Вход через цифровые сертификаты (КЭП)

Оглавление

Назначение
  # Описание работы
Сценарии
  # Аутентификация по сертификату
  # Регистрация новой учетной записи по сертификату
  # Добавление сертификата в имеющуюся учетную запись
  # Управление зарегистрированными сертификатами - получить список
  # Управление зарегистрированными сертификатами - удалить сертификат
  # АРМ: Просмотреть список сертификатов для пользователя
  # АРМ: Удаление сертификата у пользователя
Модель данных
  # Описание сущности CertificateCredentials
  # Атрибуты токена
  # Аудит

Назначение

Основным способом входа в онлайн сервисы является вход через логин-пароль, в некоторых инсталляциях усиленный вторым фактором через СМС.

Такой подход имеет преимущества - простота реализации, простота использования.

Из недостатков можно отметить относительно невысокий уровень безопасности ввиду известных уязвимостей паролей (кража, фрод, подбор словарных значений) и СМС (кража телефона, уязвимости протокола ОКС7). Фактически СМС подтверждает не пользователя, а получателя СМС на номер, указанный как контактный.

Для клиентов, у которых востребован более высокий уровень безопасности входа, разработан вход с использованием квалифицированных сертификатов. При их использовании выполняется подтверждение именно физического лица - владельца сертификата.

Способ имеет недостатки - для аутентификации нужно предустанавливать специальное программное обеспечение (плагин для браузера). Скорее всего способ найдет применение в корпоративном сегменте или среди пользователей, уделяющие большое внимание безопасности использования онлайн сервисов.

Описание работы

Аутентификация через сертификаты работает как традиционный PKI с использованием внешнего для UIDM криптографического модуля Jinn Server на серверной стороны и Jinn Client на стороне браузера.

Пользователи заранее выпускают свои сертификаты в удостоверяющем центре (за рамками описания функциональности). Регистрация сертификата в UIDM технически работает как подпись nonce и проверка подписи. После проверки метаданные сертификата сохраняются в БД UIDM как Credentials типа “сертификат” в отдельную таблицу.

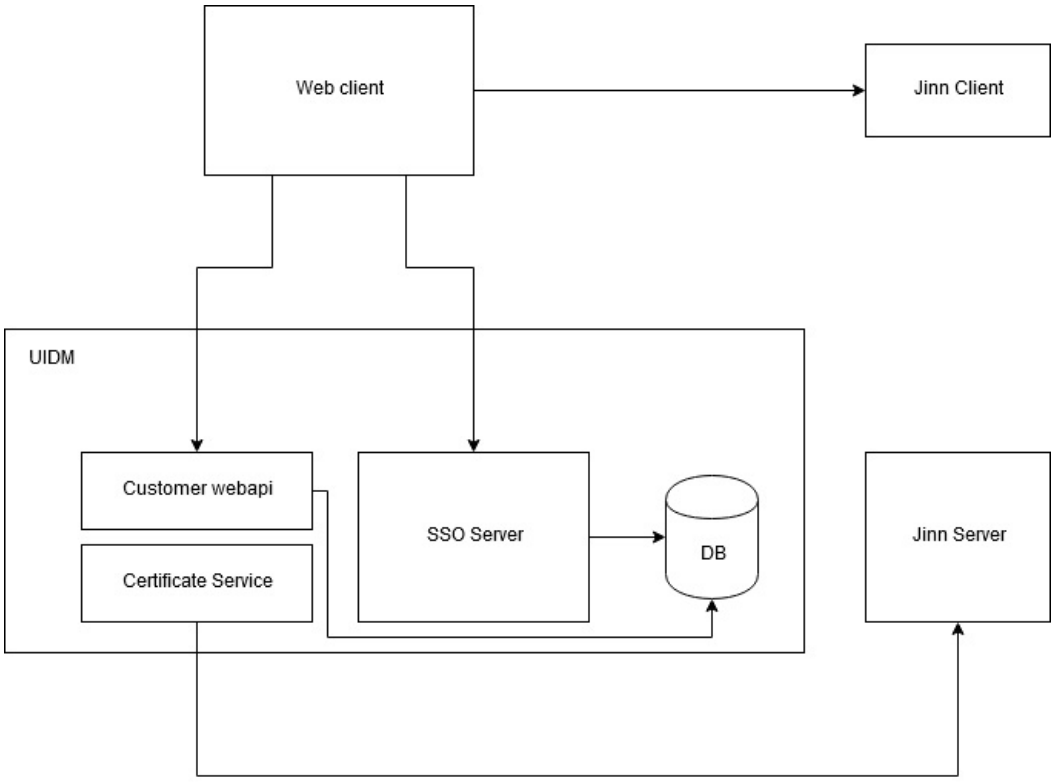
Затем при аутентификации пользователю предлагается войти с использованием сертификата. Технически аутентификация также работает через подписание nonce и проверку подписи с использованием серверного ПО Jinn Server.

Также доступен сценарий регистрации новой учетной записи по сертификату. Технически выполняется как проверка сертификата через подпись nonce, но из метаданных сертификата берется атрибут ИНН организации. Далее сценарий работает как обычная регистрация, но учетная запись получает сразу статус подтвержденной.

Для самообслуживания `Customer WebAPI` имеются RD операциями (по терминологии CRUD) по управлению сертификатами. Сценарии регистрации сертификата, регистрации учетной записи с использованием сертификата и аутентификации являются многошаговыми и реализуются на UIDM на механизме webflow.

Сервер UIDM не выполняет никаких криптографических операций по работе сценариев сам, делегируя проверки в Jinn Server.

Серверные сценарии реализуются без привязки к конкретному поставщику криптографии по подходу “интерфейсы-имплементация”, таким образом может быть разработана реализация и под другие криптопровайдеры.



Сценарии

Пользователь

Физ.лицо получающее доступ к системе.

Клиент

Клиентское фронтенд приложение выполняющееся в браузере пользователя.

Сервер

Серверное приложение, здесь в данном случае под сервером подразумевается обобщенно UIDM с его серверными компонентами..

Аутентификация по сертификату

1. Пользователь открывает страницу входа, выбирает метод аутентификации через сертификат. Клиент отправляет запрос на сервер для инициализации флоу.
2. Сервер генерирует свою часть nonce - serverNonce.
3. Сервер отображает форму валидации сертификата:

```
View data:
  serverNonce
Form data:
  M
  Sign
```

4. Клиент генерирует clientNonce, формирует полный nonce:

```
M = ( clientNonce || serverNonce || serverDomainName )
```

5. Клиент подписывает nonce через браузерный плагин (в настоящее время Jinn Client, но может быть и другое криптоприложение)
6. Клиент отправляет M и подпись (sign) на сервер
7. SSO проверяет на валидность serverNonce и serverDomainName в M. Успешная валидация - шаг 8. Неуспешная валидация - шаг 3 с ошибкой
8. SSO делает запрос в сервис сертификатов на валидацию подписи. Успешная валидация - шаг 9. Неуспешная валидация - возвращаемся к шагу 3 с ошибкой
9. Ищется запись в таблице CertificateCredentials по фингерпринт который вернул сервис валидации. Нашли запись - успешная аутентификация - шаг 10. Не нашли запись - сертификат не найден - возвращаемся к шагу 3 с ошибкой.
10. Продолжается сценарий флоу аутентификация с тем пользователем который был найден по сертификату.

Поведение дальше зависит от конфигурации сервера, может быть задействован 2 фактор или аутентификация может быть завершена сразу с успехом.

Регистрация новой учетной записи по сертификату

1. Пользователь открывает страницу регистрации, выбирает метод аутентификации через сертификат. Клиент отправляет запрос на сервер для инициализации флоу.
2. Сервер генерирует свою часть nonce: serverNonce
3. Сервер отображает форму валидации сертификата:

```
View data:
  serverNonce
Form data:
  M
  sign
```

4. Клиент генерирует clientNonce, формирует полный nonce:

```
M = ( clientNonce || serverNonce || serverDomainName)
```

5. Клиент подписывает nonce через браузерный плагин
6. Клиент отправляет M и подпись (sign) на сервер
7. SSO проверяет на валидность serverNonce и serverDomainName в M. Успешная валидация - шаг 8. Неуспешная валидация - шаг 3 с ошибкой.

8. SSO делает запрос в сервис сертификатов на валидацию подписи. Успешная валидация - шаг 9. Неуспешная валидация - возвращаемся к шагу 3 с ошибкой.
9. Сервер проверяет существует ли сертификат в таблице CertificateCredentials. Существует - авторизуем пользователя, сертификат которого мы провалидировали (переход к шагу 10 из UC-1). Не существует - шаг 10.
10. Сервер получает от сервиса валидации атрибуты сертификата, в том числе ФИО и ИНН организации
11. Сервер отображает форму регистрации

```
View data:
  ФИО
  ИНН организации
  Form data (стандартная форма регистрации пользователя)
```

12. Продолжается флоу регистрации как в стандартном сценарии регистрации.
13. Вместе с созданием принципала и стандартных credentials создается запись в таблице CertificateCredentials с данными сертификата.

Добавление сертификата в имеющуюся учетную запись

Пользователь авторизован и находится в личном кабинете.

1. Пользователь нажимает кнопку “привязать сертификат”. Клиент отправляет запрос на сервер для инициализации флоу.
2. Сервер генерирует свою часть nonce - serverNonce
3. Сервер отображает форму валидации сертификата:

```
View data:
  serverNonce
Form data:
  M
  sign
```

4. Клиент генерирует clientNonce, формирует полный nonce:

```
M = ( clientNonce || serverNonce || serverDomainName)
```

5. Клиент подписывает nonce через браузерный плагин
6. Клиент отправляет M и подпись (sign) на сервер
7. SSO проверяет на валидность serverNonce и serverDomainName в M. Успешная валидация - шаг 8. Неуспешная валидация - шаг 3 с ошибкой.
8. SSO делает запрос в сервис сертификатов на валидацию подписи. Успешная валидация - шаг 9. Неуспешная валидация - шаг 3 с ошибкой. Для данного сценария считается валидным сертификат, который еще не действует на момент добавления (дата начала сертификата - в будущем).
9. Ищем запись в таблице CertificateCredentials по фингерпринт который вернул сервис валидации. Запись не существует - переходим к шагу 10. Запись существует - сертификат уже зарегистрирован на какого-то пользователя - переходим к шагу 3 с ошибкой.
10. Сервер выводит форму подтверждения действия текущим паролем

```
Form data:
  password
```

11. Пользователь вводит пароль и клиент отправляет его на сервер

12. Сервер проверяет пароль. Пароль верный - переходим к шагу 13. Пароль неверный - возврат к шагу 10 с ошибкой.
13. Сохраняем данные сертификата в CertificateCredentials

Управление зарегистрированными сертификатами - получить список

Пользователь должен быть авторизован.

1. Клиент делает запрос на сервис customer webapi

   ```
   GET /customer-webapi/customer/@me/certificates
   ```

2. Сервер по токену пользователя находит сертификаты в таблице CertificateCredentials и возвращает в виде JSON-списка

Управление зарегистрированными сертификатами - удалить сертификат

Пользователь должен быть авторизован.

1. Клиент делает запрос на сервис customer webapi

```
DELETE /customer-webapi/customer/@me/certificates/{certificate_id}
```

1. Сервер по идентификатору находит сертификат в таблице CertificateCredentials, проверяет что сертификат принадлежит пользователю сделавшему запрос и удаляет его (в поле td записывается текущая дата-время, запись считается неактуальной)

АРМ: Просмотреть список сертификатов для пользователя

Сценарий реаловать в новом эндпоинте для SSO Provision API.

Пользователь должен быть авторизован системной ролью.

1. Клиент делает запрос на сервис SSO Provision API

```
GET /sso/provision/customer/{customerId}/certificates
```

1. Сервер по идентификатору клиента {customerId} находит пользователя, находит сертификаты этого пользователя в таблице CertificateCredentials и возвращает в виде JSON-списка

АРМ: Удаление сертификата у пользователя

Пользователь должен быть авторизован.

1. Клиент делает запрос на сервис SSO Provision API

```
DELETE /sso/provision/customer/{customerId}/certificates/{certificate_id}
```

1. Сервер по идентификатору сертификата {certificate_id} находит сертификат в таблице CertificateCredentials, проверяет что сертификат принадлежит пользователю с заданным идентификтором {customerId} и удаляет его (в поле td записывается текущая дата-время, запись считается неактуальной)

Модель данных

Описание сущности CertificateCredentials

Field name	Type	Comment
Id [pk]	VARCHAR	Not null
dsc	VARCHAR	Отладочная информация операции вставки записи
fd	TIMESTAMP	Дата-время создания записи
td	TIMESTAMP	Срок действия записи
ts	TIMESTAMP	Дата-время последнего обновления записи
principalId	VARCHAR	Not null. Идентификатор принципала.
realm	VARCHAR default 'customer'	Not null. Реалм
fingerprint	VARCHAR	Not null. Отпечаток сертификата. Уникальное поле
validFrom	TIMESTAMP	Дата-Время начала действия сертификата
validTill	TIMESTAMP	Дата-Время окончания действия сертификата
extendedAttributes	XMLTYPE	Расширенные атрибуты
providerType	VARCHAR	Not null. Тип поставщика, например JINN
displayName	VARCHAR	Название для UI управления. Формируется на основе полей сертификата (по-умолчанию DN, но может быть разный для разных провайдеров)

ЗАМЕТКА

Обеспечивается историчность данных

Атрибуты токена

При регистрации через сертификаты токен будет содержать authType = 'certificate'.

Аудит

У событий, связанных с аутентификацией, изменяется authType = 'certificate', в поле data добавляется пара: fingerprint = 'certificate fingerprint'.

Добавляются новые типы событий связанные с созданием и удалением сертификатов.