# Прописывание документа на клиенте.

Подпись осуществляется с помощью Java плагина и криптопровайдера установленного на клиентской машине. Плагин должен содержать реализации работы с различными криптопровайдерами, а конкретная реализация выбирается с помощью настроек. Реализации работы с криптопровайдерами могут взаимодействовать с JCP, SCP, аппаратными ключами, но иметь один интерфейс. Плагин взаимодействует с платформой с помощью web сервисов. На рисунке 1 отображена схема взаимодействия компонент подсистемы ЭП.



Алгоритм работы модуля:

* На странице, содержащей функциональность ЭП размещается апплет. На странице апплет должен быть в единственном экземпляре.
* При своей инициализации апплет обращается по web-service API к платформе за инициализационными данными. В этих данных присутствует вся информация, необходимая для выбора криптопровайдера, указания корректных криптоалгоритмов, указания типа ЭП итд.
* При нажатие кнопки «Подписать», java-script вызывает функцию апплета sign и передает в эту функцию строковой идентификатор подписываемого объекта.
* Апплет обращается к web сервису за подписываемым контентом, передав туда идентификатор.
* Web сервис исходя из типа запрашиваемого объекта и конфигурации модуля формирует контент и отправляет его апплету.
* Апплет выполняет ЭП контента и передает полученную подпись и сертификат обратно на сервер.
* Сервер сохраняет ЭП и сертификат которым был подписан контент в таблице ДО signature и certificate. Сертификат ищется по серийному номеру, и если не находится то создается новый.
* Апплет возвращает структуру с информацией о подписи. В структуре содержится информация о успешности выполнения ЭП в виде boolean флага и информацию о ошибках в виде строки в случае если флаг успешности операции = false.

# Крипто модуль имеет свою xml конфигурацию, которая хранится на сервере и обрабатывается сервисом конфигурации. Конфигурация содержит глобальные настройки, так и настройки для конкретных типов. Глобальные настройки содержат информацию о применяемом крипто провайдере и соответственно имя имплементации класса, которая будет инициализироваться в апплете, информацию о криптоалгоритмах, типе ЭП и иную информацию, необходимую для работы апплета. В глобальных настройках может быть раздел, который конфигурирует конкретного криптопровайдера и имеющим свободную структуру, то есть в xsd элемент any.

Глобальные настройки криптопровайдера могут быть только одни в конфигурации и это контролируется валидатором.

Так же в конфигурации настраивается способ получения контента по идентификатору. Подписываемое вложение может быть как файл документа так и XML с атрибутами. Алгоритм получения подписываемого контента задается в конфигурации. Формат xml конфигурации приведен ниже.

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<configuration xmlns="https://cm5.intertrust.ru/config">

<crypto-settings name=”crypto”>

<crypto-global>

<applet-impl class-name=”ru.intertrust.cm5.core.crypto.applett.CryptoProImpl”/>

<signature-settings>

<здесь может быть любая структура xml ее состав зависит от имплементации крипто провайдера>

</signature-settings>

</crypto-global>

<sign-datas type=”indoc”>

<sign-attachment type=”indoc\_attachment”/>

<sign-card>

<attribute name=”subject” doel=”subject”/>

<back-link name=”resolution” doel=”resolution^document”>

<attribute name=”executor” doel=” executor”/>

<back-link name=”execution\_report” doel=”report^resolution”>

<attribute name=”report-text” doel=” body”/>

</back-link>

</back-link>

</sign-card>

</sign-datas>

</crypto-settings>

</configuration>

Настройка получения подписываемых контентов по идентификатору доменного объекта производится тегом sign-datas, в атрибуте type задается имя типа, для которого описывается алгоритм. При этом учитывается наследование, то есть если явно для наследника не задан алгоритм то берется алгоритм родителя. Контенты бывают двух видов. Первый вид это вложение, задается тегом sign-attachment, у которого указан тип вложения. В случае множества вложений подписываются все по отдельности. Второй вид – атрибутивный состав карточки, для этого доменный объект преобразуется в xml. Алгоритм преобразования задается в теге sign-card. Внутри тега могут быть теги attribute и back-link. Тег attribute указывает на необходимость создать у корневого элемента атрибут с именем name и значением вычисленным с помощью doel выражения doel. Тег back-link указывает что надо создать новый элемент внутри корневого с именем name и заполнить его по правилам, которые указаны в xml. Уровней вложенности может быть бесконечно.

Для хранения ЭП и возможности произвести проверку ЭП создаются ряд ДО для хранения данных.

ДО хранения ЭП для объектов - signature

|  |  |  |
| --- | --- | --- |
| имя | тип | описание |
| data\_id | ID (type =\*) | Ссылка на любой доменный объект. |
| signature | text | Электронная подпись в формате base64 |
| certificate\_id | ID (type= certificate) | Ссылка на сертификат, которым произведена ЭП |

ДО хранения сертификатов - certificate

|  |  |  |
| --- | --- | --- |
| имя | тип | описание |
| serial\_number | String (уникальный ключ) | Серийный номер сертификата. Используется для поиска уже сохраненных сертификатов. Берется из атрибута сертификата |
| certificate | text | Сертификат в формате base64 |
| signature | text | Электронная подпись |
| person\_id | ID (type= person) | Ссылка на персону |

# Проверка ЭП на сервере.

Необходимо реализовать сервис проверки ЭП на сервере. Сервис должен быть выполнен в виде EJB со следующим интерфейсом:

/\*\*

\* Сервис для выполнения криптографических операций на платформе CM5.

\* **@author** larin

\*

\*/

**public** **interface** CryptoTool{

/\*\*

\* Проверка усовершенствованной ЭП включеннной в документ

\* **@param** inputStream поток содержащий документ со встроенной электронной подписью

\* **@return**

\*/

VerifyResult verify(InputStream document);

/\*\*

\* Проверка Усовершенствованной ЭП не включенной в документ

\* **@param** document поток содержащий документ

\* **@param** signature электронная подпись

\* **@return**

\*/

VerifyResult verify(InputStream document, **byte**[] signature);

/\*\*

\* Проверка стандартной ЭП

\* **@param** document поток содержащий документ

\* **@param** signature электронная подпись

\* **@param** signerSertificate сертификат подписавшено документ в формате DER

\* **@return**

\*/

VerifyResult verify(InputStream document, **byte**[] signature, **byte**[] signerSertificate);

/\*\*

\* Проверка стандартной ЭП с помощью сертификата, указанного в доменном объекте персоны

\* **@param** document поток содержащий документ

\* **@param** signature электронная подпись

\* **@param** personId идентификатор персоны которой был подписан документ

\* **@return**

\*/

VerifyResult verify(InputStream document, **byte**[] signature, Id personId);

/\*\*

\* Проверка стандартной ЭП с помощью сертификата, указанного в доменном объекте персоны

\* @param documentId идентификатор документа

\* @return

\*/

VerifyResult verify(Id documrntId);

}

В качестве возвращаемого значения методов проверки ЭП используется структура:

public class VerifyResult implements Dto{

private static final long *serialVersionUID* = -7995499379143873587L;

private boolean valid;

private List<String> messages;

private List<String> warnings;

private List<String> errors;

private String signer;

public List<String> getMessages() {

return messages;

}

public void setMessages(List<String> messages) {

this.messages = messages;

}

public List<String> getWarnings() {

return warnings;

}

public void setWarnings(List<String> warnings) {

this.warnings = warnings;

}

public List<String> getErrors() {

return errors;

}

public void setErrors(List<String> errors) {

this.errors = errors;

}

public boolean isValid() {

return valid;

}

public void setValid(boolean valid) {

this.valid = valid;

}

public String getSigner() {

return signer;

}

public void setSigner(String signer) {

this.signer = signer;

}

}

EJB должен вызывать спринг бин с конкретной реализацией, использующей конкретный крипто провайдер. Интерфейс спринг бина аналогичен интерфейсу EJB. Для исключения дублирования интерфейсов необходимо создать 2 интерфейса наследников CryptoTool: CryptoService и CryptoBean. Уои должен имплементировать интерфейс CryptoService а спринг бин – CryptoBean. Имплементация спринг бина должна быть вынесена в отдельный мавен артифакт, например crypto-pro-crypto-service и опционально подключен в итоговую сборку платформы. beans.xml файл проекта имплементации должен объявлять спринг бин в профиле отличном от default и далее нужная имплементация должна подключатся с помощью выбора профиля spring в файле standalone.xml. Пример файла конфигурации спринг бина:

<beans profile=*"crypto-pro"*>

<bean id=*"cryptoService"* class=*"ru.intertrust.cm.core.dao.impl.CryptoProCryptoService"* />

</beans>

Подключение конкретной реализации в standalone.xml

<system-properties>

...

<property name="spring.profiles.active" value="default, *crypto-pro*"/>

</system-properties>

Структура VerifyResult содержит информацию о результате проверки. Поле valid указывает на корректность вычисления крипто функций. Поле signer содержит информацию о подписавшем. Структура так же может содержать дополнительную информацию по проверки ЭП разделенную на 3 категории. Информационные сообщения в поле messages, например использующиеся алгоритмы ЭП, шифрования или хеширования. Предупреждающая информация warnings, например отсутствие доверенного корневого сертификата в цепочке сертификатов подписавшего. Информация об ошибке errors, например ошибка при вызове криптографических функций.

Для реализации метода VerifyResult verify(InputStream document, **byte**[] signature, Id personId) необходимо изменить ДО персоны для возможности хранения нескольких сертификатов для одного пользователя. Для этого создается новый ДО person\_certificate со следующими полями:

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип | Примечания |
| person\_id | Ссылка на персону | Идентификатор персоны |
| certificate | Text | Сертификат в формате base64 |

При проверки ЭП методом VerifyResult verify(InputStream document, **byte**[] signature, Id personId) необходимо производить проверку со всеми сертификатами персоны до получения хотя бы одного валидного результата.

Для работы функции VerifyResult verify(Id documrntId) необходимо создать ДО содержащей ЭП доменных объектов со следующими полями.

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип | Примечания |
| document\_id | Ссылка на любой доменный объект | Идентификатор подписанного доменного объекта |
| certificate\_id | Ссылка на объект типа person\_certificate | Идентификатор сертификата, которым был подписан документ |
| algorithm | Строка | Имя алгоритма подписи. Алгоритм подписи устанавливается как конфигурация платформы и содержит описание полей до, которые участвуют в формирование строки, которая затем подписывается. Конфигурация привязывается к типу доменного объекта. Формат XML в данном документе не описывается и будет уточнен на этапе разработки функциональности по подписыванию документов с помощбю ЭП. |

Реализация метода VerifyResult verify(Id documrntId) надо производить после разработки функционала подписывания документов с помощью ЭП.