Экономический и Социальный Совет

Distr.: General
3 December 2020
Russian
Original: English

Европейская экономическая комиссия

Комитет по внутреннему транспорту

Рабочая группа по таможенным вопросам, связанным с транспортом

Группа экспертов по концептуальным и техническим аспектам компьютеризации процедуры МДП

Первая сессия
Женева, 27–29 января 2021 года
Пункт 6 a) предварительной повестки дня
Международная система eTIR:
Доклад о ходе разработки международной системы eTIR

Веб-службы eTIR — Глоссарий и технические процедуры*

Записка секретариата

I. Введение — мандат

1. Комитет по внутреннему транспорту на своей восемьдесят второй сессии (23−28 февраля 2020 года) одобрил (ECE/TRANS/294, пункт 84[1]) учреждение Группы экспертов по концептуальным и техническим аспектам компьютеризации процедуры МДП (WP.30/GE.1) и положение о ее КВ[2] (ECE/TRANS/WP.30/2019/9 и ECE/TRANS/WP.30/2019/9/Corr.1) в ожидании утверждения Исполнительным комитетом ЕЭК ООН (Исполкомом). Исполком на своем дистанционном неофициальном совещании 20 мая 2020 года одобрил учреждение Группы экспертов по концептуальным и техническим аспектам компьютеризации процедуры МДП (WP.30/GE.1) до 2022 года на основе положения о круге ведения, содержащегося в документе ECE/TRANS/WP.30/2019/9 и Corr.1, как указано в документе ECE/TRANS/294 (ECE/EX/2020/L.2, пункт 5 b))[3].

* Настоящий документ был представлен для обработки с опозданием, поскольку для получения санкции на его окончательную доработку потребовалось больше времени, чем предполагалось.

[1] Решение Комитета по внутреннему транспорту, пункт 84/ECE/TRANS/294 http://www.unece.org/fileadmin/DAM/trans/doc/2020/itc/ECE-TRANS-294r.pdf.

[2] Круг ведения недавно созданной Группы, утвержденный Комитетом по внутреннему транспорту и Исполнительным комитетом (Исполкомом) ЕЭК ООН http://www.unece.org/fileadmin/DAM/trans/bcf/wp30/documents/2019/ECE-TRANS-WP30-2019-09r.pdf и исправление.

[3] Решение Исполнительного комитета, ECE/EX/2020/L.2/п. 5 b) http://www.unece.org/fileadmin/DAM/commission/EXCOM/Agenda/2020/Remote_informal_mtg_20_05_2020/Item_4_ECE_EX_2020_L.2_ITC_Sub_bodies_E.pdf.

2. В соответствии с положением о круге ведения Группе следует сосредоточить свои усилия на подготовке новой версии спецификаций eTIR в ожидании официального создания ТОО. В частности, Группе следует: a) подготовить новый вариант технических спецификаций процедуры eTIR и поправки к ним для обеспечения их соответствия функциональным спецификациям процедуры eTIR; b) подготовить новый вариант функциональных спецификаций процедуры eTIR и поправки к ним для обеспечения их соответствия концептуальным спецификациям процедуры eTIR; c) подготовка поправок к концептуальным спецификациям процедуры eTIR по просьбе WP.30.

3. В настоящем документе представлены глоссарий и технические процедуры, описывающие порядок создание сертификата X.509 и проверку подключения к веб-службам eTIR. Настоящий документ действителен для версии 1.0 международной системы eTIR, которая была разработана на основе спецификаций eTIR (версия 4.3a).

II. Резервные процедуры

4. Резервные функциональные процедуры eTIR в настоящее время описаны в утвержденных поправках к концептуальной, функциональной и технической документации eTIR — v.4.2a, которые будут включены в следующую версию спецификаций eTIR (v4.3). Кроме того, каждое сообщение может включать техническую резервную процедуру, которая будет описана в соответствующих руководствах по реализации сообщений eTIR.

III. Поддержка и контактная информация

5. Просьба иметь в виду, что в контексте проектов по подключению, осуществляемых таможенными администрациями, секретариат МДП готов оказать содействие договаривающимся сторонам при подключении их национальных таможенных систем к международной системе eTIR. Кроме того, в случае возникновения вопросов или проблем, касающихся настоящего документа или международной системы eTIR, можно воспользоваться приведенными ниже контактными данными (желательно использовать электронную почту).

Организация	*United Nations Economic Commission For Europe* *TIR secretariat* *Palais des Nations,* *1211 Geneva 10, Switzerland*
Контактная информация:	Эл. почта: etir@un.org Телефон: +41 (0) 22 917 55 06

IV. Глоссарий

API

Интерфейс прикладного программирования (API) — программный интерфейс, который используется для доступа к соответствующему приложению или услуге из какой-либо программы.

Асимметричный алгоритм шифрования

Криптографическая система, использующая два ключа: открытый ключ, известный всем, и частный (или секретный) ключ, известный только владельцу спаренных ключей. Например, когда Алиса желает отправить защищенное сообщение Бобу, она использует открытый ключ Боба для шифрования этого сообщения. Затем для расшифровки этого сообщения Боб использует свой персональный ключ. RSA — пример асимметричного алгоритма.

Аутентификация

Аутентификация представляет собой процесс проверки или тестирования, позволяющий удостовериться в том, что идентичность субъекта, который обратился с запросом, является подлинной. Аутентификация предполагает, что данный субъект должен представить дополнительную информацию, подтверждающую подлинность заявленной им идентичности. Наиболее распространенной формой аутентификации является использование соответствующего пароля (она включает вариации паролей персональных идентификационных номеров (PIN) и парольных фраз). Аутентификация предполагает проверку идентичности данного субъекта, сравнивая один или несколько факторов с базой данных действительных идентичностей (т. е. учетных записей пользователей).

B2B

Отношения между коммерческими структурами (когда коммерческие структуры относятся к частному сектору).

B2C

Отношения между коммерческими структурами и таможенными органами (когда коммерческие структуры относятся к частному сектору).

C2B

Отношения между таможенными органами и коммерческими структурами (когда коммерческие структуры относятся к частному сектору).

C2C

Отношения между таможнями.

Сертификационный орган (СО)

Сертификационный орган или СО занимает доверенное положение, поскольку выдаваемый им сертификат связывает идентичность того или иного лица или предприятия с парой открытых или закрытых ключей (асимметричная криптография), которые используются для обеспечения безопасности большинства транзакций, осуществляемых по Интернету. Например, когда какое-либо предприятие или лицо желает использовать эти технологии, оно обращается в СО с просьбой выдать ему соответствующий сертификат. Прежде чем выдать сертификат этому лицу или предприятию, которое должен сертифицировать СО, он собирает о нем необходимую информацию.

Конфиденциальность

Конфиденциальность — это концепция мер, используемых для обеспечения защиты секретности данных, объектов или ресурсов. Целью защиты конфиденциальности является предотвращение или минимизация несанкционированного доступа к данным. Конфиденциальность сосредоточена на мерах безопасности с целью обеспечить такое положение, при котором никто, кроме предполагаемого получателя сообщения, не сможет его получить или прочитать. Защита конфиденциальности дает возможность уполномоченным пользователям получать доступ к ресурсам и воздействовать на них, но активно предотвращает это воздействие со стороны не уполномоченных пользователей.

Электронно-цифровая подпись

Цифровой код, который может прилагаться к электронному сообщению, преследует две различные цели: 1) сообщения с цифровой подписью гарантируют получателю тот факт, что данное сообщение действительно пришло от заявленного отправителя. Они исключают невозможность отказа (т. е. не позволяют отправителю в последствии утверждать, что данное сообщение — это подделка) и 2): сообщения с цифровой

подписью гарантируют получателю тот факт, что данное сообщение не было изменено во время его передачи от отправителя получателю по каналу связи. Это предохраняет сообщение как от злонамеренного изменения (когда какая-либо третья сторона изменяет смысл сообщения), так и от непреднамеренного изменения (по причине сбоев в процессе передачи данных, например в случае электрических помех).

Среда

Программное обеспечение разрабатывается и поддерживается в нескольких средах: среда разработки используется для реализации части программного обеспечения, среда пользовательских приемочных испытаний используется для тестирования и его проверки с другими заинтересованными сторонами, а эксплуатационная среда используется для эксплуатации системы, когда она является «интерактивной» и доступной для конечных пользователей в качестве своего рода услуги.

Ошибка

Ошибка — это серьезный сбой в подтверждении данных, который приведет к тому, что сообщение будет отклонено.

eTIR IS

Сокращение, которое иногда используется в диаграммах для обозначения международной системы eTIR.

Участник eTIR

Субъект, которые является частью системы eTIR и пользуется процедурой eTIR, описанной в приложении 11 к Конвенции МДП. Участник eTIR использует свои информационные системы в качестве части системы eTIR и может быть любым из следующих субъектов:

- ЕЭК ООН с международной системой eTIR;
- МСАТ, представляющий гарантийную цепь, со своими информационными системами;
- Таможенные органы со своими национальными таможенными системами;
- Держатели с их информационными системами.

Хэш

Хэш-значение (или просто хэш), также называемое профилем сообщения, представляет собой определенное значение, генерируемое из соответствующего текста. Хэш значительно меньше самого текста и генерируется по соответствующей формуле таким образом, что вероятность выдачи такого же значения хэша каким-нибудь другим текстом чрезвычайно мала.

Целостность

Целостность представляет собой концепцию защиты надежности и правильности данных. Защита целостности предотвращает несанкционированное изменение данных. Она гарантирует, что данные останутся правильными, неизменными и сохраненными. Надлежащим образом организованная защита целостности предусматривает соответствующие средства, допускающие внесение санкционированных изменений в условиях одновременной защиты от преднамеренных и злонамеренных несанкционированных действий (таких, как вирусы и взломы), а также от ошибок, допущенных уполномоченными пользователями (таких, как обычные ошибки или просмотры).

МБДМДП

Международный банк данных МДП является хранилищем всех данных о держателях книжек МДП и таможенных органах ЕЭК ООН. Данные в этой информационной

системе поддерживаются национальными объединениями и таможенными органами системы МДП.

Хранилище ключей

Хранилище ключей представляет собой соответствующую базу данных ключей и используется для хранения сертификатов, в том числе сертификатов всех доверенных сторон, в целях их использования соответствующей программой. С помощью хранилища ключей тот или иной уполномоченный субъект может проверить аутентичность других сторон, а также подтвердить свою аутентичность перед другими сторонами.

Исключение возможности отказа

Исключение возможности отказа гарантирует, что субъект какого-либо действия или лицо, по причине которого произошло какое-то событие, не может отрицать, что данное событие произошло. Исключение возможности отказа не позволяет тому или иному субъекту утверждать, что он не отправил сообщение, не выполнил какое-либо действие или не стал причиной какого-либо события. Такая возможность существует благодаря идентификации, аутентификации, санкционированию, подотчетности и аудиту. Исключение возможности отказа может быть установлено с помощью цифровых сертификатов, идентификаторов сессий, журналов транзакций и многочисленных других механизмов контроля за транзакциями и доступом.

ОРССИ

Организация по развитию стандартов структурированной информации (ОРССИ) является некоммерческим международным консорциумом, цель которого состоит в содействии принятию стандартов, не зависящих от конкретной продукции.

ИПК

Инфраструктура сертификации открытых ключей (ИПК) — это инфраструктура, необходимая для поддержки асимметричной криптографии.

Получатель

В контексте данного документа и всех документов, связанных с парами сообщений, «получателем» является информационная система соответствующего субъекта eTIR, которая получает сообщение, отправленное другим субъектом eTIR, и обрабатывает его.

RSA

Алгоритм RSA был изобретен Рональдом Л. Ривестом, Ади Шамиром и Леонардом Адлеманом в 1977 году. Это асимметричный алгоритм, который использует два разных ключа с математической зависимостью друг от друга. Открытый и закрытый ключи тщательно генерируются с использованием алгоритма RSA; их можно использовать для шифрования информации или ее подписания.

Отправитель

В контексте данного документа и всех документов, связанных с парами сообщений, «отправителем» является информационная система соответствующего субъекта eTIR, которая подготавливает и отправляет сообщение eTIR другому субъекту eTIR.

SOAP

Простой протокол доступа к объектам (SOAP) — это спецификация протокола сообщений в целях обмена информацией в ходе оказания соответствующих веб-услуг. Он представляет собой протокол на основе стандарта XML, состоящий из трех частей:

оболочка, определяющая структуру сообщения (хедер и текстовая часть) и способ его обработки;

набор правил кодирования для описания таких типов данных, которые определены приложениями;

условные обозначения для отображения процедурных вызовов и ответов.

Маркер

Маркер (иногда называемый маркером безопасности) — это объект, который контролирует доступ к цифровому активу. Традиционно этот термин используется для описания соответствующего аппаратного аутентификатора — небольшого устройства, используемого в сетевой среде, — для создания одноразового пароля, который вводится владельцем в окно входа в систему вместе с идентификатором и PIN-кодом. Однако в случае веб-служб и в связи с растущей потребностью в устройствах и процессах взаимной аутентификации по открытым сетям термин «маркер» был расширен и сейчас включает в себя также соответствующие механизмы программного обеспечения. Маркер может быть сертификатом X.509, который, например, увязывает идентичность с открытым ключом.

Хранилище доверенных сертификатов

Хранилище доверенных сертификатов представляет собой соответствующий файл хранилища ключей, в котором хранятся сертификаты других сторон, с которыми вы намерены связаться, или сертификационного органа, которому вы доверяете в плане идентификации других сторон.

Веб-служба

Веб-служба, по своему определению, представляет собой сетевую коммуникацию между двумя приложениями (а не между человеком и, например, приложением). Межмашинная коммуникация — это еще один термин для определения этого вида связи.

Безопасность веб-служб (ВС-безопасность)

Спецификация «безопасность веб-служб» (ВС-безопасность) описывает усовершенствованные версии SOAP 1.1, повышающие защиту (целостность) и конфиденциальность сообщений. Эти усовершенствования включают в себя функциональные параметры обеспечения безопасности сообщений SOAP с помощью цифровой подписи XML, конфиденциальности с помощью шифрования XML и расширения учетных данных с помощью маркеров безопасности (например, маркер X.509).

WSDL

Язык описания веб-сервисов (WSDL) — это язык описания интерфейса на базе XML, который используется для описания функциональных параметров, предлагаемых веб-службами.

Цифровой сертификат X.509

В общем случае сертификат — это документ, выданный каким-либо органом в целях подтверждения истины или предоставления определенных доказательств. Цифровой сертификат обычно используется в целях представления доказательства в электронной форме о владельце сертификата. В ИПК он поступает от доверенной третьей стороны, называемой Сертификационным органом (СО), и содержит цифровую подпись этого органа.

Маркер X.509

Маркер X.509 — это сертификат X.509 с учетными данными конечного пользователя, которые могут быть переданы в сообщении SOAP. Маркер X.509 можно использовать для аутентификации пользователя на основе доверительных отношений (ИПК). Маркер X.509 также используется для подписания, шифрования и расшифровки сообщений SOAP.

XML

XML означает «eXtensible Markup Language» (расширяемый язык разметки) — язык, определяющий набор правил для кодирования документов в формате, который одновременно является человеко- и машиночитаемым. Он используется SOAP для кодирования сообщений, отправляемых веб-службами.

XML-подпись

Спецификация подписи XML является совместным проектом W3C и IETF. Подписи XML обеспечивают целостность, подтверждение подлинности сообщений и/или проверку подлинности подписи для данных любого типа независимо от того, выполнены ли они в формате XML, который включает подпись, или в другом формате.

XSD

XSD (определение схемы XML) — это рекомендация Консорциума мировой сети (W3C), которая определяет, как формально описать элементы в документе на расширяемом языке разметки (XML).

Техническая сводка

Для удобства пользования в настоящем приложении вся техническая информация представлена на одной странице.

Условное обозначение:	*Значение*	*Комментарии*
Спецификации eTIR	4.3	
URL-адрес среды UAT № 1	https://etir-uat-01.unece.org/	
Конечная точка SOAP для внутренних сообщений	{environment URL}/etir/v4.3/customs	
Конечная точка SOAP для внешних сообщений	{environment URL}/etir/v4.3/guaranteeChain	
Версия протокола SOAP	v1.2	
Протокол, используемый для HTTPS	TLS v1.2 и v1.3	Рекомендуется убедиться в том, что ваша информационная система может использовать TLS v1.3, так как версия TLS v1.2, скорее всего, будет выведена из эксплуатации по соображениям безопасности в 2021 году.

Условное обозначение:	*Значение*	*Комментарии*
Версия сертификата X.509	3	Это — сертификаты X.509, используемые в качестве электронных подписей для подписания сообщений среди различных систем eTIR.
Размер ключа для генерации сертификата X.509	2048 битов	
Алгоритм подписи для генерации сертификата X.509	SHA-256 на основе RSA	
Используемая версия UUID	4	Версии UUID используются в полях **Идентификатор сообщения** и **Контрольное функциональное обозначение**

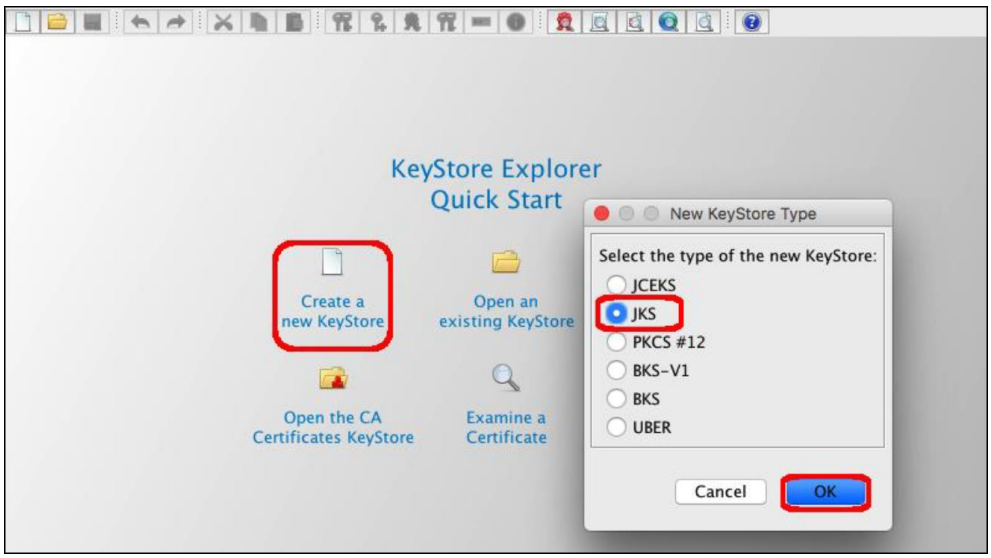
V. «KeyStore»: пошаговая генерация спаренных ключей

A. Использование приложения «KeyStore Explorer»

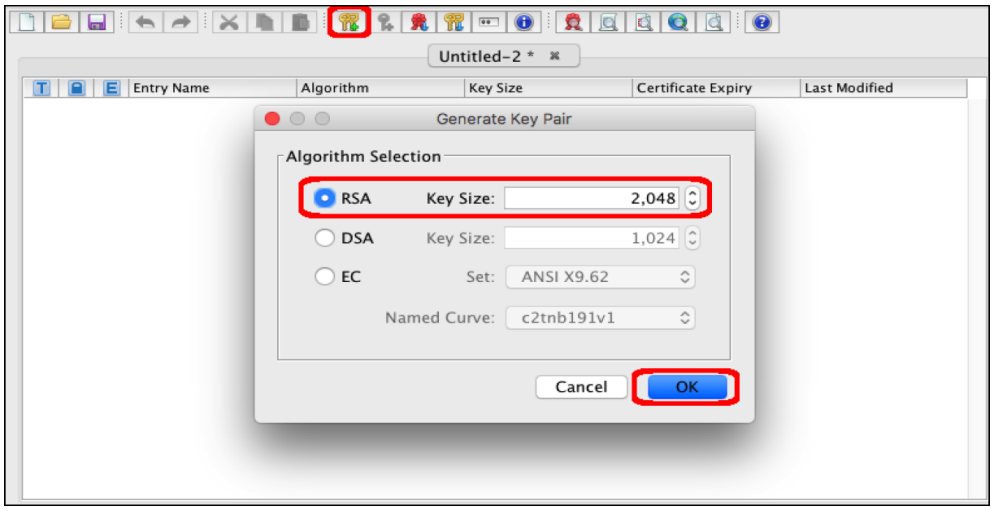
Данная процедура описывает работу с хранилищем ключей с помощью открытого инструмента ГПИ «KeyStore Explorer». Последняя версия «KeyStore Explorer» доступна для скачивания на URL: http://www.keystore-explorer.org/.

Генерация спаренных ключей

1. Открыть «KeyStore Explorer», нажать кнопку «Create a new KeyStore» и выбрать в качестве типа позицию «JKS».

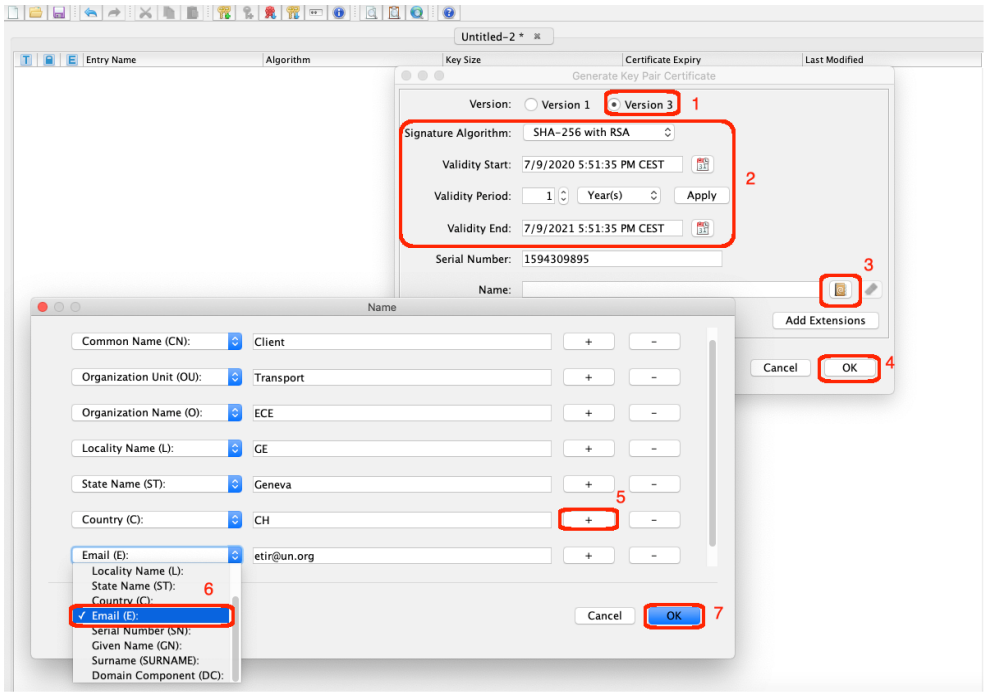


2. Сгенерировать спаренные ключи: выбрать RSA в качестве алгоритма с размером ключа 2048.

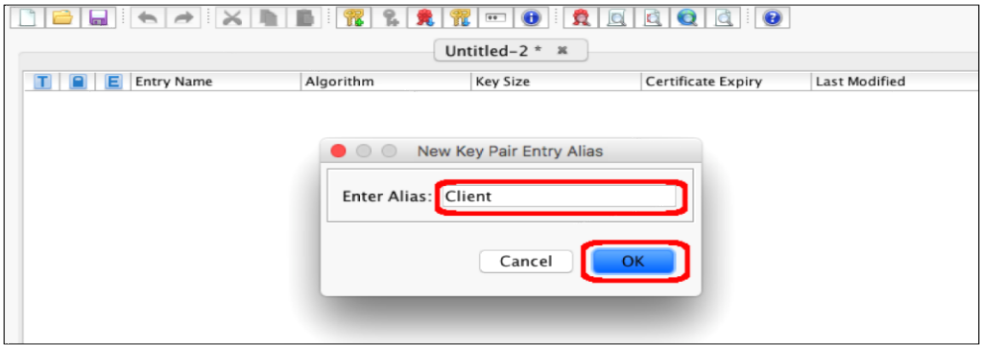


3. Сгенерировать сертификат спаренных ключей: выбрать «SHA-256 with RSA» (версия 3) в качестве алгоритма с желаемым сроком действия и выполнить шаги от единицы до четырех.

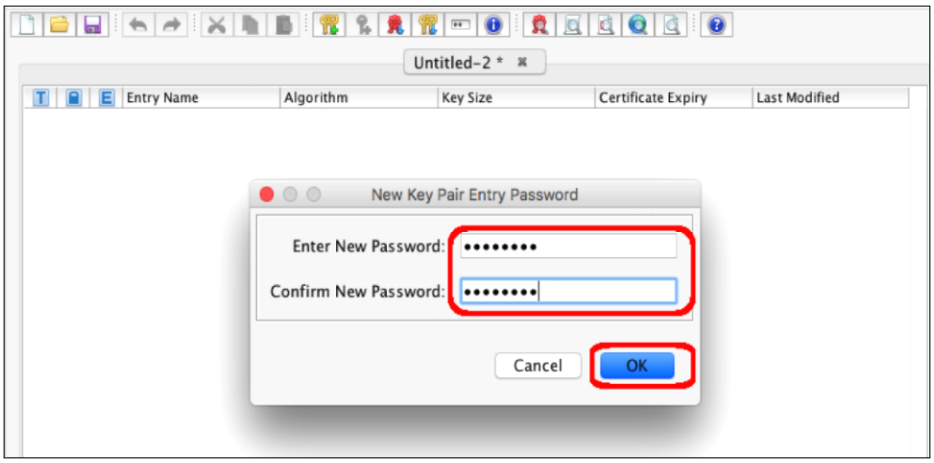
Убедиться в том, что вы вводите фактическую информацию, относящуюся к вашему таможенному отделению, а не выборочные значения, показанные на скриншоте ниже.



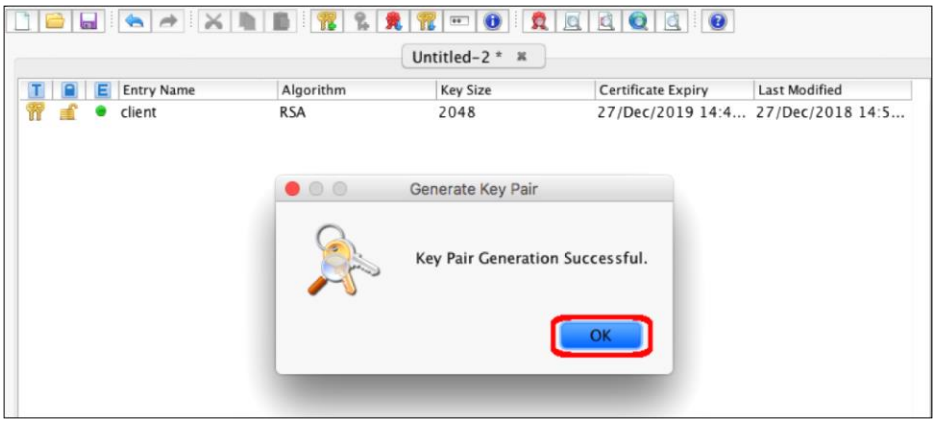
4. Появляется всплывающее окно с предложением определить псевдоним для спаренных ключей.



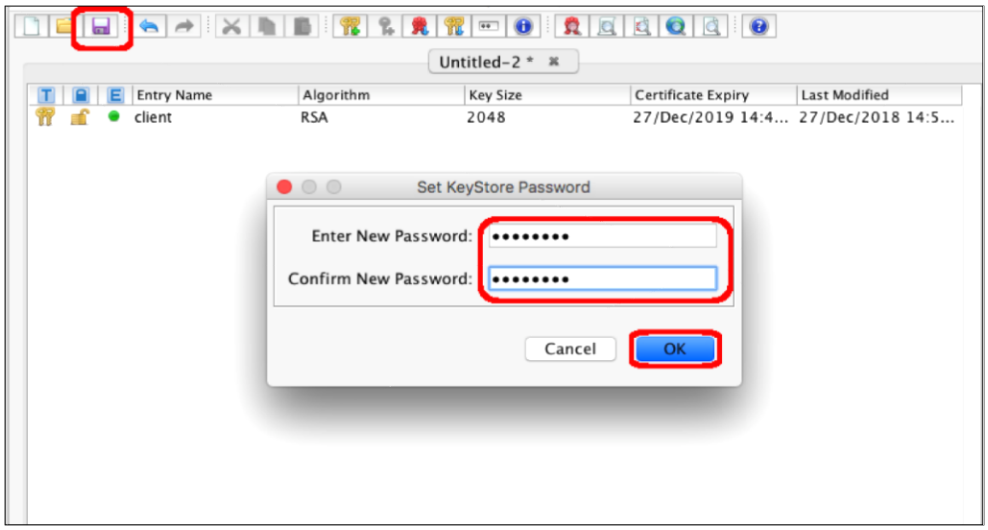
5. Появляется еще одно всплывающее окно с предложением определить пароль для спаренных ключей.



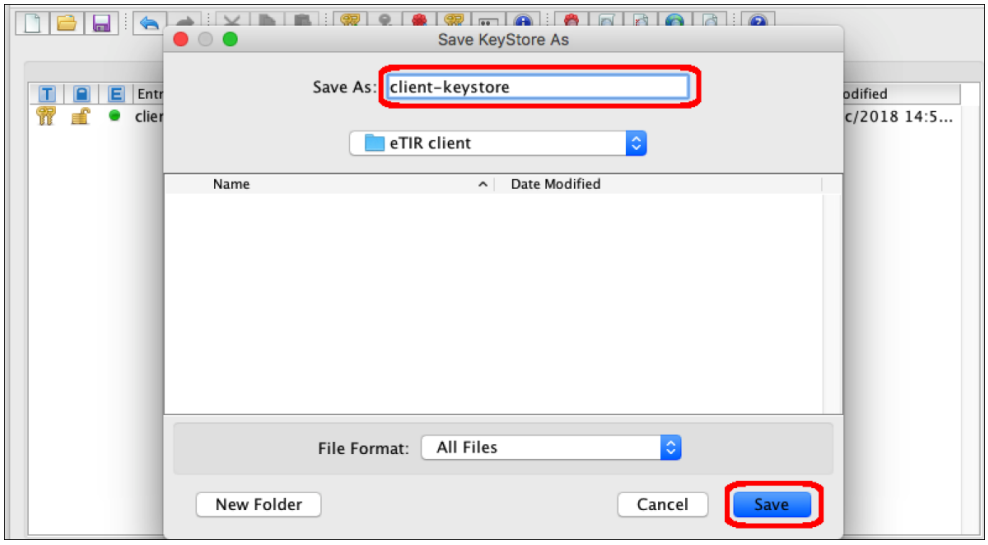
6. Генерация спаренных ключей подтверждена.



7. После генерации спаренных ключей нажать на кнопку «Save», после чего появится всплывающее окно для выбора пароля хранилища ключей (для спаренных ключей рекомендуется использовать иной пароль).

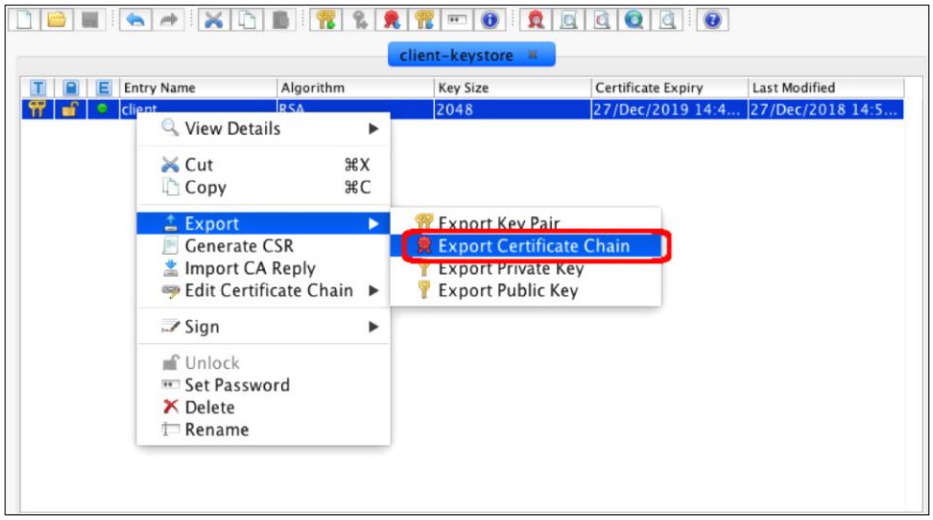


8. Указать имя и путь доступа к хранилищу ключей (расширение файла должно быть «.jks») и затем нажать кнопку «Save».

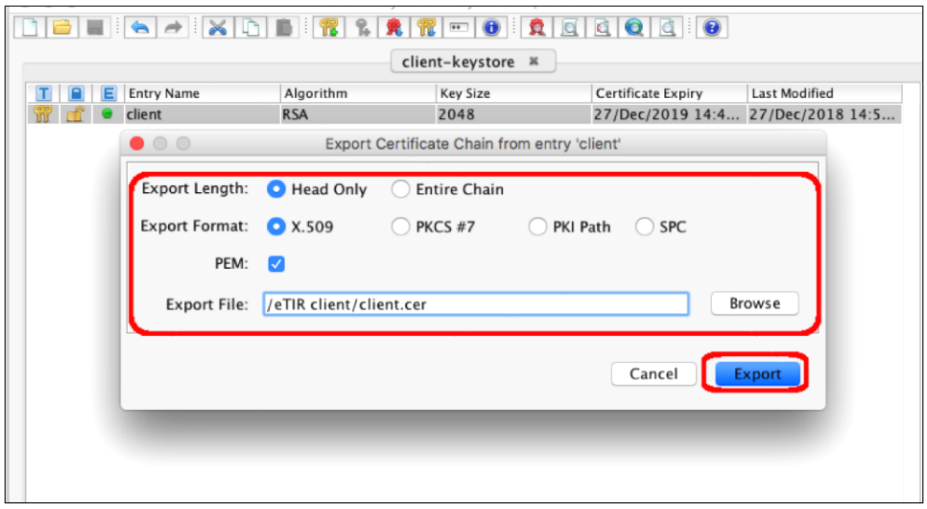


Сертификат экспортирующего клиента

1. Открыть сгенерированное хранилище ключей из проводника «KeyStore Explorer», щелкнуть правой кнопкой мышки на сгенерированную пару ключей, выбрать «Export», а затем «Export Certificate Chain».

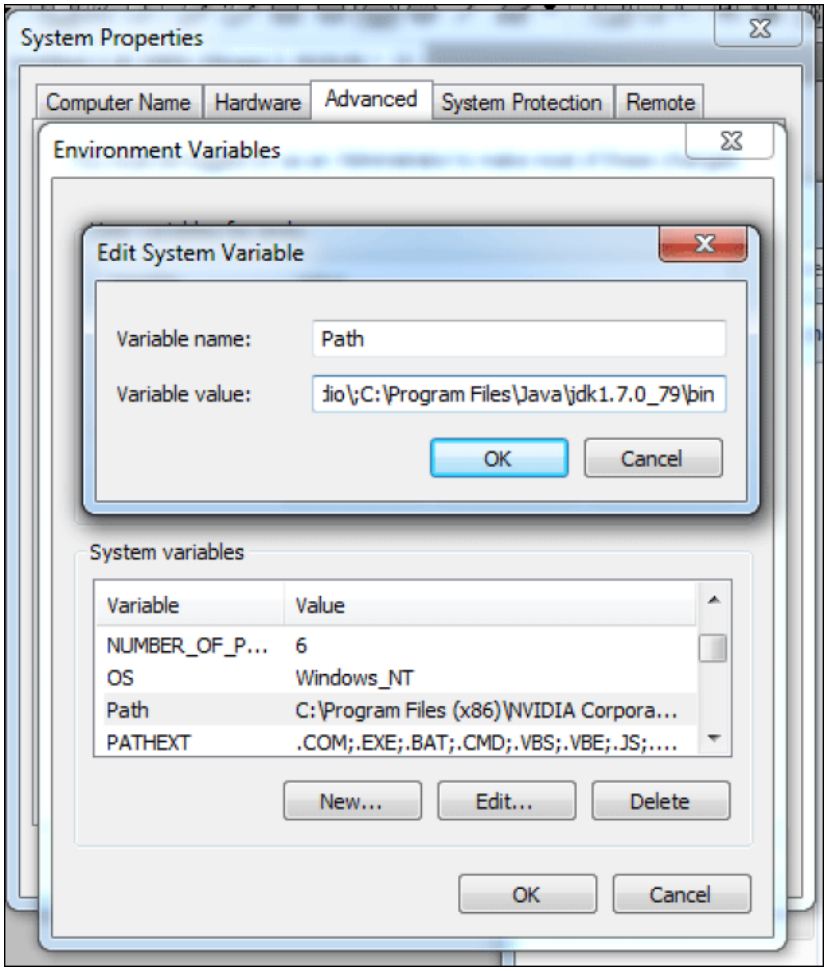


2. Сохранить значения по умолчанию, выбрать путь доступа экспортируемого сертификата и нажать на кнопку «Export».



B. Использование интерфейса командной строки «Keytool»

«Java Keytool» — это инструмент командной строки для генерации спаренных открытых/закрытых ключей и хранения их в хранилище «Java KeyStore». Перед тем как использовать командную строку «Keytool», к пути доступа к переменной среды необходимо добавить каталог «Java bin», как показано ниже:



Исполняемый модуль «Keytool» называют «keytool». Для его исполнения необходимо открыть интерфейс командной строки (cmd, console, shell и т. д.) и заменить каталог на каталог «bin» своей установки «Java SDK». Ввести ключевое инструментальное средство, после чего должно появиться приблизительно такой текст:

C:\Program Files\Java\jdk1.8.0_111\bin>keytool
Key and Certificate Management Tool

Команды:

- certreq генерирует запрос на сертификат;
- changealias изменяет псевдоним записи;
- delete удаляет запись;
- exportcert экспортирует сертификат;
- genkeypair генерирует спаренные ключи;
- genseckey генерирует секретный ключ;
- gencert генерирует сертификат из запроса на сертификат;
- importcert импортирует сертификат или цепочку сертификатов;
- importpass импортирует пароль;
- importkeystore импортирует одну или все записи из другого хранилища;
- keypasswd изменяет пароль ключа на входе;
- list выводит на экран список позиций в хранилище ключей;
- printcert печатает содержимое сертификата;
- printcertreq печатает содержимое запроса на сертификат;
- printcrl печатает содержимое файла CRL;
- storepasswd изменяет пароль хранилища ключей.

Для использования «keytool-command name-help» выбрать позицию «command_name».

Команда «Keytool» может принимать множество аргументов, поэтому запомнить, как их правильно задавать, может оказаться затруднительным. В этой связи было бы хорошо создать несколько скриптов, которые позволили бы впоследствии упростить выполнение команд «Keytool», что также позволит упростить соответствующие аспекты обслуживания.

Генерация спаренных ключей

Генерация спаренных открытых/закрытых ключей является одной из наиболее распространенных задач при использовании программного средства «Keytool». Сгенерированная пара ключей записывается в файл «Java KeyStore» в качестве самоподписанных спаренных ключей. Ниже показан общий формат командной строки для генерации спаренных ключей с помощью программного средства «Keytool»:

```
-genkeypair {-alias alias} {-keyalg keyalg} {-keysize keysize} {-sigalg sigalg} [-dname dname] [-keypass keypass] {-
validity valDays} {-storetype storetype} {-keystore keystore} [-storepass storepass] {-providerClass
provider_class_name {-providerArg provider_arg}} {-v} {-protected} {-Jjavaoption}
```

Хранилище ключей можно сгенерировать с помощью следующего набора инструкций:

```
keytool -genkeypair -alias client -keystore ClientKeyStore.jks -keyalg RSA \
-dname "CN=Client, OU= Transport, O=ECE, L=GE, ST=GENEVA, C=CH, EMAILADDRESS=etir@un.org" \
-sigalg SHA256withRSA -validity 1000

Enter keystore password: keyStorePassword
Re-enter new password: keyStorePassword

Enter key password for <client>
       (RETURN if same as keystore password): certificatePassword
Re-enter new password: certificatePassword
```

Сертификат экспортирующего клиента

Командная строка «Keytool» также позволяет экспортировать сертификаты, содержащиеся в соответствующем хранилище ключей. Ниже приведены шаблоны команд «Keytool» для экспорта сертификатов:

```
-exportcert {-alias alias} {-file cert_file} {-storetype storetype} {-keystore keystore} [-storepass storepass] {-
providerName provider_name} {-providerClass provider_class_name {-providerArg provider_arg}} {-rfc} {-v} {-
protected} {-Jjavaoption}
```

В той же папке, что и в сгенерированном хранилище ключей, извлечь сертификат открытого ключа, используя нижеследующую командную строку, и отправить его по электронным адресам ЕЭК ООН (для хранения в хранилище доверенных сертификатов сервера приложений):

```
keytool -exportcert -keystore ClientKeyStore.jks -alias client -file client.cer -storepass certificatePassword
```

Щелкнуть мышкой на «Enter» с целью сохранить сертификат в той же папке, что и хранилище ключей:

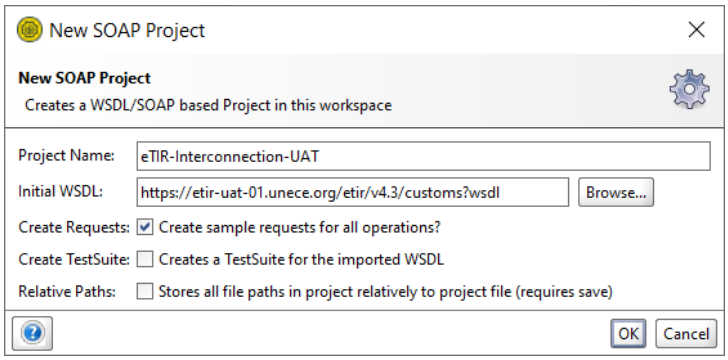
```
Certificate stored in file <client.cer>
```

VI. Краткое руководство по использованию программного средства SOAP UI

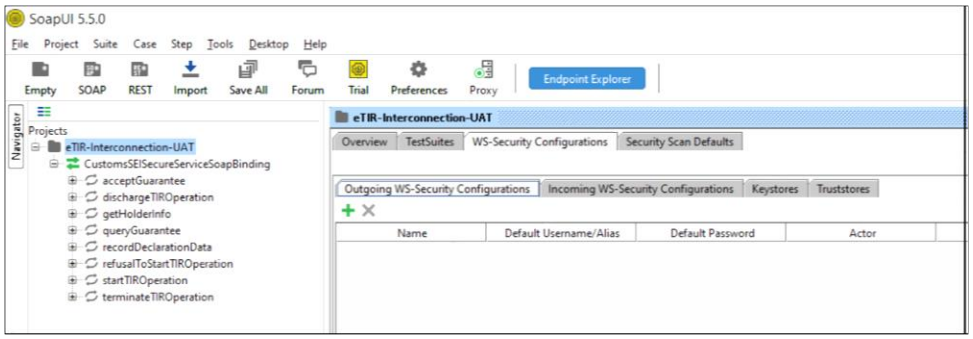
Ниже приведены основные шаги, необходимые для установки и конфигурации инструмента тестирования пользовательского интерфейса SoapUI, а также для отправки запроса SOAP на шифрование с использованием SoapUI.

1. Скачать и установить приложение «SoapUI Open Source edition» с веб-страницы загрузки с сайта SOAP UI.

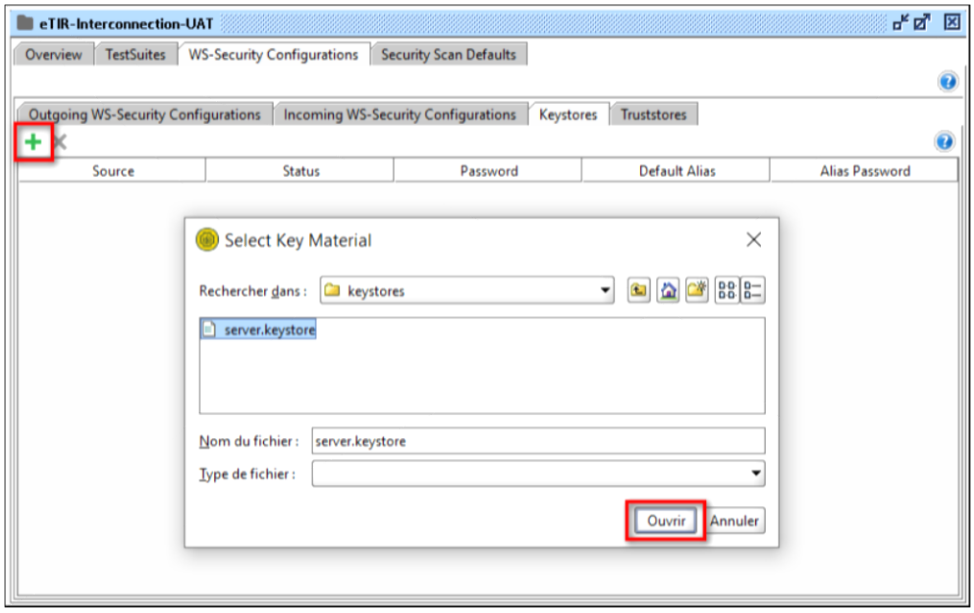
2. Запустить приложение SoapUI и создать новый проект SOAP, нажав на кнопку «File» и «New SOAP Project». Затем ввести имя и установить тестовый URL для WSDL: https://etir-uat-01.unece.org/etir/v4.3?wsdl.



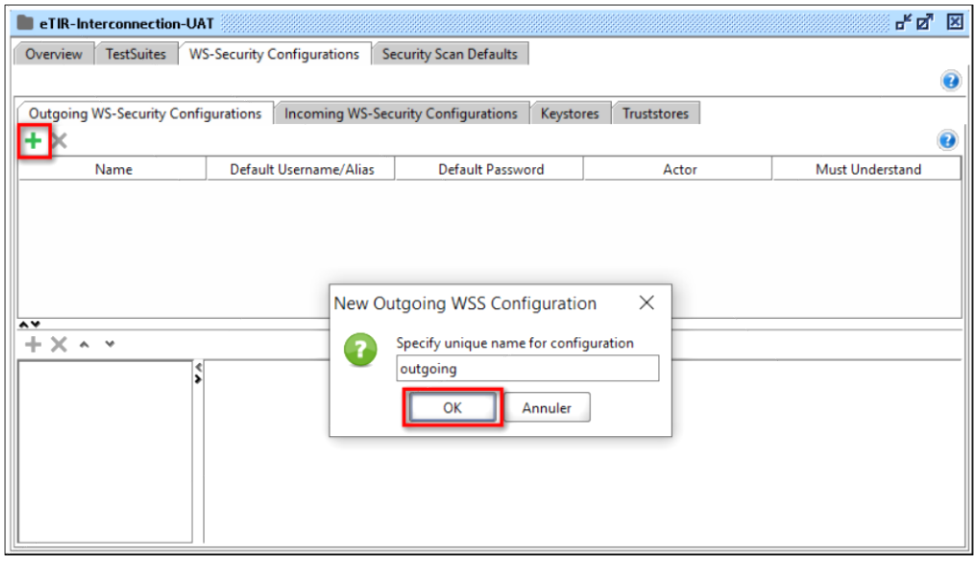
3. После создания тестового проекта перейти на вкладку «WS-Security Configurations» в меню «Show Project View», щелкнув правой кнопкой мыши на папку проекта.



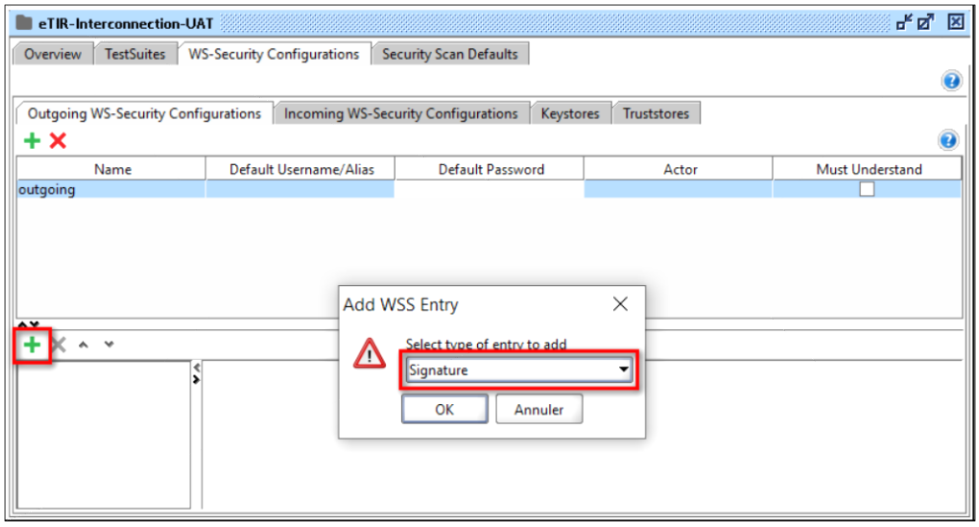
4. Добавить «KeyStore», нажав на кнопку «Add» и просмотрев свой файл «KeyStore», ввести пароль и нажать на «OK».



5. Затем перейти на вкладку «Outgoing WS-Security Configurations» с целью отобразить конфигурации, которые должны применяться к исходящим сообщениям, включая запросы. Добавить новую исходящую конфигурацию.

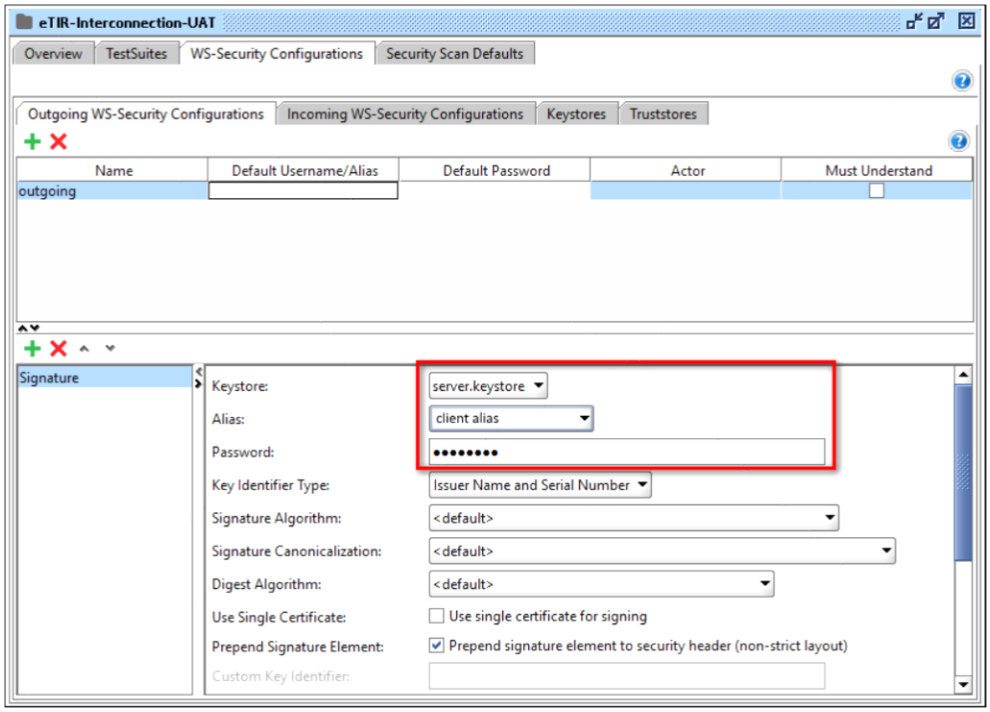


6. Этот тип конфигурации используется для шифрования, подписания и добавления заголовков SAML, временных меток и имен пользователей. В этой ситуации используется только часть, которая подписывается. Добавить позицию для подписи.

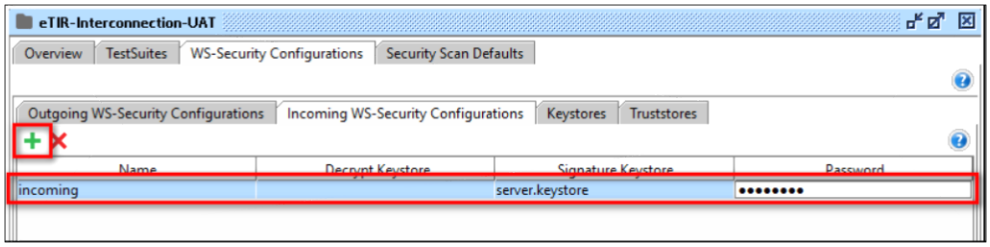


7. Выбрать хранилище ключей и псевдоним ключа (спаренных ключей), который будет использоваться вместе с паролем для этого псевдонима. Это — поля с настраиваемой конфигурацией:

- **Хранилище ключей**: хранилище ключей используется при подписании сообщения.
- **Псевдоним**: псевдоним (спаренные ключи), используемый при подписании сообщения.
- **Пароль**: пароль псевдонима.

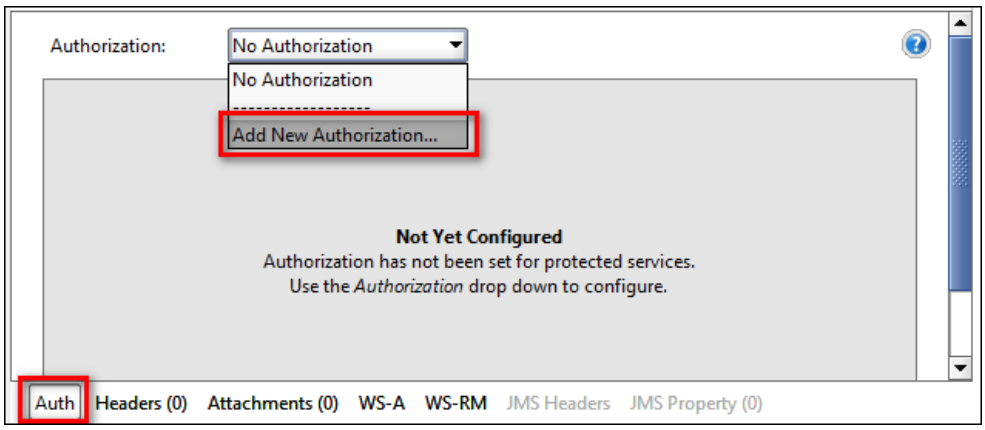


8. Затем перейти на вкладку «**Incoming WS-Security Configurations**» с целью отобразить конфигурации, которые должны применяться к входящим сообщениям, включая ответы. Указать название хранилища ключей и пароль. Такая конфигурация обеспечит валидацию подписи входящих ответов с сервера eTIR.

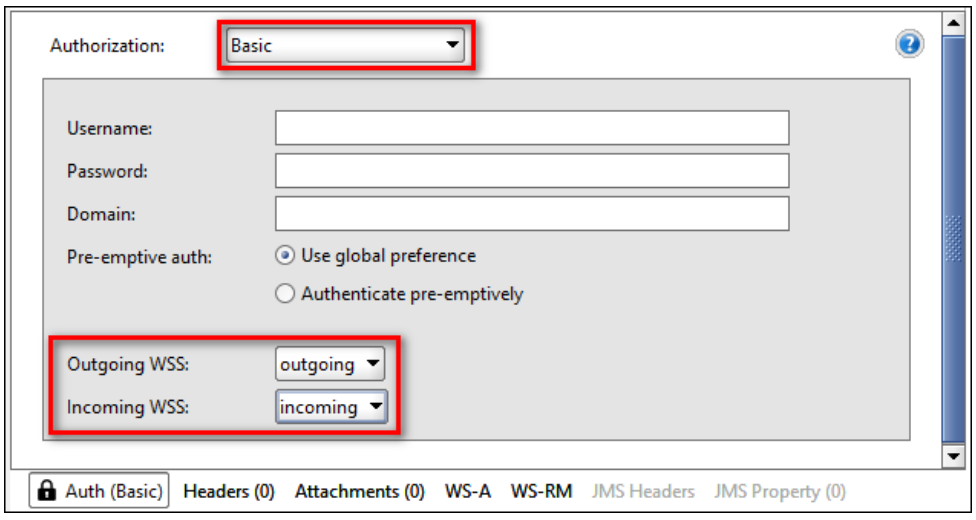


9. Открыть SOAP-запрос, щелкнув правой кнопкой мыши на «**queryGuarantee**» на древовидное меню с левой стороны приложения, а затем выбрать «**New Request**» и дать ему соответствующее имя.

10. Перейти на вкладку «Auth» в левом нижнем углу и в списке выпадающего меню «Authorization» выбрать «**Add New Authorization**»... и затем выбрать «**Basic**» для базового разрешения.



11. В появившемся в новом окне конфигурации выбрать вкладку с предварительно настроенным исходящим и входящим WSS.

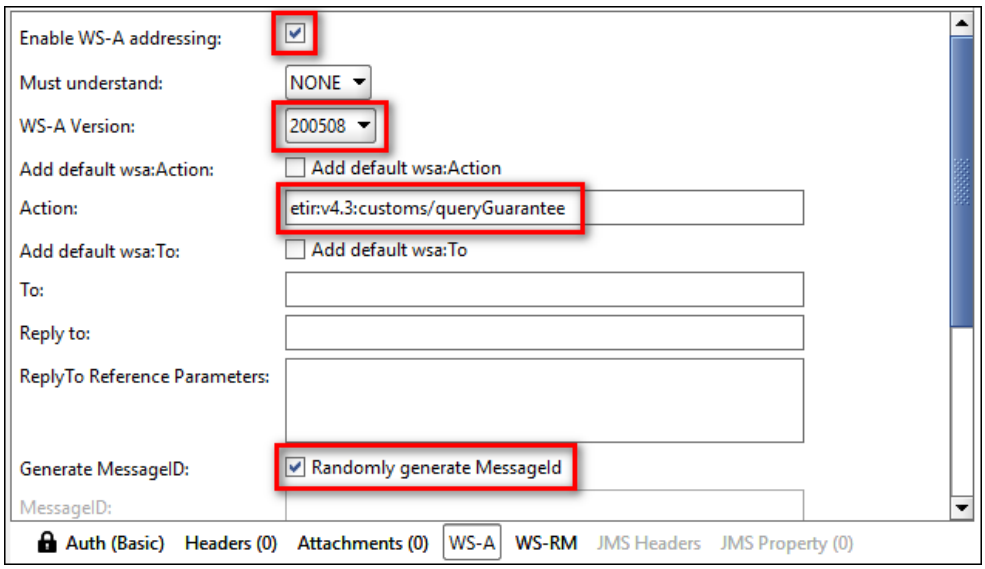


12. Перейти на вкладку WS-A. Убедиться в том, что конфигурация следующих настроек выполнена правильно:

a) Активировать адресацию WS-A

b) Активировать позицию «Randomly generated MessageId»;

c) Убедиться в том, что поле действия правильное и выглядит следующим образом: https://etir-uat-01.unece.org/etir/v4.3?wsdl.



13. Еще раз активировать «SoapUI» применительно к веб-службе, передавая на сервер сообщения SOAP. В этой связи можно воспользоваться нижеприведенным примером:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:etir="etir:v4.3:customs" xmlns:etir1=
"etir:I5:v4.3">
   <soap:Header/>
   <soap:Body>
      <etir:queryGuarantee>
         <etir1:InterGov>
            <etir1:FunctionCode>9</etir1:FunctionCode>
            <etir1:ID>e393d591-963a-4cd1-9d4a-08467afcf524</etir1:ID>
            <etir1:TypeCode>I5</etir1:TypeCode>
            <etir1:ReplyTypeCode>00</etir1:ReplyTypeCode>
            <etir1:ObligationGuarantee>
               <etir1:ReferenceID>XC95000003</etir1:ReferenceID>
            </etir1:ObligationGuarantee>
         </etir1:InterGov>
      </etir:queryGuarantee>
   </soap:Body>
</soap:Envelope>
```

VII. Пример SOAP-запроса с использованием WSS4J

В данном примере описывается, как использовать стандарт «Apache Web Services Security» для «Java WSS4J» в целях создания SOAP-клиента. Проект WSS4J предусматривает разработку базовых стандартов безопасности на языке программирования «Java» для веб-служб, т. е. спецификаций безопасности веб-служб ОРССИ (WS-Security), разрабатываемых Техническим комитетом по безопасности веб-служб ОРССИ. WSS4J ориентирован на введение в практику профиля маркера X.509. В данном примере подпись полученных ответов из международной системы eTIR может быть проверена и подтверждена методом «перехвата». Эта часть поясняется в следующем разделе.

Сначала нужно создать свой сертификат. В этих целях можно выполнить две процедуры, детально изложенные в разделе «KeyStore»: пошаговая генерация спаренных ключей с помощью приложения «KeyStore Explorer» или интерфейса командной строки «Keytool».

Создать файл криптографических свойств

Проект WSS4J нуждается в некоторых криптографических свойствах. Эти свойства сгруппированы в трех разделах, как показано ниже:

```
#Crypto properties
#General properties:
#WSS4J specific provider used to create Crypto instances. Defaults to
"org.apache.ws.security.components.crypto.Merlin".
#org.apache.ws.security.crypto.provider=
#Keystore properties:
#The location of the keystore
org.apache.ws.security.crypto.merlin.keystore.file=keystoreFile
#The password used to load the keystore. Default value is "security".
org.apache.ws.security.crypto.merlin.keystore.password=password
#Type of keystore. Defaults to: java.security.KeyStore.getDefaultType()), normally it is JKS
#org.apache.ws.security.crypto.merlin.keystore.type=JKS
#The default keystore alias to use, if none is specified.
org.apache.ws.security.crypto.merlin.keystore.alias=aliasName
```

Пароль блока обработки обратных вызовов

Пароль хранилища ключей указывается в файле криптографических свойств, как показано в предыдущем разделе. Однако пароль закрытого ключа пока не предоставлен. В файле криптографических свойств есть одно свойство, которое позволяет уточнить этот момент, а именно:

```
#The default password used to load the private key. Normally it is provided through a password-callback class
#org.apache.ws.security.crypto.merlin.keystore.private.password=
```

Один из видов надежной практики в этом деле состоит в том, чтобы предоставлять его через блок обработки пароля обратных вызовов, что обеспечит более высокий уровень безопасности. Блок обработки пароля обратных вызовов может извлечь пароль из базы данных, протокола LDAP или из более защищенных мест. В данном примере мы просто отображаем пароль закрытого ключа из класса блока обработки пароля обратного вызова. Ниже приведена базовая реализация блока обработки пароля обратного вызова:

```
import java.io.IOException;
import javax.security.auth.callback.Callback;
import javax.security.auth.callback.CallbackHandler;
import javax.security.auth.callback.UnsupportedCallbackException;
import org.apache.ws.security.WSPasswordCallback;

public class ServerPasswordCallback implements CallbackHandler {

  public void handle(Callback[] callbacks) throws IOException, UnsupportedCallbackException {

     for (int i = 0; i < callbacks.length; i++) {
      WSPasswordCallback pc = (WSPasswordCallback) callbacks[i];
      pc.setPassword("key-pass");
    }
  }
}
```

«Перехватчики WSS4J»

Последний шаг — запись входящих и исходящих «перехватчиков WSS4J» со стороны клиента. Для запроса SOAP необходимы два перехватчика. Компоненты перехватчика настраивают остальные параметры безопасности, такие как: наличие в сообщении SOAP временно́й метки, подпись, какие части должны быть подписаны,

какой используется алгоритм, какие используются типы двоичного маркера (BinarySecurityToken) и ссылки, какой будет указатель класса блока обработки пароля обратного вызова и т. д. Все эти параметры настроены в соответствующей карте как пары «ключ/значение». Вся эта карта отображена в виде тегов «WSHandler» на странице конфигурации WSS4J. Просьба обратить внимание на то, что многие ключи/настройки используются по умолчанию.

Входящие и исходящие «перехватчики WSS4J» — конфигурация «XML-Spring»

```
<beans xmlns="http://www.springframework.org/schema/beans"
 xmlns:jaxws="http://cxf.apache.org/jaxws" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="
 http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
 http://cxf.apache.org/jaxws http://cxf.apache.org/schemas/jaxws.xsd">

 <bean id="logInBound" class="org.apache.cxf.interceptor.LoggingInInterceptor" />
 <bean id="logOutBound" class="org.apache.cxf.interceptor.LoggingOutInterceptor" />
 <jaxws:client id="customsClient" serviceClass="org.unece.etir.ws.cusc.CustomsSEISecure"
 address="https://ece-dev-etir.unece.org/etir/services/CustomsToETIR-1">
 <jaxws:inInterceptors>
 <ref bean="logInBound" />
 <ref bean="inbound-security" />
 </jaxws:inInterceptors>
 <jaxws:outInterceptors>
 <ref bean="logOutBound" />
 <ref bean="outbound-security" />
 </jaxws:outInterceptors>
 </jaxws:client>

 <!-- WSS4JOutInterceptor for signing outbound SOAP messages -->
    <bean class="org.apache.cxf.ws.security.wss4j.WSS4JOutInterceptor" id="outbound-security">
        <constructor-arg>
            <map>
                <entry key="action" value="Signature"/>
                <entry key="user" value="client"/>
                <entry key="signaturePropFile" value="client-crypto.properties"/>
                <entry key="passwordCallbackClass" value="client.ClientPasswordCallback"/>
                <entry key="signatureParts"
                value="{Element}{http://schemas.xmlsoap.org/soap/envelope/}Body"/>
            </map>
        </constructor-arg>
    </bean>

    <!--  WSS4JInInterceptor for validating the signature of inbound SOAP messages -->
    <bean class="org.apache.cxf.ws.security.wss4j.WSS4JInInterceptor" id="inbound-security">
        <constructor-arg>
            <map>
                <entry key="action" value="Signature"/>
                <entry key="signaturePropFile" value="client-crypto.properties"/>
                <entry key="passwordCallbackClass" value="client.ClientPasswordCallback"/>
            </map>
        </constructor-arg>
    </bean>

</beans>
```

Входящие и исходящие «перехватчики WSS4J» — конфигурация Java

«WSS4JOutInterceptor» подписывает исходящие SOAP-сообщения:

```
Map<String, Object> propsOut = new HashMap<String, Object>();
propsOut.put(WSHandlerConstants.ACTION, WSHandlerConstants.SIGNATURE);
propsOut.put(WSHandlerConstants.SIGNATURE_PARTS, "{Element}{http://www.w3.org/2003/05/soap-envelope}Body;");
propsOut.put(WSHandlerConstants.PW_CALLBACK_CLASS, PasswordCallback.class.getName());
propsOut.put(WSHandlerConstants.USER, "client");
propsOut.put(WSHandlerConstants.SIG_PROP_FILE, "META-INF/client-security.properties");
```

«WSS4JInInterceptor» проверяет подписи входящих SOAP-сообщений:

```
Map<String, Object> propsIn = new HashMap<String, Object>();
propsIn.put(WSHandlerConstants.ACTION, WSHandlerConstants.SIGNATURE);
propsIn.put(WSHandlerConstants.PW_CALLBACK_CLASS, PasswordCallback.class.getName());
propsIn.put(WSHandlerConstants.SIG_PROP_FILE, "META-INF/client-security.properties");

WSS4JOutInterceptor wssOut = new WSS4JOutInterceptor(propsOut);
WSS4JInInterceptor wssIn = new WSS4JInInterceptor(propsIn);
Client client = ClientProxy.getClient(port);
Endpoint endpoint = client.getEndpoint();
endpoint.getOutInterceptors().add(wssOut);
endpoint.getInInterceptors().add(wssIn);
endpoint.getInInterceptors().add(new LoggingInInterceptor());
endpoint.getOutInterceptors().add(new LoggingOutInterceptor());
```