET STRING      [HEX DUMP]:161D6869676877
1271:d=5  hl=2 l=  10 cons: SEQUENCE
1273:d=6  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
1283:d=5  hl=2 l= 103 prim: BIT STRING
1388:d=3  hl=4 l= 260 cons: SET
1392:d=4  hl=4 l= 256 cons: SEQUENCE
1396:d=5  hl=2 l=   1 prim: INTEGER           :01
1399:d=5  hl=2 l=  46 cons: SEQUENCE
1401:d=6  hl=2 l=  38 cons: SEQUENCE
1403:d=7  hl=2 l=  36 cons: SET
1405:d=8  hl=2 l=  34 cons: SEQUENCE
1407:d=9  hl=2 l=   3 prim: OBJECT            :commonName
1412:d=9  hl=2 l=  27 prim: UTF8STRING        :highway-test.example.com
1441:d=6  hl=2 l=   4 prim: INTEGER           :0D83AFD9
1447:d=5  hl=2 l=  11 cons: SEQUENCE
1449:d=6  hl=2 l=   9 prim: OBJECT            :sha256
1460:d=5  hl=2 l= 105 cons: cont [ 0 ]
1462:d=6  hl=2 l=  24 cons: SEQUENCE
1464:d=7  hl=2 l=   9 prim: OBJECT            :contentType
1475:d=7  hl=2 l=  11 cons: SET
1477:d=8  hl=2 l=   9 prim: OBJECT            :pkcs7-data
1488:d=6  hl=2 l=  28 cons: SEQUENCE
1490:d=7  hl=2 l=   9 prim: OBJECT            :signingTime
1501:d=7  hl=2 l=  15 cons: SET
1503:d=8  hl=2 l=  13 prim: UTCTIME           :210413214323Z
1518:d=6  hl=2 l=  47 cons: SEQUENCE
1520:d=7  hl=2 l=   9 prim: OBJECT            :messageDigest
1531:d=7  hl=2 l=  34 cons: SET
1533:d=8  hl=2 l=  32 prim: OCTET STRING      [HEX DUMP]:49C21C9889B223
1567:d=5  hl=2 l=  10 cons: SEQUENCE
1569:d=6  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
1579:d=5  hl=2 l=  71 prim: OCTET STRING      [HEX DUMP]:3045022100A662
```

Код JSON, содержащийся в voucher-request, имеет вид

```
{"ietf-voucher-request:voucher":{"assertion":"proximity","cr
eated-on":"2021-04-13T17:43:23.747-04:00","serial-number":"0
0-D0-E5-F2-00-02","nonce":"-_XE9zK9q8Ll1qylMtLKeg","proximit
y-registrar-cert":"MIIB/DCCAYKgAwIBAgIEP5ibUjAKBggqhkjOPQQDA
jBtMRIwEAYKCZImiZPyLGQBGRYCY2ExGTAXBgoJkiaJk/IsZAEZFglzYW5kZ
WxtYW4xPDA6BgNVBAMMM2ZvdW50YWluLXRlc3QuZXhhbXBsZS5jb20gVW5zd
HJ1bmcgRm91bnRhaW4gUm9vdCBDQTAeFw0yMDAyMjUyMTMxNTRaFw0yMjAyM
jQyMTMxNTRaMFMxEjAQBgoJkiaJk/IsZAEZFgJjYTEZMBcGCgmSJomT8ixkA
RkWCXNhbmRlbG1hbjEiMCAGA1UEAwwZZm91bnRhaW4tdGVzdC5leGFtcGxlL
mNvbTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABJZlUHI0up/l3eZf9vCBb
+lInoEMEgc7Ro+XZCtjAI0CD1fJfJR/hIyyDmHWyYiNFbRCH9fyarfkzgX4p
0zTizqjKjAoMBYGA1UdJQEB/wQMMAoGCCsGAQUFBwMcMA4GA1UdDwEB/wQEA
wIHgDAKBggqhkjOPQQDAgNoADBlAjBmT2BMVUgelgf43R+5yBKNRTaHmyPAv
Lvxyz0mFVZvXx+/1RwOagmvG3aXmRkj/X4CMQC8rMNBsLoNr1L5nG56fwAdI
8hiAWG8S8XAR5k1Cgx3YUQBSgdScFcAdf++Bw6Yy+U="}}
```

C.2.2. От регистратора к MASA

Как указано в параграфе 5.5, регистратор подписывает свой voucher-request и включает voucher-request заявителя в prior-signed-voucher-request.

```
<CODE BEGINS> file "parboiled_vr_00-D0-E5-F2-00-02.b64"
MIIPYwYJKoZIhvcNAQcCoIIPVDCCD1ACAQExDTALBglghkgBZQMEAgEwggl4BgkqhkiG
9w0BBwGggglpBIIJZXsiaWV0Zi12b3VjaGVyLXJlcXVlc3Q6dm91Y2hlciI6eyJhc3Nl
cnRpb24iOiJwcm94aW1pdHkiLCJjcmVhdGVkLW9uIjoiMjAyMS0wNC0xM1QyMTo0Mzoy
My43ODdaIiwic2VyaWFsLW51bWJlciI6IjAwLUQwLUU1LUYyLTAwLTAyIiwibm9uY2Ui
OiItX1hFOXpLOXE4TGwxcXlsTXRMS2VnIiwicHJpb3Itc2lnbmVkLXZvdWNoZXItcmVx
dWVzdCI6Ik1JSUdjQVlKS29aSWh2Y05BUWNDb01JR1lUQ0NCbDBDQVFFeERUQUxCZ2xn
aGtnQlpRTUVBZ0V3Z2dPSkJna3Foa2lHOXcwQkJ3R2dnZ042QklJRGRuc2lhV1YwWmkx
MmIzVmphR1Z5TFhKbGNYVmxjM1E2ZG05MVkyaGxjaUk2ZXlKaGMzTmxjblJwYjI0aU9p
SndjbTk0YVcxcGRIa2lMQ0pqY21WaGRHVmtMVzl1SWpvaU1qQXlNUzB3TkMweE0xUXhO
em8wTXpveU15NDNORGN0TURRNk1EQWlMQ0p6WlhKcFlXd3RiblZ0WW1WeUlqb2lNREF0
UkRBdFJUVXRSak10TURBdE1ESWlMQ0p1YjI1alpTSTZJaTFmV0VVNWVrczVjVGhNYkRG
```

```
   eGVXeE5kRXhMWldjaUxDSndjbTk0YVcxcGRIa3RjbVZuYVhOMGNtRnlMV05sY25RaU9p
   Sk5TVWxDTDBSRFEwRlpTMmRCZDBsQ1FXZEpSVkExYVdKVmFrRkxRbWRuY1docmFrOVFV
   VkZFUVdwQ2RFMVNTWGRGUVZsTFExcEpiV2xhVUhsTVIxRkNSMUpaUTFreVJYaEhWRUZZ
   UW1kdlNtdHBZVXByTDBseldrRkZXa1puYkhwWlZ6VnJXbGQ0ZEZsWE5IaFFSRUUyUW1k
   T1ZrSkJUVTFOTWxwMlpGYzFNRmxYYkhWTVdGSnNZek5SZFZwWWFHaGlXRUp6V2xNMWFt
   SXlNR2RXVnpWNlpFaEtNV0p0WTJkU2JUa3hZbTVTYUdGWE5HZFZiVGwyWkVOQ1JGRlVR
   V1ZHZHpCNVRVUkJlVTFxVlhsTlZFMTRUbFJTWVVaM01IbE5ha0Y1VFdwUmVVMVVUWGhP
   VkZKaFRVWk5lRVZxUVZGQ1oyOUthMmxoU21zdlNYTmFRVVZhUm1kS2FsbFVSVnBOUWlO
   SFEyZHRVMHB2YlZRNGFYaHJRVkpyVjBOWVRtaGliVkpzWWtjeGFHSnFSV2xOUTBGSFFU
   RlZSVUYzZDFwYWJUa3hZbTVTYUdGWE5IUmtSMVo2WkVNMWJHVkhSblJqUjNoc1RHMU9k
   bUpVUWxwTlFrMUhRbmx4UjFOTk5EbEJaMFZIUTBOeFIxTk5ORGxCZDBWSVFUQkpRVUpL
   V214VlNFa3dkWEF2YkRObFdtWTVka05DWWl0c1NXNXZSVTFGWjJNM1VtOHJXRnBEZEdw
   QlNUQkRSREZtU21aS1VpOW9TWGw1UkcxSVYzbFphVTVHWWxKRFNEbG1lV0Z5Wm10Nlox
   ZzBjREI2VkdsNmNXcExha0Z2VFVKWlIwRXhWV1JLVVVWQ0wzZFJUVTFCYjBkRFEzTkhR
   VkZWUmtKM1RXTk5RVFJIUVRGVlpFUjNSVU12ZDFGRlFYZEpTR2RFUVV0Q1oyZHhhR3Rx
   VDFCUlVVUkJaMDV2UVVSQ2JFRnFRbTFVTWtKTlZsVm5aV3huWmpRelVpczFlVUpMVGxK
   VVlVaHRlVkJCZGt4MmVIbDZNRzFHVmxwMldIZ3JMekZTZDA5aFoyMTJSek5oV0cxU2Ey
   b3ZXRFJEVFZGRE9ISk5Ua0p6VEc5T2NqRk1OVzVITlRabWQwRmtTVGhvYVVGWFJ6aFRP
   RmhCVWpWck1VTm5lRE5aVlZGQ1UyZGtVMk5HWTBGa1ppc3JRbmMyV1hrclZUMGlmWDJn
   Z2dHeU1JSUJyakNDQVRXZ0F3SUJBZ01FRFlPdjJUQUtCZ2dxaGtqT1BRUURBakFtTVNR
   d0lnWURWUVFEREJ0b2FXZG9kMkY1TFhSbGMzUXVaWGhoYlhCc1pTNWpiMjBnUTBFd0lC
   Y05NakV3TkRFek1qQXpOek01V2hnUE1qazVPVEV5TXpFd01EQXdNREJhTUJ3eEdqQVlC
   Z05WQkFVTUVUQXdMVVF3TFVVMUxVWXlMVEF3TFRBeU1Ga3dFd1lIS29aSXpqMENBUVlJ
   S29aSXpqMERBUWNEUWdBRUE2TjFRNGV6Zk1BS21vZWNyZmIwT0JNYzFBeUVIK0JBVGtG
   NThGc1RTeUJ4czBTYlNXTHhGakRPdXdCOWdMR24yVHNUVUp1bUo2VlB3NVovVFA0aEo2
   TlpNRmN3SFFZRFZSME9CQllFRkVXSXpKYVdBR1Ezc0xvalpXUmtWQWdHYkZhdE1Ba0dB
   MVVkRXdRQ01BQXdLd1lJS3dZQkJRVUhBU0FFSHhZZGFHbG5hSGRoZVMxMFpYTjBMbVY0
   WVcxd2JHVXVZMjl0T2prME5ETXdDZ1lJS29aSXpqMEVBd01EWndBd1pBSXdUbWxHOHNY
   a0tHTmJ3YktRY1lNYXBGYm1TYm5ISFVSRlVvRnVScXZiZ1lYN0ZsWHBCY3pmd0Yya2xs
   TnV1amlnQWpBb3cxa2M0cjU1RW1pSCtPTUVYakJObFdsQlNaQzVRdUpqRWYwSnNteHNz
   YytwdWNqT0o0U2hxbmV4TUV5N2JqQXhnZ0VFTUlJQkFBSUJBVEF1TUNZeEpEQWlCZ05W
   QkFNTUcyaHBaMmgzWVhrdGRHVnpkQzVsZUdGdGNHeGxMbU52YlNCRFFRSUVEWU92MlRB
   TEJnbGdoa2dCWlFNRUFnR2dhVEFZQmdrcWhraUc5dzBCQ1FNeEN3WUpLb1pJaHZjTkFR
   Y0JNQndHQ1NxR1NJYjNEUUVKQlRFUEZ3MHlNVEEwTVRNeU1UUXpNak5hTUM4R0NTcUdT
   SWIzRFFFSkJERWlCQ0JKd2h5WWliSWplcWVSM2JPYUxVUnpNbEdyYzNGMlgra3ZKMWVy
   cnRvQ3RUQUtCZ2dxaGtqT1BRUURBZ1JITUVVQ0lRQ21ZdUNFNjFIRlFYSC9FMTZHRE9D
   c1ZxdUR0Z3IrUS82L0R1LzlRa3pBN2dJZ2Y3TUZoQUlQVzJQTndSYTJ2WkZRQUtYVWJp
   bWtpSEt6WEJBOG1kMFZIYlU9In19oIIEbzCCAfwwggGCoAMCAQICBD+Ym1IwCgYIKoZI
   zj0EAwIwbTESMBAGCgmSJomT8ixkARkWAmNhMRkwFwYKCZImiZPyLGQBGRYJc2FuZGVs
   bWFuMTwwOgYDVQQDDDNmb3VudGFpbi10ZXN0LmV4YW1wbGUuY29tIFVuc3RydW5nIEZv
   dW50YWluIFJvb3QgQ0EwHhcNMjAwMjI1MjEzMTU0WhcNMjIwMjI0MjEzMTU0WjBTMRIw
   EAYKCZImiZPyLGQBGRYCY2ExGTAXBgoJkiaJk/IsZAEZFglzYW5kZWxtYW4xIjAgBgNV
   BAMMGWZvdW50YWluLXRlc3QuZXhhbXBsZS5jb20wWTATBgcqhkjOPQIBBggqhkjOPQMB
   BwNCAASWZVByNLqf5d3mX/bwgW/pSJ6BDBIHO0aPl2QrYwCNAg9XyXyUf4SMsg5h1smI
   jRW0Qh/X8mq35M4F+KdM04s6oyowKDAWBgNVHSUBAf8EDDAKBggrBgEFBQcDHDAOBgNV
   HQ8BAf8EBAMCB4AwCgYIKoZIzj0EAwIDaAAwZQIwZk9gTFVIHpYH+N0fucgSjUU2h5sj
   wLy78cs9JhVWb18fv9UcDmoJrxt2l5kZI/1+AjEAvKzDQbC6Da9S+Zxuen8AHSPIYgFh
   vEvFwEeZNQoMd2FEAUoHUnBXAHX/vgcOmMvlMIICazCCAfKgAwIBAgIEKWsGWTAKBggq
   hkjOPQQDAjBtMRIwEAYKCZImiZPyLGQBGRYCY2ExGTAXBgoJkiaJk/IsZAEZFglzYW5k
   ZWxtYW4xPDA6BgNVBAMMM2ZvdW50YWluLXRlc3QuZXhhbXBsZS5jb20gVW5zdHJ1bmcg
   Rm91bnRhaW4gUm9vdCBDQTAeFw0yMDAyMjUyMTMxNDVaFw0yMjAyMjQyMTMxNDVaMG0x
   EjAQBgoJkiaJk/IsZAEZFgJjYTEZMBcGCgmSJomT8ixkARkWCXNhbmRlbG1hbjE8MDoG
   A1UEAwwzZm91bnRhaW4tdGVzdC5leGFtcGxlLmNvbSBVbnN0cnVuZyBGb3VudGFpbiBS
   b290IENBMHYwEAYHKoZIzj0CAQYFK4EEACIDYgAEG39ZuhfDGrxmjYxs4MP6MXEPZfYi
   kb23VoAH29+4eqVFTXmKRpcZWZYxsY9pfTK+PMg+W2O1Rvao1W+b+cDz07kUP3JHUaR2
   3dLwv/XKjzABf/jCX9P1IANEi4vyB2y0o2MwYTAPBgNVHRMBAf8EBTADAQH/MA4GA1Ud
   DwEB/wQEAwIBBjAdBgNVHQ4EFgQUuaX2yxHhB6RJLKcIxnwQvIezdCYwHgYDVR0jBBgw
   FoAUuaX2yxHhB6RJLKcIxnwQvIezdCYwCgYIKoZIzj0EAwIDZwAwZAIwIIMGzo2YpFR6
   ZkxKOnDCUjZaUo1ZfSCbKmkUWIc42FV53f0pOJUekZN2tPVmKUS0AjBvOPmvEu0w1YUp
   fLEWWL1nkUPEDTD52BysLwbdvNUGQiyEogTqAqRfF1Em+9kv0lwxggFLMIIBRwIBATB1
   MG0xEjAQBgoJkiaJk/IsZAEZFgJjYTEZMBcGCgmSJomT8ixkARkWCXNhbmRlbG1hbjE8
   MDoGA1UEAwwzZm91bnRhaW4tdGVzdC5leGFtcGxlLmNvbSBVbnN0cnVuZyBGb3VudGFp
   biBSb290IENBAgQ/mJtSMAsGCWCGSAFlAwQCAaBpMBgGCSqGSIb3DQEJAzELBgkqhkiG
   9w0BBwEwHAYJKoZIhvcNAQkFMQ8XDTIxMDQxMzIxNDMyM1owLwYJKoZIhvcNAQkEMSIE
   IEnOrdWj1G70K74IhCJ7UXi+wPS+r2C8DFEqjabGP+G8MAoGCCqGSM49BAMCBEcwRQIh
   AMhO3M+tSWb2wKTBOXPArN+XvjSzAhaQA/uLj3qhPwi/AiBDDthf6mjMuirqXE0yjMif
   C2UY9oNUFF9Nl0wEQpBBAA==
   <CODE ENDS>
```

Ниже представлено декодирование ASN1 артефакта из файла examples/parboiled_vr_00_D0-E5-02-00-2D.b64.

```
    0:d=0  hl=4 l=3939 cons: SEQUENCE
    4:d=1  hl=2 l=   9 prim: OBJECT            :pkcs7-signedData
   15:d=1  hl=4 l=3924 cons: cont [ 0 ]
   19:d=2  hl=4 l=3920 cons: SEQUENCE
   23:d=3  hl=2 l=   1 prim: INTEGER           :01
   26:d=3  hl=2 l=  13 cons: SET
   28:d=4  hl=2 l=  11 cons: SEQUENCE
   30:d=5  hl=2 l=   9 prim: OBJECT            :sha256
   41:d=3  hl=4 l=2424 cons: SEQUENCE
   45:d=4  hl=2 l=   9 prim: OBJECT            :pkcs7-data
   56:d=4  hl=4 l=2409 cons: cont [ 0 ]
   60:d=5  hl=4 l=2405 prim: OCTET STRING      :{"ietf-voucher-request:v
 2469:d=3  hl=4 l=1135 cons: cont [ 0 ]
 2473:d=4  hl=4 l= 508 cons: SEQUENCE
 2477:d=5  hl=4 l= 386 cons: SEQUENCE
 2481:d=6  hl=2 l=   3 cons: cont [ 0 ]
```

```
2483:d=7  hl=2 l=   1 prim: INTEGER           :02
2486:d=6  hl=2 l=   4 prim: INTEGER           :3F989B52
2492:d=6  hl=2 l=  10 cons: SEQUENCE
2494:d=7  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
2504:d=6  hl=2 l= 109 cons: SEQUENCE
2506:d=7  hl=2 l=  18 cons: SET
2508:d=8  hl=2 l=  16 cons: SEQUENCE
2510:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
2522:d=9  hl=2 l=   2 prim: IA5STRING         :ca
2526:d=7  hl=2 l=  25 cons: SET
2528:d=8  hl=2 l=  23 cons: SEQUENCE
2530:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
2542:d=9  hl=2 l=   9 prim: IA5STRING         :sandelman
2553:d=7  hl=2 l=  60 cons: SET
2555:d=8  hl=2 l=  58 cons: SEQUENCE
2557:d=9  hl=2 l=   3 prim: OBJECT            :commonName
2562:d=9  hl=2 l=  51 prim: UTF8STRING        :fountain-test.example.co
2615:d=6  hl=2 l=  30 cons: SEQUENCE
2617:d=7  hl=2 l=  13 prim: UTCTIME           :200225213154Z
2632:d=7  hl=2 l=  13 prim: UTCTIME           :220224213154Z
2647:d=6  hl=2 l=  83 cons: SEQUENCE
2649:d=7  hl=2 l=  18 cons: SET
2651:d=8  hl=2 l=  16 cons: SEQUENCE
2653:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
2665:d=9  hl=2 l=   2 prim: IA5STRING         :ca
2669:d=7  hl=2 l=  25 cons: SET
2671:d=8  hl=2 l=  23 cons: SEQUENCE
2673:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
2685:d=9  hl=2 l=   9 prim: IA5STRING         :sandelman
2696:d=7  hl=2 l=  34 cons: SET
2698:d=8  hl=2 l=  32 cons: SEQUENCE
2700:d=9  hl=2 l=   3 prim: OBJECT            :commonName
2705:d=9  hl=2 l=  25 prim: UTF8STRING        :fountain-test.example.co
2732:d=6  hl=2 l=  89 cons: SEQUENCE
2734:d=7  hl=2 l=  19 cons: SEQUENCE
2736:d=8  hl=2 l=   7 prim: OBJECT            :id-ecPublicKey
2745:d=8  hl=2 l=   8 prim: OBJECT            :prime256v1
2755:d=7  hl=2 l=  66 prim: BIT STRING
2823:d=6  hl=2 l=  42 cons: cont [ 3 ]
2825:d=7  hl=2 l=  40 cons: SEQUENCE
2827:d=8  hl=2 l=  22 cons: SEQUENCE
2829:d=9  hl=2 l=   3 prim: OBJECT            :X509v3 Extended Key Usag
2834:d=9  hl=2 l=   1 prim: BOOLEAN           :255
2837:d=9  hl=2 l=  12 prim: OCTET STRING      [HEX DUMP]:300A06082B0601
2851:d=8  hl=2 l=  14 cons: SEQUENCE
2853:d=9  hl=2 l=   3 prim: OBJECT            :X509v3 Key Usage
2858:d=9  hl=2 l=   1 prim: BOOLEAN           :255
2861:d=9  hl=2 l=   4 prim: OCTET STRING      [HEX DUMP]:03020780
2867:d=5  hl=2 l=  10 cons: SEQUENCE
2869:d=6  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
2879:d=5  hl=2 l= 104 prim: BIT STRING
2985:d=4  hl=4 l= 619 cons: SEQUENCE
2989:d=5  hl=4 l= 498 cons: SEQUENCE
2993:d=6  hl=2 l=   3 cons: cont [ 0 ]
2995:d=7  hl=2 l=   1 prim: INTEGER           :02
2998:d=6  hl=2 l=   4 prim: INTEGER           :296B0659
3004:d=6  hl=2 l=  10 cons: SEQUENCE
3006:d=7  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
3016:d=6  hl=2 l= 109 cons: SEQUENCE
3018:d=7  hl=2 l=  18 cons: SET
3020:d=8  hl=2 l=  16 cons: SEQUENCE
3022:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
3034:d=9  hl=2 l=   2 prim: IA5STRING         :ca
3038:d=7  hl=2 l=  25 cons: SET
3040:d=8  hl=2 l=  23 cons: SEQUENCE
3042:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
3054:d=9  hl=2 l=   9 prim: IA5STRING         :sandelman
3065:d=7  hl=2 l=  60 cons: SET
3067:d=8  hl=2 l=  58 cons: SEQUENCE
3069:d=9  hl=2 l=   3 prim: OBJECT            :commonName
3074:d=9  hl=2 l=  51 prim: UTF8STRING        :fountain-test.example.co
3127:d=6  hl=2 l=  30 cons: SEQUENCE
3129:d=7  hl=2 l=  13 prim: UTCTIME           :200225213145Z
3144:d=7  hl=2 l=  13 prim: UTCTIME           :220224213145Z
3159:d=6  hl=2 l= 109 cons: SEQUENCE
3161:d=7  hl=2 l=  18 cons: SET
3163:d=8  hl=2 l=  16 cons: SEQUENCE
3165:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
3177:d=9  hl=2 l=   2 prim: IA5STRING         :ca
3181:d=7  hl=2 l=  25 cons: SET
3183:d=8  hl=2 l=  23 cons: SEQUENCE
3185:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
3197:d=9  hl=2 l=   9 prim: IA5STRING         :sandelman
3208:d=7  hl=2 l=  60 cons: SET
3210:d=8  hl=2 l=  58 cons: SEQUENCE
3212:d=9  hl=2 l=   3 prim: OBJECT            :commonName
```

```
3217:d=9  hl=2 l=  51 prim: UTF8STRING        :fountain-test.example.co
3270:d=6  hl=2 l= 118 cons: SEQUENCE
3272:d=7  hl=2 l=  16 cons: SEQUENCE
3274:d=8  hl=2 l=   7 prim: OBJECT            :id-ecPublicKey
3283:d=8  hl=2 l=   5 prim: OBJECT            :secp384r1
3290:d=7  hl=2 l=  98 prim: BIT STRING
3390:d=6  hl=2 l=  99 cons: cont [ 3 ]
3392:d=7  hl=2 l=  97 cons: SEQUENCE
3394:d=8  hl=2 l=  15 cons: SEQUENCE
3396:d=9  hl=2 l=   3 prim: OBJECT            :X509v3 Basic Constraints
3401:d=9  hl=2 l=   1 prim: BOOLEAN           :255
3404:d=9  hl=2 l=   5 prim: OCTET STRING      [HEX DUMP]:30030101FF
3411:d=8  hl=2 l=  14 cons: SEQUENCE
3413:d=9  hl=2 l=   3 prim: OBJECT            :X509v3 Key Usage
3418:d=9  hl=2 l=   1 prim: BOOLEAN           :255
3421:d=9  hl=2 l=   4 prim: OCTET STRING      [HEX DUMP]:03020106
3427:d=8  hl=2 l=  29 cons: SEQUENCE
3429:d=9  hl=2 l=   3 prim: OBJECT            :X509v3 Subject Key Ident
3434:d=9  hl=2 l=  22 prim: OCTET STRING      [HEX DUMP]:0414B9A5F6CB11
3458:d=8  hl=2 l=  31 cons: SEQUENCE
3460:d=9  hl=2 l=   3 prim: OBJECT            :X509v3 Authority Key Ide
3465:d=9  hl=2 l=  24 prim: OCTET STRING      [HEX DUMP]:30168014B9A5F6
3491:d=5  hl=2 l=  10 cons: SEQUENCE
3493:d=6  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
3503:d=5  hl=2 l= 103 prim: BIT STRING
3608:d=3  hl=4 l= 331 cons: SET
3612:d=4  hl=4 l= 327 cons: SEQUENCE
3616:d=5  hl=2 l=   1 prim: INTEGER           :01
3619:d=5  hl=2 l= 117 cons: SEQUENCE
3621:d=6  hl=2 l= 109 cons: SEQUENCE
3623:d=7  hl=2 l=  18 cons: SET
3625:d=8  hl=2 l=  16 cons: SEQUENCE
3627:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
3639:d=9  hl=2 l=   2 prim: IA5STRING         :ca
3643:d=7  hl=2 l=  25 cons: SET
3645:d=8  hl=2 l=  23 cons: SEQUENCE
3647:d=9  hl=2 l=  10 prim: OBJECT            :domainComponent
3659:d=9  hl=2 l=   9 prim: IA5STRING         :sandelman
3670:d=7  hl=2 l=  60 cons: SET
3672:d=8  hl=2 l=  58 cons: SEQUENCE
3674:d=9  hl=2 l=   3 prim: OBJECT            :commonName
3679:d=9  hl=2 l=  51 prim: UTF8STRING        :fountain-test.example.co
3732:d=6  hl=2 l=   4 prim: INTEGER           :3F989B52
3738:d=5  hl=2 l=  11 cons: SEQUENCE
3740:d=6  hl=2 l=   9 prim: OBJECT            :sha256
3751:d=5  hl=2 l= 105 cons: cont [ 0 ]
3753:d=6  hl=2 l=  24 cons: SEQUENCE
3755:d=7  hl=2 l=   9 prim: OBJECT            :contentType
3766:d=7  hl=2 l=  11 cons: SET
3768:d=8  hl=2 l=   9 prim: OBJECT            :pkcs7-data
3779:d=6  hl=2 l=  28 cons: SEQUENCE
3781:d=7  hl=2 l=   9 prim: OBJECT            :signingTime
3792:d=7  hl=2 l=  15 cons: SET
3794:d=8  hl=2 l=  13 prim: UTCTIME           :210413214323Z
3809:d=6  hl=2 l=  47 cons: SEQUENCE
3811:d=7  hl=2 l=   9 prim: OBJECT            :messageDigest
3822:d=7  hl=2 l=  34 cons: SET
3824:d=8  hl=2 l=  32 prim: OCTET STRING      [HEX DUMP]:49CEADD5A3946E
3858:d=5  hl=2 l=  10 cons: SEQUENCE
3860:d=6  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
3870:d=5  hl=2 l=  71 prim: OCTET STRING      [HEX DUMP]:3045022100C84E
```

Код JSON, содержащийся в voucher-request, приведён ниже. Отметим, что предыдущий voucher-request находится в атрибуте prior-signed-voucher-request.

```
{"ietf-voucher-request:voucher":{"assertion":"proximity","cr
eated-on":"2021-04-13T21:43:23.787Z","serial-number":"00-D0-
E5-F2-00-02","nonce":"-_XE9zK9q8Ll1qylMtLKeg","prior-signed-
voucher-request":"MIIGcAYJKoZIhvcNAQcCoIIGYTCCBl0CAQExDTALBg
lghkgBZQMEAgEwggOJBgkqhkiG9w0BBwGgggN6BIIDdnsiaWV0Zi12b3VjaG
VyLXJlcXVlc3Q6dm91Y2hlciI6eyJhc3NlcnRpb24iOiJwcm94aW1pdHkiLC
JjcmVhdGVkLW9uIjoiMjAyMS0wNC0xM1QxNzo0MzoyMy43NDctMDQ6MDAiLC
JzZXJpYWwtbnVtYmVyIjoiMDAtRDAtRTUtRjItMDAtMDIiLCJub25jZSI6Ii
1fWEU5eks5cThMbDFxeWxNdExLZWciLCJwcm94aW1pdHktcmVnaXN0cmFyLW
NlcnQiOiJNSUlCL0RDQ0FZS2dBd0lCQWdJRVA1aWJVakFLQmdncWhrak9QUV
FEQWpCdE1SSXdFQVlLQ1pJbWlaUHlMR1FCR1JZQ1kyRXhHVEFYQmdvSmtpYU
prL0lzWkFFWkZnbHpZVzVrWld4dFlXNHhQREE2QmdOVkJBTU1NMlp2ZFc1MF
lXbHVMWFJsYzNRdVpYaGhiWEJzWlM1amIyMGdWVzV6ZEhKMWJtY2dSbTkxYm
5SaGFXNGdVbTl2ZENCRFFUQWVGdzB5TURBeU1qVXlNVE14TlRSYUZ3MHlNak
F5TWpReU1UTXhOVFJhTUZNeEVqQVFCZ29Ka2lhSmsvSXNaQUVaRmdKallURV
pNQmNHQ2dtU0pvbVQ4aXhrQVJrV0NYTmhibVJsYkcxaGJqRWlNQ0FHQTFVRU
F3d1pabTkxYm5SaGFXNHRkR1Z6ZEM1bGVHRnRjR3hsTG1OdmJUQlpNQk1HQn
lxR1NNNDlBZ0VHQ0NxR1NNNDlBd0VIQTBJQUJKWmxVSEkwdXAvbDNlWmY5dk
NCYitsSW5vRU1FZ2M3Um8rWFpDdGpBSTBDRDFmSmZKUi9oSXl5RG1IV3lZaU
5GYlJDSDlmeWFyZmt6Z1g0cDB6VGl6cWpLakFvTUJZR0ExVWRKUUVCL3dRTU
1Bb0dDQ3NHQVFVRkJ3TWNNQTRHQTFVZER3RUIvd1FFQXdJSGdEQUtCZ2dxaG
tqT1BRUURBZ05vQURCbEFqQm1UMkJNVlVnZWxnZjQzUis1eUJLTlJUYUhteV
```

```
BBdkx2eHl6MG1GVlp2WHgrLzFSd09hZ212RzNhWG1Sa2ovWDRDTVFDOHJNTk
JzTG9OcjFMNW5HNTZmd0FkSThoaUFXRzhTOFhBUjVrMUNneDNZVVFCU2dkU2
NGY0FkZisrQnc2WXkrVT0ifX2gggGyMIIBrjCCATWgAwIBAgIEDYOv2TAKBg
gqhkjOPQQDAjAmMSQwIgYDVQQDDBtoaWdod2F5LXRlc3QuZXhhbXBsZS5jb2
0gQ0EwIBcNMjEwNDEzMjAzNzM5WhgPMjk5OTEyMzEwMDAwMDBaMBwxGjAYBg
NVBAUMETAwLUQwLUU1LUYyLTAwLTAyMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQ
cDQgAEA6N1Q4ezfMAKmoecrfb0OBMc1AyEH+BATkF58FsTSyBxs0SbSWLxFj
DOuwB9gLGn2TsTUJumJ6VPw5Z/TP4hJ6NZMFcwHQYDVR0OBBYEFEWIzJaWAG
Q3sLojZWRkVAgGbFatMAkGA1UdEwQCMAAwKwYIKwYBBQUHASAEHxYdaGlnaH
dheS10ZXN0LmV4YW1wbGUuY29tOjk0NDMwCgYIKoZIzj0EAwIDZwAwZAIwTm
lG8sXkKGNbwbKQcYMapFbmSbnHHURFUoFuRqvbgYX7FlXpBczfwF2kllNuuj
igAjAow1kc4r55EmiH+OMEXjBNlWlBSZC5QuJjEf0Jsmxssc+pucjOJ4Shqn
exMEy7bjAxggEEMIIBAAIBATAuMCYxJDAiBgNVBAMMG2hpZ2h3YXktdGVzdC
5leGFtcGxlLmNvbSBDQQIEDYOv2TALBglghkgBZQMEAgGgaTAYBgkqhkiG9w
0BCQMxCwYJKoZIhvcNAQcBMBwGCSqGSIb3DQEJBTEPFw0yMTA0MTMyMTQzMj
NaMC8GCSqGSIb3DQEJBDEiBCBJwhyYibIjeqeR3bOaLURzMlGrc3F2X+kvJl
errtoCtTAKBggqhkjOPQQDAgRHMEUCIQCmYuCE61HFQXH/E16GDOCsVquDtg
r+Q/6/Du/9QkzA7gIgf7MFhAIPW2PNwRa2vZFQAKXUbimkiHKzXBA8md0VHb
U="}}
```

C.2.3. От MASA к регистратору

Агент MASA возвращает регистратору ваучер, который транслируется заявителю.

```
<CODE BEGINS> file "voucher_00-D0-E5-F2-00-02.b64"
MIIGIgYJKoZIhvcNAQcCoIIGEzCCBg8CAQExDTALBglghkgBZQMEAgEwggN4BgkqhkiG
9w0BBwGgggNpBIIDZXsiaWV0Zi12b3VjaGVyOnZvdWNoZXIiOnsiYXNzZXJ0aW9uIjoi
bG9nZ2VkIiwiY3JlYXRlZC1vbiI6IjIwMjEtMDQtMTNUMTc6NDM6MjQuNTg5LTA0OjAw
Iiwic2VyaWFsLW51bWJlciI6IjAwLUQwLUU1LUYyLTAwLTAyIiwibm9uY2UiOiItX1hF
OXpLOXE4TGwxcXlsTXRMS2VnIiwicGlubmVkLWRvbWFpbi1jZXJ0IjoiTUlJQi9EQ0NB
WUtnQXdJQkFnSUVQNWliVWpBS0JnZ3Foa2pPUFFRREFqQnRNUkl3RUFZS0NaSW1pWlB5
TEdRQkdSWUNZMkV4R1RBWEJnb0praWFKay9Jc1pBRVpGZ2x6WVc1a1pXeHRZVzR4UERB
NkJnTlZCQU1NTTJadmRXNTBZV2x1TFhSbGMzUXVaWGhoYlhCc1pTNWpiMjBnVlc1emRI
SjFibWNnUm05MWJuUmhhVzRnVW05dmRDQkRRVEFlRncweU1EQXlNalV5TVRNeE5UUmFG
dzB5TWpBeU1qUXlNVE14TlRSYU1GTXhFakFRQmdvSmtpYUprL0lzWkFFWkZnSmpZVEVa
TUJjR0NnbVNKb21UOGl4a0FSa1dDWE5oYm1SbGJHMWhiakVpTUNBR0ExVUVBd3daWm05
MWJuUmhhVzR0ZEdWemRDNWxlR0Z0Y0d4bExtTnZiVEJaTUJNR0J5cUdTTTQ5QWdFR0ND
cUdTTTQ5QXdFSEEwSUFCSlpsVUhJMHVwL2wzZVpmOXZDQmIrbElub0VNRWdjN1JvK1ha
Q3RqQUkwQ0QxZkpmSlIvaEl5eURtSFd5WWlORmJSQ0g5ZnlhcmZremdYNHAwelRpenFq
S2pBb01CWUdBMVVkSlFFQi93UU1NQW9HQ0NzR0FRVUZCd01jTUE0R0ExVWREd0VCL3dR
RUF3SUhnREFLQmdncWhrak9QUVFEQWdOb0FEQmxBakJtVDJCTVZVZ2VsZ2Y0M1IrNXlC
S05SVGFIbXlQQXZMdnh5ejBtRlZadlh4Ky8xUndPYWdtdkczYVhtUmtqL1g0Q01RQzhy
TU5Cc0xvTnIxTDVuRzU2ZndBZEk4aGlBV0c4UzhYQVI1azFDZ3gzWVVRQlNnZFNjRmNB
ZGYrK0J3Nll5K1U9In19oIIBdDCCAXAwgfagAwIBAgIEC4cKMTAKBggqhkjOPQQDAjAm
MSQwIgYDVQQDDBtoaWdod2F5LXRlc3QuZXhhbXBsZS5jb20gQ0EwHhcNMjEwNDEzMjE0
MDE2WhcNMjMwNDEzMjE0MDE2WjAoMSYwJAYDVQQDDB1oaWdod2F5LXRlc3QuZXhhbXBs
ZS5jb20gTUFTQTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABKoEFaNEueJE+Mn5Gwcb
pnRznB66bKmzqTCpojJZ96AdRwFtuTCVfoKouLTBX0idIhMLfJLM31lyuKy4CUtpp6Wj
EDAOMAwGA1UdEwEB/wQCMAAwCgYIKoZIzj0EAwIDaQAwZgIxAK7LYS3UXIluhqoLBh3G
02C6MnM2JdMjhUmHHM6UI3kankFVJB0VIqFIuwrAqzwTcwIxAIY8Z7OVouLl+a35HZzB
NDJ49c/q1UcDnwC/0FnLUcKYBIEkilETULF1si+dqLT0uTGCAQUwggEBAgEBMC4wJjEk
MCIGA1UEAwwbaGlnaHdheS10ZXN0LmV4YW1wbGUuY29tIENBAgQLhwoxMAsGCWCGSAFl
AwQCAaBpMBgGCSqGSIb3DQEJAzELBgkqhkiG9w0BBwEwHAYJKoZIhvcNAQkFMQ8XDTIx
MDQxMzIxNDMyNFowLwYJKoZIhvcNAQkEMSIEIFUUjg4WYVO+MpX122Qfk/7zm/G6/B59
HD/xrVR0lGIjMAoGCCqGSM49BAMCBEgwRgIhAOhUfxbH2dwpB2BrTDcsYSjRkCCk/WE6
Mdt+y4z5KD9IAiEAphwdIUb40A0noNIUpH7N2lTyAFZgyn1lNHTteY9DmYI=
<CODE ENDS>
```

Ниже представлено декодирование ASN1 для артефакта из файла examples/voucher_00-D0-E5-F2-00-02.b64

```
  0:d=0  hl=4 l=1570 cons: SEQUENCE
  4:d=1  hl=2 l=   9 prim: OBJECT            :pkcs7-signedData
 15:d=1  hl=4 l=1555 cons: cont [ 0 ]
 19:d=2  hl=4 l=1551 cons: SEQUENCE
 23:d=3  hl=2 l=   1 prim: INTEGER           :01
 26:d=3  hl=2 l=  13 cons: SET
 28:d=4  hl=2 l=  11 cons: SEQUENCE
 30:d=5  hl=2 l=   9 prim: OBJECT            :sha256
 41:d=3  hl=4 l= 888 cons: SEQUENCE
 45:d=4  hl=2 l=   9 prim: OBJECT            :pkcs7-data
 56:d=4  hl=4 l= 873 cons: cont [ 0 ]
 60:d=5  hl=4 l= 869 prim: OCTET STRING      :{"ietf-voucher:voucher":
933:d=3  hl=4 l= 372 cons: cont [ 0 ]
937:d=4  hl=4 l= 368 cons: SEQUENCE
941:d=5  hl=3 l= 246 cons: SEQUENCE
944:d=6  hl=2 l=   3 cons: cont [ 0 ]
946:d=7  hl=2 l=   1 prim: INTEGER           :02
949:d=6  hl=2 l=   4 prim: INTEGER           :0B870A31
955:d=6  hl=2 l=  10 cons: SEQUENCE
957:d=7  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
967:d=6  hl=2 l=  38 cons: SEQUENCE
969:d=7  hl=2 l=  36 cons: SET
971:d=8  hl=2 l=  34 cons: SEQUENCE
973:d=9  hl=2 l=   3 prim: OBJECT            :commonName
978:d=9  hl=2 l=  27 prim: UTF8STRING        :highway-test.example.com
1007:d=6  hl=2 l=  30 cons: SEQUENCE
1009:d=7  hl=2 l=  13 prim: UTCTIME           :210413214016Z
1024:d=7  hl=2 l=  13 prim: UTCTIME           :230413214016Z
```

```
1039:d=6  hl=2 l=  40 cons: SEQUENCE
1041:d=7  hl=2 l=  38 cons: SET
1043:d=8  hl=2 l=  36 cons: SEQUENCE
1045:d=9  hl=2 l=   3 prim: OBJECT            :commonName
1050:d=9  hl=2 l=  29 prim: UTF8STRING        :highway-test.example.com
1081:d=6  hl=2 l=  89 cons: SEQUENCE
1083:d=7  hl=2 l=  19 cons: SEQUENCE
1085:d=8  hl=2 l=   7 prim: OBJECT            :id-ecPublicKey
1094:d=8  hl=2 l=   8 prim: OBJECT            :prime256v1
1104:d=7  hl=2 l=  66 prim: BIT STRING
1172:d=6  hl=2 l=  16 cons: cont [ 3 ]
1174:d=7  hl=2 l=  14 cons: SEQUENCE
1176:d=8  hl=2 l=  12 cons: SEQUENCE
1178:d=9  hl=2 l=   3 prim: OBJECT            :X509v3 Basic Constraints
1183:d=9  hl=2 l=   1 prim: BOOLEAN           :255
1186:d=9  hl=2 l=   2 prim: OCTET STRING      [HEX DUMP]:3000
1190:d=5  hl=2 l=  10 cons: SEQUENCE
1192:d=6  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
1202:d=5  hl=2 l= 105 prim: BIT STRING
1309:d=3  hl=4 l= 261 cons: SET
1313:d=4  hl=4 l= 257 cons: SEQUENCE
1317:d=5  hl=2 l=   1 prim: INTEGER           :01
1320:d=5  hl=2 l=  46 cons: SEQUENCE
1322:d=6  hl=2 l=  38 cons: SEQUENCE
1324:d=7  hl=2 l=  36 cons: SET
1326:d=8  hl=2 l=  34 cons: SEQUENCE
1328:d=9  hl=2 l=   3 prim: OBJECT            :commonName
1333:d=9  hl=2 l=  27 prim: UTF8STRING        :highway-test.example.com
1362:d=6  hl=2 l=   4 prim: INTEGER           :0B870A31
1368:d=5  hl=2 l=  11 cons: SEQUENCE
1370:d=6  hl=2 l=   9 prim: OBJECT            :sha256
1381:d=5  hl=2 l= 105 cons: cont [ 0 ]
1383:d=6  hl=2 l=  24 cons: SEQUENCE
1385:d=7  hl=2 l=   9 prim: OBJECT            :contentType
1396:d=7  hl=2 l=  11 cons: SET
1398:d=8  hl=2 l=   9 prim: OBJECT            :pkcs7-data
1409:d=6  hl=2 l=  28 cons: SEQUENCE
1411:d=7  hl=2 l=   9 prim: OBJECT            :signingTime
1422:d=7  hl=2 l=  15 cons: SET
1424:d=8  hl=2 l=  13 prim: UTCTIME           :210413214324Z
1439:d=6  hl=2 l=  47 cons: SEQUENCE
1441:d=7  hl=2 l=   9 prim: OBJECT            :messageDigest
1452:d=7  hl=2 l=  34 cons: SET
1454:d=8  hl=2 l=  32 prim: OCTET STRING      [HEX DUMP]:55148E0E166153
1488:d=5  hl=2 l=  10 cons: SEQUENCE
1490:d=6  hl=2 l=   8 prim: OBJECT            :ecdsa-with-SHA256
1500:d=5  hl=2 l=  72 prim: OCTET STRING      [HEX DUMP]:3046022100E854
```

Благодарности

Спасибо рецензентам за их вклад, в частности, William Atwood, Brian Carpenter, Fuyu Eleven, Eliot Lear, Sergey Kasatkin, Anoop Kumar, Tom Petch, Markus Stenberg, Peter van der Stok, Thomas Werner.

Важные обзоры представили Jari Arkko, Christian Huitema, Russ Housley.

Henk Birkholz представил CDDL для отклика audit-log.

Этот документ начался с двухстраничного изложения идей от Steinthor Bjarnason.

Кроме того, многие члены IESG, включая Adam Roach, Alexey Melnikov, Alissa Cooper, Benjamin Kaduk, Éric Vyncke, Roman Danyliw, Magnus Westerlund, представили существенные обзорные комментации.

Адреса авторов

Max Pritikin
Cisco
Email: pritikin@cisco.com

Michael C. Richardson
Sandelman Software Works
Email: mcr+ietf@sandelman.ca
URI: http://www.sandelman.ca/

Toerless Eckert
Futurewei Technologies Inc. USA
2330 Central Expy
Santa Clara, CA 95050
United States of America
Email: tte+ietf@cs.fau.de

Michael H. Behringer
Email: Michael.H.Behringer@gmail.com

Kent Watsen
Watsen Networks
Email: kent+ietf@watsen.net

Перевод на русский язык

Николай Малых
nmalykh@protokols.ru