Directum RX

Администрирование Directum Omni

Версия 25.2

Содержание

Введение

В главе содержатся описания установки, настройки и сопровождения Directum Omni в операционной системе на базе Linux. Эти действия выполняет администратор.

1. Перед установкой и настройкой Directum Omni ознакомьтесь с его архитектурой и требованиями к аппаратному и программному обеспечению.
2. Установите Directum Omni.
3. Если планируется использовать *помощника Ario* в *сулерапне Directum Omni*, настройте инструменты для него.
4. При необходимости добавьте в Directum Omni собственные миниаппы. Поддерживается добавление бизнес-процессов Directum RX с помощью средств no-code, а также встраивание сторонних сайтов через элемент iframe.
5. По мере использования продукта контролируйте состояние его компонентов с помощью лог-файлов и выполняйте регламентные работы в соответствии с политикой компании.

Основные понятия

Суперапп Directum Omni

Кроссплатформенное приложение, с помощью которого сотрудники получают доступ к различным корпоративным сервисам, в том числе на базе экосистемы решений Directum. Суперапп сочетает в себе возможности мессенджера, бизнес-приложений и интеллектуального помощника. Он дает сотрудникам единую точку доступа для удобного взаимодействия с коллегами и сервисами компании через интуитивно понятный интерфейс.

Миниапп (Мини-приложение)

Приложение, которое открывается из *супераппа Directum Omni* и нацелено на решение конкретных бизнес-задач. Миниаппы бывают двух видов:

- на базе цифровой платформы Directum RX. Они создаются с помощью инструментов *no-code*, имеют адаптивную верстку и дают доступ к наиболее востребованным действиям с карточками, списками и обложками модулей Directum RX;
- встроенные с помощью iframe. При нажатии на значок такого приложения оно открывается в окне супераппа, сохраняя свой дизайн, верстку и логику. Таким образом можно встроить в суперапп любое веб-приложение.

Обсуждение

Личный или групповой чат *супераппа Directum Omni* в проводнике Directum RX. В отличие от приложения Directum Omni, в проводнике можно создать чат с привязкой к определенному объекту: к документу, задаче, заданию, уведомлению, папке или записи справочника.

Тред

Последовательность ответов на определенное сообщение в чате *супераппа Directum Omni*.

Чат-бот

Бот в специальном чате *супераппа Directum Omni* или *обсуждении* в проводнике. В этом чате пользователь может отправлять запросы *помощнику Ario*.

Интеллектуальный помощник

Интеллектуальный помощник в специальном чате *супераппа Directum Omni* или *обсуждении* в проводнике. С помощью технологий искусственного интеллекта помощник анализирует сообщения пользователя и запускает нужные действия в различных корпоративных сервисах, в том числе на базе экосистемы решений Directum. Возможности интеллектуального помощника зависят от того, какие сервисы подключены, решения установлены и *инструменты* созданы.

Инструмент интеллектуального помощника

Запись справочника **Инструменты интеллектуального чат-бота**, в которой описано действие пользователя в Directum RX, которое можно выполнить с помощью *чат-бота*. Например, создать обращение на замену монитора. По сообщению пользователя в чате *помощник Ario* понимает, к какому инструменту обратиться.

Установка Directum Omni

1. Выполните подготовительные действия.
2. Установите прикладную разработку Directum Omni.
3. Установите сервис идентификации и сервис сообщений.
4. Если планируется использовать *помощника Ario* в *суперапп Directum Omni*, установите сервис Directum Smart Agent.
5. Установите основные сервисы Directum Omni.
6. Настройте подключение к сервисам в Directum RX.
7. Если используются HR Pro или Directum Portal, настройте их совместную работу с Directum Omni.
8. Настройте внешнюю аутентификацию в супераппе, если это требуется в соответствии с политикой безопасности компании.
9. Перезапустите сервисы на каждом сервере с помощью команды:

   ```
   docker-compose -f docker-compose.yaml up -d
   ```
10. Задайте прикладные настройки интеллектуального помощника, если планируется его использование.
11. Подключите сотрудников к Directum Omni. Подробнее см. в руководстве администратора Directum RX, раздел «Настройка подключения к внешним приложениям».
12. При необходимости массово установите приложение Directum Omni на компьютеры пользователей с Microsoft Windows. Для этого можно использовать те же способы, что для массовой установки веб-агента. При этом установка сертификатов на компьютеры пользователей не требуется. Подробнее см. в руководстве администратора Directum RX, раздел «Массовая установка веб-агента (Windows)».

Подготовка к установке

1. Ознакомьтесь с информацией о Directum Omni:
   - требования к аппаратному и программному обеспечению. Подробнее см. в документе «Directum RX 25.2. Типовые требования к аппаратному и программному обеспечению», входит в комплект поставки;
   - архитектура.

   В соответствии с этой информацией определите конфигурацию серверов Directum RX и Directum Omni.
2. Приобретите SSL-сертификаты с проверкой домена для работы по защищенному протоколу HTTPS. Для работы сервисов требуются:
   - для сервиса идентификации – ключевая пара в форматах CRT и PFX;
   - для приложения Directum Omni – ключевая пара в форматах CRT и PFX, а также сертификат в формате PEM. Для конвертации в формат PEM можно использовать, например, библиотеку OpenSSL.

СОВЕТ. Если для Directum RX приобретен Wildcard-сертификат, его также можно использовать для настройки сервисов Directum Omni. В этом случае скопируйте файлы сертификата в нужных форматах с сервера, на котором установлена платформа Directum RX.

3. Убедитесь, что в соответствии с требованиями установлены СУБД для сервисов Directum Omni, и сохраните данные для подключения к ним в удобном виде:
   - PostgreSQL – для сервиса идентификации и сервиса мессенджера;
   - Redis – для сервиса мессенджера.
4. Если планируется вход в суперапп по цифровому коду, который отправляется сотрудникам в SMS, запросите в службе поддержки Directum данные для подключения к провайдеру SMS-сообщений: хост, порт и ключ API.

На серверах, где планируется развернуть сервисы Directum Omni:

1. Подготовьте сертификат для подписания JWT-токенов авторизации.
2. Установите компоненты для разворачивания сервисов:
   - Docker Engine. Подробнее см. в руководстве по установке Directum RX, раздел «Установка Docker Engine»;
   - Docker Compose. Подробнее см. в документации Docker, статья «Overview of installing Docker Compose».
3. Распакуйте дистрибутив Directum Omni в локальную папку. Например, /opt/DirectumOmni.
4. Запросите в службе поддержки Directum данные для подключения к репозиторию Docker-образов registry.directum.ru. Затем авторизуйтесь в реестре Docker-образов с помощью команды:

   ```
   docker login registry.directum.ru
   ```

   ПРИМЕЧАНИЕ. Если установка выполняется на сервере без доступа к интернету, запросите локальные Docker-образы сервисов у вендора и скопируйте файл с ними из дистрибутива на сервер. Затем загрузите образы с помощью команды:

   ```
   docker load -i <Архив с образами>.tar.gz
   ```

Подготовка сертификата для подписания токенов

Для авторизации запросов сервис идентификации выпускает JWT-токены и подписывает их с помощью специального сертификата. Если планируется использование миниаппов HR Pro и Directum Portal, то для сервиса идентификации в составе этих продуктов и сервиса идентификации Directum Omni рекомендуется использовать один и тот же сертификат. Это нужно, чтобы сотрудникам не требовалось заново проходить аутентификацию при входе в миниаппы.

Если в компании уже используются HR Pro или Directum Portal, создавать новый сертификат для Directum Omni не требуется. Вместо этого скопируйте уже используемый сертификат на серверы, где планируется установка сервисов Directum Omni:

- контейнер с закрытым ключом сертификата – на сервер, где планируется установить сервис идентификации;
- открытый ключ сертификата – на все серверы, где планируется установка сервисов. Также убедитесь, что открытый ключ размещен на серверах с Directum RX.

Если продукты HR Pro и Directum Portal не используются, создайте сертификат для подписания токенов. Для этого:

1. На сервере, где планируется установить сервисы Directum Omni, установите пакет OpenSSL с помощью менеджера пакетов, который применяется в используемой операционной системе. Например, для Ubuntu выполните команду:

   ```
   sudo apt update && sudo apt install openssl
   ```

2. Создайте локальную папку для хранения сертификатов. Например, /opt/certificates. Дальнейшие команды выполняйте из этой папки.
3. Для сервиса идентификации создайте открытый ключ сертификата в формате CRT и контейнер с закрытым ключом в формате PFX с помощью команд:
   - команды создания ключевой пары:

     ```
     openssl genrsa -des3 -passout pass:<Пароль для ключа> -out ids.pass.key
     2048
     openssl rsa -passin pass:<Пароль для ключа> -in ids.pass.key -out ids.key
     openssl req -new -key ids.key -out ids.csr
     ```

     На этом этапе запрашивается дополнительная информация: страна, город, название подразделения и пр. Укажите информацию в произвольном формате;

   - команды создания сертификата:

     ```
     openssl x509 -req -sha256 -days 365 -in ids.csr -signkey ids.key -out ids-
     jwt.crt
     openssl pkcs12 -export -certpbe PBE-SHA1-3DES -keypbe PBE-SHA1-3DES -out
     ids-jwt.pfx -inkey ids.key -in ids-jwt.crt -passout pass:<Пароль для ключа>
     ```

   При необходимости измените настройки, выделенные жирным шрифтом, в соответствии с параметрами системы. Подробное описание параметров см. в документации OpenSSL.

   ПРИМЕЧАНИЕ. В параметре **days** указывается срок действия сертификата в днях. Этот срок определяется индивидуально с учетом политики безопасности компании.
4. Разместите ключевую пару на серверах:
   - контейнер с закрытым ключом сертификата – на сервере, где планируется установить сервис идентификации;
   - открытый ключ сертификата – на всех серверах, где планируется установка сервисов, и на сервер с Directum RX.

Установка прикладной разработки

Для установки используется кроссплатформенный инструмент Directum Launcher. Подробнее об инструменте см. в руководстве администратора, раздел «Установка системы (Directum Launcher)» (Linux, Windows).

Прикладную разработку Directum Omni можно установить:

- вместе с серверной частью Directum RX;
- дополнительно к Directum RX, если платформа была развернута ранее.

Порядок установки прикладной разработки в операционных системах Linux и Windows совпадает, при этом расширение архивов и синтаксис команд отличается в зависимости от операционной системы.

Пример команды:

Linux

```
./do.sh <команда>
```

Windows

```
do <команда>
```

В разделах приведен порядок установки на Linux.

Установка одновременно с Directum RX

Установка выполняется в визуальном режиме или с помощью командной строки.

Перед началом установки:

1. Архив с Directum Launcher распакуйте в локальную папку на сервере с помощью команды:

   ```
   tar -xvf <Имя архива> -C <Имя папки>
   ```

2. В корень папки скопируйте архивы с компонентами Directum RX. Подробнее см. в инструкции по установке Directum RX, раздел «Серверная часть Directum RX».
3. В локальную папку на сервере скопируйте пакет разработки omni.dat.

   ВАЖНО. Если планируется установить другую прикладную разработку, в которой перекрыт модуль «Компания», подготовьте общий пакет с этой разработкой и omni.dat. Используйте этот пакет при установке вместо omni.dat. Подробнее см. в руководстве разработчика Directum RX, раздел «Экспорт разработки».
4. Увеличьте максимально допустимое количество наблюдателей за файлами на текущем компьютере в системном параметре /proc/sys/fs/inotify/max_user_instances. Для этого из папки Directum Launcher выполните команду:

   ```
   ./do.sh set_inotify_instances_limit
   ```

5. Дальнейшие действия выполняйте под учетной записью пользователя, от имени которого запущена служба Docker. Если используется учетная запись без привилегий суперпользователя, то выдайте пользователю полные права на папку с Directum Launcher и вложенные в нее папки и файлы.
6. При необходимости задайте параметры:
   - **extra_hosts**, если в компании не используется DNS-сервер;
   - **is_rootless**, если служба Docker запускается в режиме Rootless.

   Подробнее см. в инструкции по установке Directum RX, раздел «Серверная часть Directum RX».

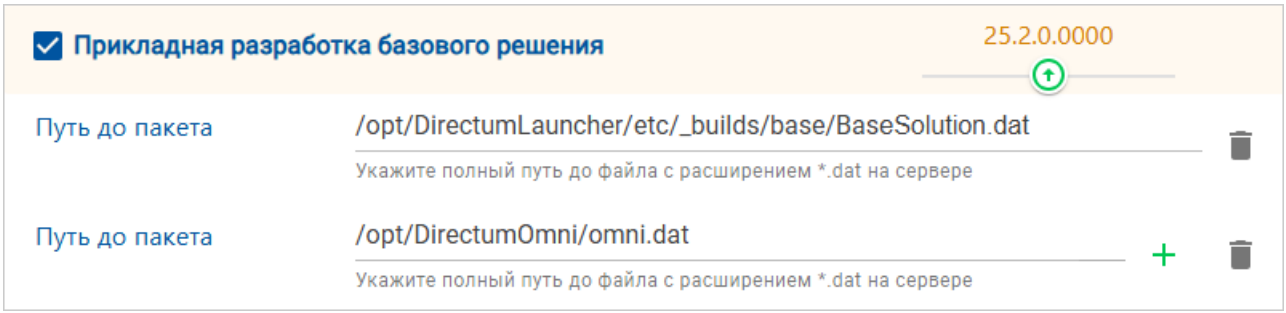
Установка в визуальном режиме

1. Запустите Directum Launcher с помощью команды:

   ```
   ./DirectumLauncher --host=0.0.0.0
   ```

   В командной строке выводится ссылка на страницу Directum Launcher.
2. Перейдите по полученной ссылке в браузере. Убедитесь, что на открывшейся странице установлен переключатель **Установка**.

3. Укажите настройки веб-сервера и остальных компонентов Directum RX. Подробнее см. в руководстве по установке Directum RX, раздел «Серверная часть Directum RX».
4. Убедитесь, что установлен флажок **Прикладная разработка базового решения** и указан путь до пакета разработки с базовым решением Directum RX.
5. Нажмите на кнопку + и в открывшемся окне укажите путь до пакета omni.dat:



6. Убедитесь, что установлен флажок **Справка о системе**.
7. Ознакомьтесь с текстом лицензионного соглашения и установите флажок **Я принимаю условия лицензионного соглашения**.
8. Нажмите на кнопку **Установить** и дождитесь окончания установки.

Установка с помощью командной строки

1. Добавьте компоненты для установки с помощью команды:

   ```
   ./do.sh components add_all
   ```

2. Выполните подготовительные действия и установите платформу Directum RX. Подробнее см. в руководстве по установке Directum RX, раздел «Установка с помощью командной строки».
3. Вместе с базовым решением Directum RX опубликуйте пакет omni.dat. Для этого выполните команду:

   ```
   ./do.sh base install --package="<Путь к пакету omni.dat>
   ```

   Пример команды:

   ```
   ./do.sh base install --package="/opt/DirectumOmni/omni.dat"
   ```

Установка дополнительно к Directum RX

Установка выполняется в визуальном режиме или с помощью командной строки.

Перед началом установки:

1. Создайте резервную копию используемой базы данных Directum RX.
2. В локальную папку на сервере скопируйте пакет разработки omni.dat.

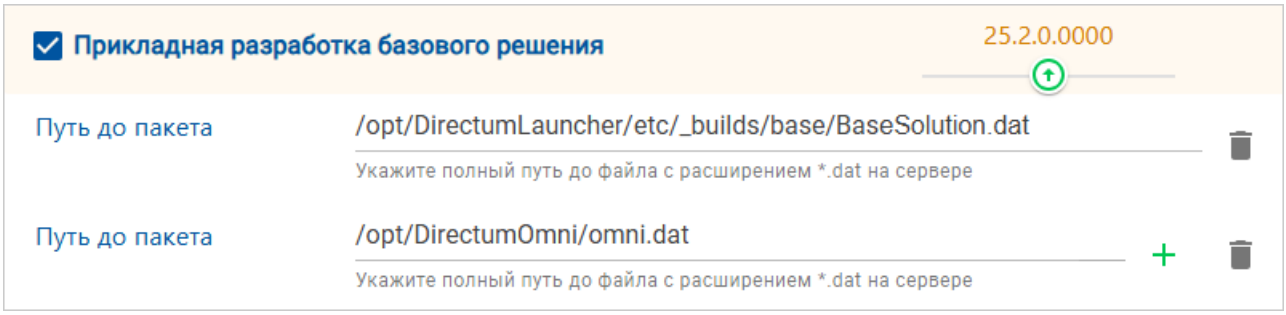
   ВАЖНО. Если в компании используется прикладная разработка, в которой перекрыт модуль «Компания», подготовьте общий пакет с этой разработкой и omni.dat. Используйте этот пакет при установке вместо omni.dat. Подробнее см. в руководстве разработчика Directum RX, раздел «Экспорт разработки».
3. Дальнейшие действия выполняйте под учетной записью пользователя, от имени которого запущена служба Docker. Если используется учетная запись без привилегий суперпользователя, то выдайте пользователю полные права на папку с Directum Launcher и вложенные в нее папки и файлы.

Установка в визуальном режиме

1. Запустите Directum Launcher с помощью команды:

   ```
   ./DirectumLauncher --host=0.0.0.0
   ```

   В командной строке выводится ссылка на страницу Directum Launcher.
2. Перейдите по полученной ссылке в браузере. Убедитесь, что на открывшейся странице установлен переключатель **Обновление** и снят флажок **Платформа Sungero**.
3. Убедитесь, что установлен флажок **Прикладная разработка базового решения** и указан путь до пакета разработки с базовым решением Directum RX.
4. Нажмите на кнопку + и в открывшемся окне укажите путь до пакета omni.dat:



5. Убедитесь, что установлен флажок **Справка о системе**.
6. Ознакомьтесь с текстом лицензионного соглашения и установите флажок **Я принимаю условия лицензионного соглашения**.
7. Нажмите на кнопку **Обновить** и дождитесь окончания установки.

Установка с помощью командной строки

Переопубликуйте базовое решение Directum RX вместе с пакетом разработки omni.dat. Для этого выполните команду:

```
./do.sh dt deploy --package="<Путь к пакету с базовым решением>";"<Путь к пакету omni.dat>"
```

Пример команды:

```
./do.sh dt deploy --
package="/opt/DirectumLauncher/etx/_builds/base/BaseSolution.dat";"/opt/DirectumOmni/omni.dat"
```

Установка сервиса идентификации и сервиса сообщений

Чтобы установить сервисы и подготовить их к работе:

1. Настройте сервис сообщений.
2. Настройте сервис идентификации.
3. Создайте служебные учетные записи в сервисе идентификации.
4. Запустите контейнеры с сервисами и проверьте их работоспособность.

В дистрибутив Directum Omni включены шаблоны:

- файл docker-compose-ids.yaml для разворачивания контейнеров с сервисами;
- файлы в формате ENV с настройками сервисов.

В этих файлах уже заполнена часть настроек. Измените значения тех параметров, которые указаны в соответствующих разделах инструкции. Для остальных параметров оставьте значения по умолчанию.

Строки подключения

Для строк подключения указаны шаблоны, в которых нужно заполнить данные:

- **Host** и **Port** – адрес хоста и порт компонента;
- **User ID** и **Password** – данные сервисной учетной записи для подключения к сервису идентификации, базе данных или Directum RX;
- **Database** – название базы данных.

Не изменяйте значения параметров **Name** и **UseSsl**.

Пример строки подключения к сервису идентификации:

```
ConnectionStrings__IdentityService:
'Name=Directum.Core.IdentityService;Host=id.contoso.ru;UseSsl=true;Port=443;User
ID=ExampleUser;Password=ExamplePassword;'
```

Установка в закрытый контур

Для установки сервисов в закрытом контуре используйте образы, загруженные при подготовке к установке. В этом случае в файле docker-compose.yml в параметре image указывается путь до образа:

```
services:
  <Имя сервиса>:
    container_name: <Имя контейнера>
    image: <Имя образа>: <Тег>
```

Список загруженных образов и их тегов можно посмотреть с помощью команды:

```
docker image
```

Настройка сервиса сообщений

1. В СУБД PostgreSQL создайте базу данных **Messages**. Инициализируйте ее с помощью скрипта из дистрибутива init/mb_init.sql.
2. В конфигурационном файле docker-compose-ids.yaml заполните секцию **message-broker**:

```
  message-broker:
    container_name: message-broker
    image: <Имя и версия образа из шаблона конфигурационного файла>
      entrypoint: sh -c 'cp /certificates/*.crt /etc/pki/ca-trust/source/anchors/ &&
update-ca-trust && dotnet Core.MessageBroker.WebApiService.dll'
    env_file:
      - message-broker.env
    ports:
    volumes:
      - /opt/certificates:/certificates:ro
      - /opt/logs:/app/Logs
```

3. В конфигурационном файле message-broker.env в параметре **ConnectionStrings__Database** заполните строку подключения к базе данных в PostgreSQL. В конфигурационном файле из дистрибутива заполнен шаблон строки, в котором нужно указать данные для подключения.
4. Убедитесь, что значения параметров указаны верно:
   - **Authentication__TrustedIssuers__0__Issuer** – имя издателя токенов для авторизации запросов;
   - **Authentication__TrustedIssuers__0__EncryptionKey** – ключ шифрования для токенов или **Authentication__TrustedIssuers__0__SigningCertificatePath** – путь до открытого ключа JWT-сертификата внутри Docker-контейнера. Пример: /certificates/ids jwt.crt.

   Необходимо задать только один из указанных параметров. Если указаны оба, для подписания токенов используется параметр **Authentication__TrustedIssuers__0__SigningCertificatePath**.

   ВАЖНО. Имя издателя и способ выпуска токенов должны быть такими же, как в настройках сервиса идентификации.
5. Если планируется вход в суперапп по цифровому коду, который отправляется в SMS, то укажите данные для подключения к провайдеру отправки SMS-сообщений:
   - **Transport__Proxies__SmsProxyTransportPlugin__Host** – имя хоста для подключения к сервису SmsProxy;
   - **Transport__Proxies__SmsProxyTransportPlugin__Port** – номер порта для подключения к сервису SmsProxy;
   - **Transport__Proxies__SmsProxyTransportPlugin__ApiKey** – API ключ для авторизации в сервисе SmsProxy.

Настройка сервиса идентификации

1. В СУБД PostgreSQL создайте базу данных **Identities**. Инициализируйте ее с помощью скрипта из дистрибутива init/ids_init.sql.
2. В папке /opt/certificates разместите сертификаты, подготовленные до начала установки:
   - JWT-сертификат сервиса идентификации для подписания токенов;
   - SSL-сертификат сервиса идентификации.

   Для каждого сертификата скопируйте открытый ключ в формате CRT и контейнер с закрытым ключом в формате PFX.
3. В конфигурационном файле docker-compose-ids.yml заполните секцию **identity-service**:

```
  identity-service:
    container_name: identity-service
    image: <Имя и версия образа из шаблона конфигурационного файла>
      entrypoint: sh -c 'cp /certificates/*.crt /etc/pki/ca-trust/source/anchors/ &&
update-ca-trust && dotnet Directum.IdentityService.dll'
    env_file:
      - identity-service.env
    ports:
      - "443:443"
    volumes:
      - /opt/certificates:/certificates:ro
      - /opt/logs:/app/Logs
```

4. В конфигурационном файле identity-service.env настройте адрес для доступа к сервису и SSL-сертификат. Для этого укажите значение параметров:
   - **General__ServiceEndpoint** – внешний адрес сервиса идентификации, по которому он будет доступен пользователям. Например, https://id.contoso.ru;
   - **ASPNETCORE_Kestrel__Certificates__Default__Path** – путь до SSL-сертификата сервиса внутри контейнера. Пример: /certificates/ids ssl.pfx;
   - **ASPNETCORE_Kestrel__Certificates__Default__Password** – пароль от сертификата.
5. Задайте строки подключения:
   - **ConnectionStrings__Database** – к базе данных сервиса идентификации;
   - **ConnectionStrings__MessagingService** – к сервису сообщений.

   В конфигурационном файле из дистрибутива заполнены шаблоны строк, в которых нужно указать данные для подключения.
6. Задайте настройки выпуска токенов для аутентификации запросов к сервисам:
   - **TokenIssuer__Issuer** – имя издателя токенов. Укажите значение **OmniId**;
   - **TokenIssuer__EncryptionKey** – ключ шифрования для токенов или **TokenIssuer__SigningCertificatePfxPath** – путь до контейнера закрытого ключа для подписания токенов внутри Docker-контейнера. Пример: /certificates/ids jwt.pfx.

     Необходимо задать только один из указанных параметров. Если указаны оба, для подписания токенов используется параметр **TokenIssuer__SigningCertificatePfxPath**.
   - **TokenIssuer__SigningCertificatePfxPassword** – пароль от контейнера. Указывается, если заполнен путь до контейнера.

   ВАЖНО. Имя издателя и способ выпуска токенов должны быть одинаковыми в настройках всех сервисов.
7. В параметре **Security__DataProtectorKey** укажите ключ для хеширования данных. Он должен быть криптографически случайным с минимальной длиной 256 бит (32 байта). Рекомендуемый формат – Base64.

   ВАЖНО. Ключ задается однократно при установке сервиса и не должен меняться. Замена ключа приведет к невозможности расшифровки ранее сохраненных данных.

   Сгенерировать ключ можно, например, с помощью библиотеки OpenSSL. Для этого выполните команду:

   ```
   openssl rand -base64 32
   ```
8. По умолчанию сотрудники могут входить в Directum Omni только по номеру телефона. При необходимости можно добавить возможность входа по адресу электронной почты. Для этого в конфигурационный файл добавьте параметр **UserAccounts__IdentityCredentials__1** и укажите в нем значение **Email**:

   ```
   UserAccounts__IdentityCredentials__1: "Email"
   ```

Создание учетных записей и регистрация ресурсов в сервисе идентификации

Для корректной работы сервисов Directum Omni в базу данных сервиса идентификации необходимо добавить:

- служебные учетные записи, от имени которых отправляются запросы между сервисами Directum Omni;
- формат данных о пользователях (claims), который используется разными ресурсами, и правила преобразования таких данных.

Для работы с базой данных сервиса идентификации используется утилита id. Чтобы подготовить ее к работе:

1. Из папки с распакованным дистрибутивом, подпапки /distrib/Tools извлеките содержимое архива Id-Cli-linux-x64.zip в любую локальную папку сервера.
2. В созданной папке в подпапке Id-Cli откройте конфигурационный файл утилиты appsettings.json. Укажите в нем:
   - в секции **ConnectionStrings** в строке подключения **Database** – параметры для подключения к базе данных сервиса идентификации;
   - в секции **Security** в параметре **DataProtectorKey** – ключ для хеширования данных, заданный в настройках сервиса идентификации.
3. Выдайте пользователям права для работы с утилитой с помощью команды:

   ```
   chmod 777 /<путь до папки Id-Cli>/id
   ```
4. Добавьте для утилиты псевдоним, позволяющий запускать ее из любой папки. Для этого откройте на редактирование файл /etc/bash.bashrc и добавьте в конец файла строку:

   ```
   alias id-cli=/<путь до папки Id-Cli>/id
   ```

   Сохраните файл, затем примените настройки с помощью команды:

   ```
   source /etc/bash.bashrc
   ```
5. Добавьте сервисную роль в базу данных сервиса. Для этого выполните команду:

   ```
   id-cli add role service
   ```

   Если настройка корректна, будет получен ответ: **Role added successfully**.

Создание учетных записей

Создайте учетные записи для сервисов:

- сервисы Directum RX;
- сервис мессенджера;
- API-шлюз;
- BFF-сервер;
- сервис супераппа;
- чат-бот. Имя учетной записи для него должно быть **OmniBot**.

Чтобы создать учетные записи:

1. Поочередно для каждого сервиса выполните команду:

   ```
   id-cli add user "<Имя учетной записи>" -p password="<Пароль служебного пользователя>"
   ```

   Пример для сервиса супераппа:

   ```
   id-cli add user "SuperappServiceUser" -p password="!$xqoodZ9S"
   ```

   СОВЕТ. Сохраните данные учетных записей в удобном виде. Их необходимо указать в настройках сервисов.

2. Включите учетные записи в роль service с помощью команды:

   ```
   id-cli assign -u "<Имя учетной записи>" -r "service"
   ```

   Пример для сервиса супераппа:

   ```
   id-cli assign -u "SuperappServiceUser" -r "service"
   ```

Регистрация ресурсов

Поочередно зарегистрируйте ресурсы в сервисе идентификации с помощью команд:

- для сервиса сообщений:

  ```
  id-cli add resource "Directum.Core.MessageBroker" -c "<Папка с распакованным дистрибутивом Directum Omni>\MessageBrokerAudience.json"
  ```

- для Directum Omni:

  ```
  id-cli add resource "Directum Omni" -c "<Папка с распакованным дистрибутивом Directum Omni>/OmniAudience.json"
  ```

- для сервиса PublicAPI Directum RX:

  ```
  id-cli add resource "PublicApi" -c "<Папка с распакованным дистрибутивом Directum Omni>/PublicApiAudience.json"
  ```

- если планируется использование помощника Ario – для сервиса интеграции Directum RX:

  ```
  id-cli add resource "IntegrationService" -c "<Папка с распакованным дистрибутивом Directum Omni>/IntegrationServiceAudience.json"
  ```

Запуск и проверка работоспособности сервисов

После настройки сервиса идентификации и сервиса сообщений:

1. Разверните Docker-контейнеры с сервисами. Для этого из папки с файлом docker-compose.yml последовательно выполните команды:

   ```
   sudo docker compose -f docker-compose.yml up -d message-broker
   sudo docker compose -f docker-compose.yml up -d identity-service
   ```

2. В браузере перейдите на страницу: https://<адрес сервиса идентификации>/ready.

   Запрос возвращает информацию о состоянии сервиса в формате JSON. Убедитесь, что в каждой секции значение параметра **Status** равно **Healthy**.

Установка сервиса Directum Smart Agent

Для установки сервиса Directum Smart Agent используется инструмент Directum Launcher.

Перед установкой ознакомьтесь с системными требованиями сервиса. Подробнее см. в документации Directum RX в типовых требованиях к аппаратному и программному обеспечению.

Порядок установки Directum Smart Agent в операционных системах Linux и Windows совпадает. При этом расширение архивов и синтаксис команд отличается в зависимости от операционной системы.

Пример команды:

Linux

```
./do.sh <команда>
```

Windows

```
do <команда>
```

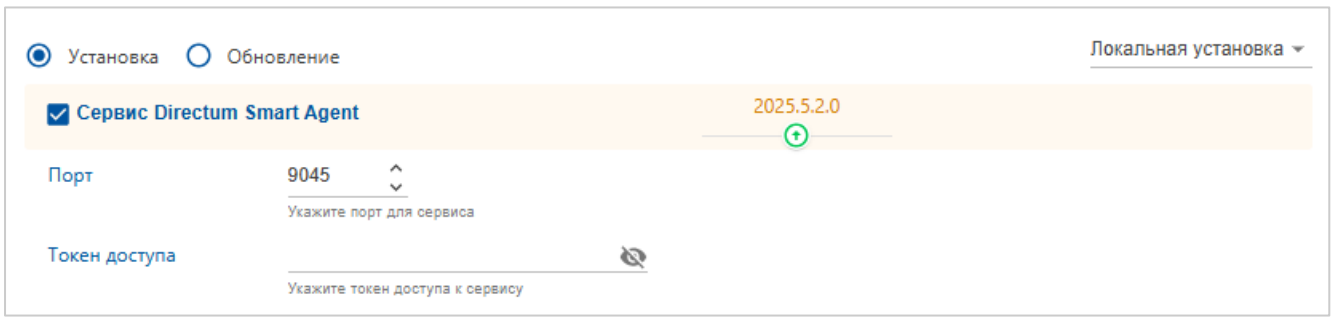
В разделе приведен порядок установки на Linux.

Установите сервис одним из способов:

- в визуальном режиме Directum Launcher
- с помощью командной строки

Установка в визуальном режиме

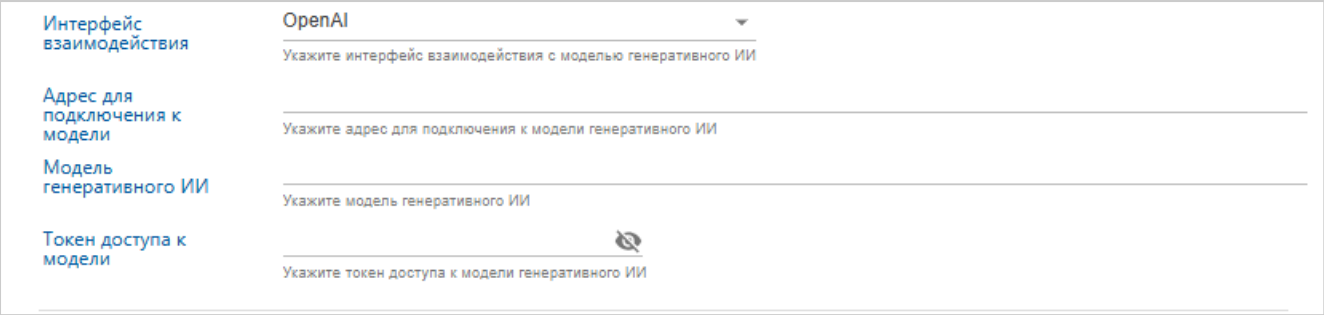
1. Архив с Directum Launcher распакуйте в локальную папку на сервере. Подробнее см. в документации Directum RX в руководстве администратора, раздел «Серверная часть Directum RX».
2. В корень папки скопируйте архив smartagent.tar.gz. Если на сервере отсутствует доступ к интернету, также скопируйте архив smartagentImage.tar.gz, в котором находится базовый образ сервиса.
3. Запустите Directum Launcher.
4. На странице **Развертывание** убедитесь, что установлен переключатель **Установка**.
5. Установите флажок **Сервис Directum Smart Agent** и укажите основные настройки сервиса:



   Порт, по которому доступен сервис Directum Smart Agent. Значение по умолчанию **9045**.

   Токен доступа, который передается в запросах к сервису Directum Smart Agent. Укажите произвольную строку. Это же значение необходимо указать в настройках чат-бота в конфигурационном файле docker-compose.yml в параметре **AISettings_AIToken**;

6. Укажите настройки подключения к сервису генеративного ИИ:



Интерфейс взаимодействия. Способ формирования запросов от пользователя. В выпадающем списке выберите значение **OpenAI**.

Адрес для подключения к модели генеративного ИИ. Задается в формате:

https://<IP адрес сервера>/v1

Где:

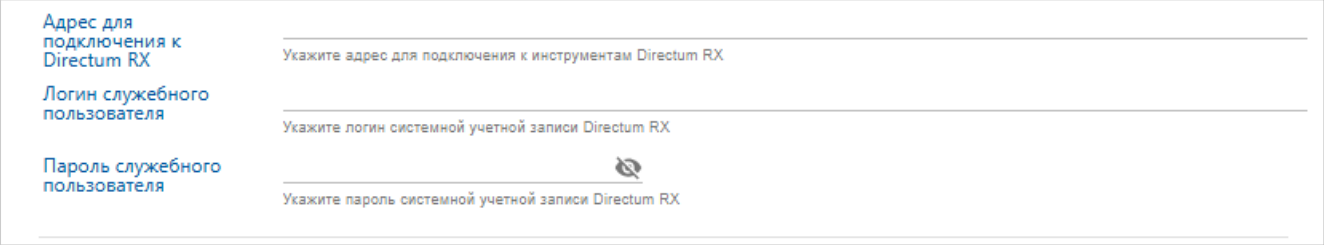
IP-адрес сервера – адрес сервера, на котором расположен сервис генеративного ИИ;

v1 – обязательный постфикс для корректной обработки запросов.

Токен доступа к модели генеративного ИИ. Если планируется использовать сервис Directum LLM, укажите значение, заданное при установке сервиса.

Модель генеративного ИИ. Наименование модели генеративного ИИ, которая будет использоваться по умолчанию.

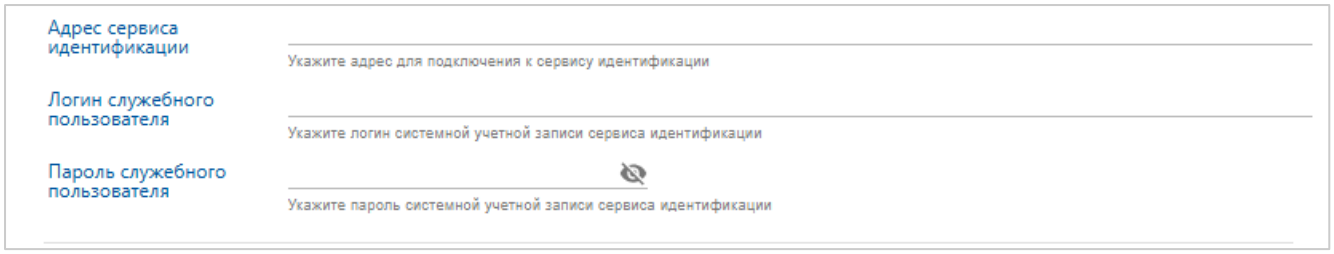
7. По умолчанию *помощник Ario* может взаимодействовать с корпоративными сервисами, которые работают по протоколу MCP. Directum RX работает по протоколу HTTP или HTTPS. Для взаимодействия помощника с Directum RX дополнительно выполняется преобразование HTTP-запросов в формат MCP. Укажите настройки подключения к Directum RX:



Адрес для подключения к Directum RX. Адрес для подключения к сервису интеграции в формате https://<Наименование сервера>:<Протокол>/<Имя сервиса интеграции>/odata/IntegrationAIAgent/HandleMCPRequests. Если для сервиса интеграции используется имя по умолчанию **Integration**, в поле укажите значение **https://<Наименование сервера>:<Протокол>/Integration/odata/IntegrationAIAgent/HandleMCPRequests**. Подробнее см. в документации Directum RX в руководстве администратора, раздел «Минимальные настройки системы».

Логин служебного пользователя и **Пароль служебного пользователя**. Логин и пароль системной учетной записи Directum RX для получения системных данных, например для получения списка инструментов.

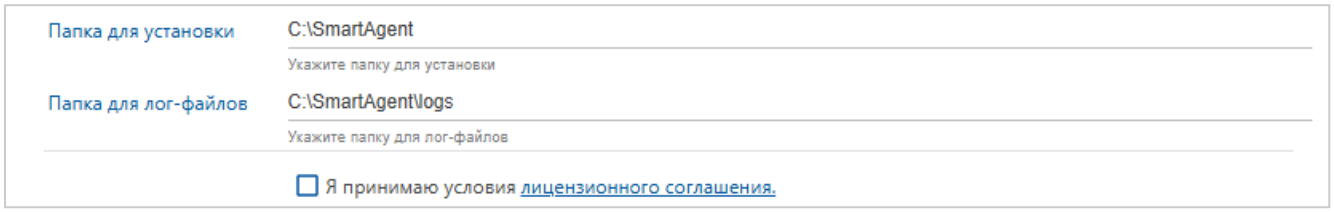
8. Укажите настройки подключения сервиса Directum Smart Agent к сервису идентификации Directum RX:



Адрес сервиса идентификации. Адрес для подключения к сервису идентификации.

Логин служебного пользователя и **Пароль служебного пользователя**. Логин и пароль системной учетной записи сервиса идентификации.

9. Укажите папки сервиса:



Папка для установки. При необходимости укажите относительный или полный путь до папки, в которую нужно установить Directum Smart Agent. Путь по умолчанию: /opt/directum/smartagent.

Папка для лог-файлов. При необходимости укажите относительный или полный путь до папки, в которую будут сохраняться лог-файлы. Путь по умолчанию: /opt/directum/smartagent/logs.

10. По умолчанию в Directum Launcher задаются минимально необходимые настройки сервиса. Чтобы изменить дополнительные, последовательно выполните шаги:
    - сгенерируйте конфигурационный файл. Для этого в Directum Launcher нажмите на кнопку **Сохранить**;
    - откройте страницу **Настройка** и измените положение переключателя **Визуальный режим конфигурирования**. Подробнее см. в документации Directum RX в руководстве администратора, раздел «Встроенный редактор YAML»;
    - внесите изменения в нужные параметры config.yml;
    - нажмите на кнопку **Сохранить**. Измененные настройки запишутся в конфигурационный файл config.yml.
    - вернитесь на страницу **Развертывание** и убедитесь, что установлен переключатель **Установка**.
11. Ознакомьтесь с текстом лицензионного соглашения и установите флажок **Я принимаю условия лицензионного соглашения**.
12. Нажмите на кнопку **Установить** и дождитесь окончания установки. На странице выводятся этапы установки. В раскрывающейся области с названием этапа отображаются сообщения из лог-файла установки. Во время установки могут появляться дополнительные окна. Чтобы корректно завершить процесс, не закрывайте их.

Установка с помощью командной строки

1. Распакуйте архив с Directum Launcher в локальную папку на сервере с помощью команды:

```
tar -xvf <Имя архива> -C <Имя папки>
```

ВАЖНО. Для корректной установки общий путь к файлам должен быть не более 256 символов. Также он не должен содержать пробелы, символы кириллицы, запятые и спецсимволы. Поэтому используйте, например, папку /srv/DirectumLauncher. В зависимости от настроек операционной системы для дальнейших действий могут потребоваться права суперпользователя.

2. В корень папки с Directum Launcher скопируйте архив smartagent.tar.gz. Если на сервере отсутствует доступ к сети Интернет, также скопируйте архив smartagentImage.tar.gz, в котором находится базовый образ сервиса.
3. Добавьте компоненты в Directum Launcher. Для этого выполните команду:

   ```
   ./do.sh components add_all
   ```
4. Создайте конфигурационный файл config.yml на основе файла DirectumLauncher/etc/config.yml.example.
5. Сгенерируйте настройки установки сервиса с помощью команды:

   ```
   ./do.sh smartagent generate_config_yaml
   ```
6. Откройте конфигурационный файл config.yml и настройте его параметры.
7. Установите сервис Directum Smart Agent. Для этого выполните команду:

   ```
   ./do.sh smartagent install
   ```

Параметры установки Directum Smart Agent

В разделе приведен полный список параметров установки и настройки сервиса Directum Smart Agent, которые можно использовать в конфигурационном файле config.yml.

ПРИМЕЧАНИЕ. После установки сервиса параметры конфигурационного файла можно изменить. Для этого измените значения нужных параметров и выполните команды:

```
./do.sh smartagent config_up
./do.sh smartagent restart
```

Настройки сервиса

Переменные для работы с сервисом задаются в секции **variables**:

- **DSA_INSTALL_PATH** – папка, в которую нужно установить сервис. Необязательный параметр. Значение по умолчанию:

  Linux **~/directum/SmartAgent/**

  Windows **C:/SmartAgent/**
- **DSA_LOG_PATH** – папка, в которую сохраняются лог-файлы. Необязательный параметр. Значение по умолчанию:

  Linux **~/directum/SmartAgent /logs/**

  Windows **C:/SmartAgent/logs/**

Параметры задаются в секции **services_config** в секции **DirectumSAService**:

- **PORT** – порт для подключения к сервису. Значение по умолчанию **9035**;

- **LOG_PATH** – относительный или полный путь до папки с лог-файлами. По умолчанию они записываются в папку DirectumSAService, которая расположена в папке, указанной в параметре **DSA_LOG_PATH**;
- **ACCESS_TOKEN** – токен доступа, который передается в запросах к сервису Directum Smart Agent. Укажите произвольную строку. Это же значение необходимо указать в настройках чат-бота в конфигурационном файле docker-compose.yml в параметре **AISettings_AIToken**.

Настройки логирования

Параметры задаются в секции **DirectumSAService** в секции **LOGGING**:

- **PERIOD** – периодичность создания новых лог-файлов. Возможные значения:

  per_minute – каждую минуту;

  per_hour – каждый час;

  per_day – каждый день;

  midnight – каждый день в полночь.

  Значение по умолчанию **midnight**;
- **SUFFIX** – формат названия лог-файла. Значение по умолчанию **%Y%m%d**;
- **FILENAME** – имя лог-файла;
- **PATH** – путь до папки с лог-файлами;
- **LEVEL** – минимальный уровень логирования. Возможные значения:

  Debug – отладочные сообщения. Помогают разработчику восстановить ход работы по этапам конкретного процесса в продуктивной системе;

  Information – логирование значимых действий сервиса. По ним администратор может отследить всю последовательность действий до возникновения ошибки и устранить ее;

  Warning – предупреждения. При возникновении ошибок такого уровня во время действий они не прерываются;

  Error – все ошибки в работе сервиса Directum Smart Agent.

  Значение по умолчанию **Debug**;
- **WRITE_TO_ROLLING_FILE_ENABLED** – признак записи лог-файла в файл. Возможные значения: **True**, **False**. Значение по умолчанию **True**;
- **WRITE_TO_ROLLING_FILE_FORMAT** – формат лог-файла. Возможные значения: **text**, **json**. Значение по умолчанию **json**;
- **WRITE_TO_STDOUT_ENABLED** – разрешить выводить лог-файл через командную строку. Возможные значения: **True**, **False**. Значение по умолчанию **True**;
- **WRITE_TO_STDOUT_FORMAT** – формат вывода лог-файла через командную строку. Возможные значения: **text**, **json**. Значение по умолчанию **text**.

Подключение к модели генеративного ИИ

Параметры задаются в секции **DirectumSAService** в секции **LLM**:

- **URL** – адрес для подключения к модели генеративного ИИ. Задается в формате:

  https://<IP адрес сервера>/v1

  Где:

IP-адрес сервера – адрес сервера, на котором расположена модель;

v1 – обязательный постфикс для корректной обработки запросов;

- **ACCESS_TOKEN** – токен доступа, который передается в запросах к сервису генеративного ИИ. Если планируется использовать сервис Directum LLM, укажите значение, заданное при установке сервиса;
- **MAX_OUTPUT_TOKENS** – максимальное количество токенов в ответе на запрос. Значение по умолчанию **1024**;
- **TEMPERATURE** – креативность ответа модели на запрос. Значение параметра указывается в интервале от **0** до **2**. Например, при значении **0,1** модель формирует наиболее вероятные и точные ответы, а при значении **2** – креативные и разнообразные ответы;
- **MODEL** – наименование модели генеративного ИИ, которая будет использоваться по умолчанию.

Фильтрация используемых инструментов

При обработке сообщения пользователя Directum Smart Agent проверяет, насколько каждый инструмент подходит для выполнения запроса и присваивает коэффициент. Затем все подобранные инструменты отправляются в модель генеративного ИИ, которая выбирает среди них нужный.

Параметры коэффициентов задаются в секции **DirectumSAService** в секции **SELECTOR**:

- **MODEL** – модель генеративного ИИ, которая фильтрует инструменты. Оставьте значение по умолчанию **./models/deepvk/USER-bge-m3**;
- **MIN_SCORE_TO_SELECT** – минимальный коэффициент инструмента, который выбирается для выполнения запроса пользователя. Рекомендуется оставить значение по умолчанию **0,67**;
- **SCORE_EPSILON** – максимальная разница между коэффициентами инструментов, чтобы модель генеративного ИИ могла самостоятельно выбрать нужный. Например, если коэффициенты двух инструментов отличаются на меньшее значение, то пользователь получает сообщение с необходимостью выбрать один из инструментов. Рекомендуется оставить значение по умолчанию **0,02**;
- **MAX_TOOLS_TO_USE** – максимальное количество инструментов, среди которых модель подбирает нужный. Рекомендуется оставить значение по умолчанию **5**.

Системный запрос к модели генеративного ИИ

При общении с интеллектуальным помощником в чате автоматически формируются запросы к модели генеративного ИИ: системный и пользовательский. Если чат-бот не формирует текст системного запроса, в этом случае в модель передается стандартный. Его параметры задаются в секции **DirectumSAService** в секции **chat_template**:

- **ROLE** – роль, от которой передаются текст для обработки в модель генеративного ИИ. Оставьте значение по умолчанию **system**;
- **CONTENT** – текст для обработки, который передается модели генеративного ИИ. Текст от системной роли указывается в повелительном наклонении, а также в нем рекомендуется:

- описывать правила, которым нужно следовать при формировании ответа. Например, писать на русском языке, проверять орфографию или соблюдать определенную структуру;
- указывать роль, от которой нужно формировать текст, например от делопроизводителя или менеджера продаж. Это позволяет формировать ответы корректнее.

Подключение к корпоративным сервисам

По умолчанию интеллектуальный помощник может работать с корпоративными системами или сервисами, которые работают по протоколу MCP. Directum RX работает по протоколу HTTP или HTTPS. Для взаимодействия интеллектуального помощника с Directum RX дополнительно выполняется преобразование HTTP-запросов в формат MCP, поэтому настройки подключения отличаются. Они задаются в секции **DirectumSAService** в секции **MCP_SERVERS** в секции **<Название системы или сервиса, например RX>** и зависят от используемого сервиса или системы:

- подключение к Directum RX:
  - **TYPE** – тип сервиса, к которому подключается помощник. Укажите значение weird;
  - **URL** – адрес для подключения к сервису интеграции в формате https://<Наименование сервера>:<Протокол>/<Имя сервиса интеграции>/odata/IntegrationAIAgent/HandleMCPRequests. Если для сервиса интеграции используется имя по умолчанию **Integration**, в поле укажите значение **https://<Наименование сервера>:<Протокол>/Integration/odata/IntegrationAIAgent/HandleMCPRequests**. Подробнее см. в документации Directum RX в руководстве администратора, раздел «Минимальные настройки системы»;
  - **AUDIENCE** – имя ресурса в сервисе идентификации. Указывается имя, заданное при регистрации ресурса. Подробнее см. в разделе «Создание учетных записей и регистрация ресурсов в сервисе идентификации»;
  - **REFRESH_INTERVAL** – частота обновления данных из Directum RX в сервисе в секундах.;
  - **LOGIN** и **PASSWORD** – логин и пароль системной учетной записи Directum RX для получения системных данных, например для получения списка инструментов;
- подключение к сервису или системе, которые работают по протоколу MCP:
  - **TYPE** – тип сервиса, к которому подключается помощник. Укажите значение **StreamableHttp**;
  - **URL** – адрес для подключения к корпоративному сервису или системе.

  Если в такой системе требуется авторизация, добавьте секцию **headers**. В ней нужно добавить заголовок запроса, в котором передается токен доступа к системе. Пример заполнения секции:

```
"headers": {
 " Authorization ": "Bearer srfgdsfsgtgs"
},
```

Подключение к сервису идентификации

Параметры задаются в секции **DirectumSAService** в секции **IDENTITY_SERVICE**:

- **URL** – адрес для подключения к сервису идентификации;
- **LOGIN** и **PASSWORD** – логин и пароль системной учетной записи сервиса идентификации;
- **AUDIENCE** – имя ресурса сервиса супераппа Directum Omni. Указывается имя, заданное в конфигурационном файле saservice.env в параметре **Authentification_Audience**. Значение по умолчанию **Directum Omni**.
- **TOKEN_TYPE** – тип передаваемого токена. Укажите значение по умолчанию **jwt**.

Запуск сервиса Directum Smart Agent в контейнере от имени пользователя Linux

Реквизиты пользователя, от которого запускается сервис внутри контейнера задаются в секции **variables**:

- **DSA_USER_ID** – идентификатор пользователя. По умолчанию используется UID пользователя, под которым была запущена команда для генерации настроек в config.yml.
- **DSA_USER_NAME** – имя пользователя. Значение по умолчанию **admin**.

Запись трейсов с помощью Elastic APM Service

Если нужно записывать трейсы о работе Directum Smart Agent, установите сервис Elastic APM Service. Настройки для работы с ним задаются в секции **DirectumSAService** в секции **TRACING**:

- **SERVER_URL** – адрес сервиса Elastic APM Service;
- **METRICS_INTERVAL** – интервал сбора метрик о состоянии памяти и процессора. Например, при значении параметра **3s** метрики собираются каждые 3 секунды. Значение по умолчанию **0s**, при котором метрики не собираются;
- **SPAN_FRAMES_MIN_DURATION** – минимальная длительность обработки события в трейсе, которое записывается в него. Если время обработки события превышает значение параметра, в трейс передается не только факт события, но и подробная информация о нем, например, при значении параметра **1s** в трейс записывается подробная информация о событиях длительностью более 1 секунды. Значение по умолчанию **0ms**, при котором подробная информации не записывается в трейс.

Установка основных сервисов

Перед началом установки убедитесь, что установлены компоненты Directum RX и продуктов на ее основе, с которыми планируется работа в Directum Omni. Подготовьте данные для подключения:

- к сервису Public API;
- к брокеру сообщений RabbitMQ, который используется платформой Directum RX.

Установка и настройка выполняется поэтапно в следующем порядке:

1. Сервис мессенджера (MatrixHomeServer).

2. Сервис отправки push-уведомлений (MatrixPushGateway).
3. BFF-сервер (SuperAppBFF).
4. Сервис супераппа (SuperAppService).
5. API-шлюз (ApiGateway).
6. Чат-бот (SuperAppChatBot).
7. Прокси-сервер HAProxy.

В дистрибутив Directum Omni включены шаблоны:

- файл docker-compose.yaml для разворачивания контейнеров с сервисами;
- файл homeserver.yaml с настройками сервиса мессенджера;
- файлы в формате ENV с настройками остальных сервисов.

В этих файлах уже заполнена часть настроек. Измените значения тех параметров, которые указаны в соответствующих разделах инструкции. Для остальных параметров оставьте значения по умолчанию.

Строки подключения

Для строк подключения указаны шаблоны, в которых нужно заполнить данные:

- **Host** и **Port** – адрес хоста и порт компонента;
- **User ID** и **Password** – данные сервисной учетной записи для подключения к сервису идентификации, базе данных или Directum RX;
- **Database** – название базы данных.

Не изменяйте значения параметров **Name** и **UseSsl**.

Пример строки подключения к сервису идентификации:

```
ConnectionStrings__IdentityService:
'Name=Directum.Core.IdentityService;Host=id.contoso.ru;UseSsl=true;Port=443;User
ID=ExampleUser;Password=ExamplePassword;'
```

Установка в закрытый контур

Для установки сервисов в закрытом контуре используйте образы, загруженные при подготовке к установке. В этом случае в файле docker-compose.yml в параметре image указывается путь до образа:

```
services:
  <Имя сервиса>:
    container_name: <Имя контейнера>
    image: <Имя образа>: <Тег>
```

Список загруженных образов и их тегов можно посмотреть с помощью команды:

```
docker image
```

Сервис мессенджера (MatrixHomeServer)

Для работы сервиса требуется настроить его основной узел, а также служебные узлы, если это необходимо в соответствии с типовыми требованиями. Подробнее см. в документе «Directum RX 25.2. Типовые требования к аппаратному и программному обеспечению», входит в комплект поставки.

После настройки запустите сервис и проверьте его работоспособность.

Полный список настроек см. в документации Synapse, статья «Configuration Manual».

Настройка основного узла

1. В СУБД PostgreSQL создайте служебную учетную запись. Для этого выполните скрипт:

   ```
   CREATE USER <Имя учетной записи> WITH PASSWORD '<Пароль>'
   ```

   ВАЖНО. Используйте уникальное имя учетной записи и сильный пароль, содержащий буквы в верхнем и нижнем регистре, цифры и специальные символы.

2. Создайте базу данных для сервиса:

   ```
   CREATE DATABASE <Имя базы данных>
       OWNER <Имя созданной учетной записи>
       ENCODING 'UTF8'
       LOCALE 'C'
       TEMPLATE template0;
   ```

3. Выдайте права на базу данных созданной учетной записи:

   ```
   GRANT ALL PRIVILEGES ON DATABASE <Имя базы данных> TO <Имя созданной учетной записи>
   ```

4. В конфигурационном файле docker-compose.yaml заполните секцию **superapp-synapse-main**:

   ```
   superapp-synapse-main:
     container_name: superapp-synapse-main
     environment:
       - ROLE=MASTER
     image: <Имя и версия образа из шаблона конфигурационного файла>
     volumes:
       - /opt/DirectumOmni/synapse/synapse-data:/data
     networks:
      - sapp-network
   ```

5. Выберите имя сервера, которое будет использовать сервис мессенджера. Оно будет использоваться для взаимодействия между его узлами, а также в именах учетных записей пользователей в сервисе.

   Формат имени учетной записи: @<Имя пользователя>:<Имя сервера>.

   ВАЖНО. После установки сервиса изменить имя сервера нельзя.

6. Выполните команду:

   ```
   sudo docker run --rm \
       -v /opt/DirectumOmni/synapse/synapse-data:/data \
       -e SYNAPSE_SERVER_NAME=<Имя сервера> \
       -e SYNAPSE_REPORT_STATS=yes \
       registry.directum.ru/rnd/synapse:v1.123.0 \
       generate
   ```

В результате создаются:

- конфигурационный файл сервиса homeserver.yaml. Для его заполнения рекомендуется использовать одноименный шаблон из дистрибутива Directum Omni;
- конфигурационный файл с настройками логирования <Имя сервера>.log.config;
- сертификат <Имя сервера>.signing.key, который используется для подписания токенов сервиса.

7. В сгенерированном файле homeserver.yaml укажите настройки в секции **listeners** по аналогии с шаблоном из дистрибутива:

```
listeners:
    # Настройки взаимодействия с клиентским приложением.
  - port: 8008
    tls: false
    type: http
    bind_addresses: ['0.0.0.0']
    x_forwarded: true
    resources:
      - names: [client]
        compress: false
```

Если в соответствии с типовыми требованиями необходимо установить служебные узлы сервиса, также добавьте настройки для взаимодействия с ними:

```
    # Настройки взаимодействия со служебными узлами.
  - port: 9093
    bind_address: '0.0.0.0'
    type: http
    resources:
      - names: [replication]
```

8. Скопируйте из шаблона секцию **instance_map** с информацией об основном узле:

```
instance_map:
    main:
        host: superapp-synapse-main
        port: 9093
```

9. Скопируйте из шаблона секцию **redis** и укажите в ней настройки подключения к СУБД Redis:

- **enabled** – укажите **true**;
- **host** – укажите адрес хоста для подключения к СУБД Redis, подготовленный до начала установки.

Пример настройки:

```
redis:
    enabled: true
    host: synapse_redis
```

10. Скопируйте из шаблона секцию **database** и укажите в ней настройки подключения к базе данных:

- **name** – название используемой СУБД. Укажите значение **psycopg2**;
- **user** – имя служебного пользователя для подключения к СУБД;
- **password** – пароль служебного пользователя;
- **dbname** – название БД, созданной при подготовке к установке;
- **host** – хост для подключения к СУБД.

11. Скопируйте из шаблона секцию **jwt_config** и укажите в ней открытый ключ сертификата для подписания JWT-токенов:

```
jwt_config:
    enabled: true
    secret: |-
      -----BEGIN CERTIFICATE-----
      <Открытый ключ сертификата сервиса идентификации для подписания JWT-токенов>
      -----END CERTIFICATE-----
    algorithm: "RS256"
    subject_claim: http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name
    audiences:
        - "Directum Omni"
```

Открытый ключ можно скопировать из файла CRT, подготовленного до начала установки, в текстовом редакторе.

12. Добавьте параметр **ip_range_whitelist** и укажите в нем адрес Docker-сети, в которой находятся контейнеры с сервисами. Пример настройки:

```
ip_range_whitelist:
  - 172.26.1.0/24 #диапазон IP-адресов для Docker-сети
```

13. Задайте настройки предпросмотра документов в мессенджере. Подробнее см. в документации Synapse, статья «Configuration Manual», раздел **Media Store**.

Настройка служебных узлов

1. В конфигурационном файле docker-compose.yaml заполните секции **superapp-synapse-worker-1**, **superapp-synapse-worker-2** и т.д. Количество секций соответствует количеству служебных узлов сервиса. Подробнее см. в документе «Directum RX 25.2. Типовые требования к аппаратному и программному обеспечению» из комплекта поставки Directum RX, раздел «Сервисы Directum Omni».

   Образец заполнения секции для первого узла:

```
  synapse-w1:
    container_name: synapse-w1
    image: <Имя и версия образа из шаблона конфигурационного файла>
    volumes:
      - ./config/synapse:/data
    # Команда запуска служебного узла
    command: run -m synapse.app.generic_worker --config-path=/data/homeserver.yaml --
config-path=/data/generic_worker_1.yaml
    networks:
      - sapp-network
```

2. Создайте конфигурационный файл generic_worker_1.yml и задайте в нем настройки:

```
worker_app: synapse.app.generic_worker
worker_name: generic_worker_1

worker_listeners:
  - type: http
    port: 8008
    x_forwarded: true
    resources:
      - names: [client]
```

3. Аналогичным образом задайте настройки для остальных служебных узлов. Для каждого узла необходимо задать уникальное имя сервиса, контейнера и конфигурационного файла. Например, для следующего узла можно использовать имя сервиса и контейнера **synapse-w2** и конфигурационный файл generic_worker_2.yaml.

Запуск и проверка работоспособности сервиса

1. Разверните Docker-контейнеры с сервисами. Для этого из папки с файлом docker-compose.yml поочередно выполните команды для всех настроенных контейнеров:

   ```
   sudo docker compose –f docker-compose.yml up –d superapp-synapse-main
   sudo docker compose –f docker-compose.yml up –d synapse-w1
   <…>
   ```

2. В браузере перейдите на страницу: http://localhost:8008/_matrix/client/versions.

   Если сервис работает нормально, то запрос возвращает информацию с версиями API сервиса в формате JSON.

3. Создайте учетную запись администратора. Для этого перейдите в контейнер с основным узлом сервиса с помощью команды:

   ```
   docker exec -it superapp-synapse-main bash
   ```

   Затем создайте учетную запись с помощью команды:

   ```
   register_new_matrix_user -c /data/homeserver.yaml --user <Имя учетной записи> --password <Пароль> --admin
   ```

   ВАЖНО. Укажите такое же имя, что у учетной записи сервиса мессенджера в сервисе идентификации.

   Подробнее см. в документации Synapse, статья «Configuration Manual», раздел «Registering a user».

Сервис отправки push-уведомлений (MatrixPushGateway)

1. В конфигурационном файле docker-compose.yaml заполните секцию **matrixpushgateway**:

   ```
     matrixpushgateway:
       container_name: matrixpushgateway
       image: <Имя и версия образа из шаблона конфигурационного файла>
       entrypoint: sh -c 'cp /certificates/*.crt /etc/pki/ca-trust/source/anchors/ &&
   update-ca-trust &&  dotnet Directum.SuperApp.MatrixPushGateway.dll'
       env_file:
         - samatrixpushgateway.env
       volumes:
         - /opt/certificates:/certificates:ro
       ports:
         - "81:80"
   ```

2. В конфигурационном файле samatrixpushgateway.env задайте строки подключения:
   - **ConnectionStrings__IdentityService** – к сервису идентификации;
   - **ConnectionStrings__MessageBroker** – к сервису сообщений.

3. Разверните Docker-контейнер с сервисом. Для этого из папки с файлом docker-compose.yml выполните команду:

   ```
   sudo docker compose –f docker-compose.yml up –d matrixpushgateway
   ```

BFF-сервер (SuperAppBFF)

1. В конфигурационном файле docker-compose.yaml заполните секцию **saserver**:

```
  saserver:
    container_name: saserver
    image: <Имя и версия образа из шаблона конфигурационного файла>
    entrypoint: sh -c 'cp /certificates/*.crt /etc/pki/ca-trust/source/anchors/ &&
update-ca-trust && dotnet Directum.SuperApp.Server.dll'
    env_file:
      - saserver.env
    volumes:
      - /opt/certificates:/certificates:ro
      - /opt/logs:/app/Logs
```

2. В конфигурационном файле saserver.env укажите строки подключения:
   - **ConnectionStrings__IdentityService** – к сервису идентификации;
   - **ConnectionStrings__MatrixPushGateway** – к сервису отправки push-уведомлений. В параметре **Host** укажите адрес Docker-сети, **Port** – **81**;
   - **ConnectionStrings__Synapse** – к сервису сообщений;
   - **Services__SuperApp-Service** – к сервису супераппа.
3. Задайте настройки для работы с зашифрованным трафиком:
   - **ASPNETCORE_URLS** – список URL-адресов в запросах, которые будет прослушивать сервер. В шаблоне конфигурационного файла задано значение **https://+:443**, то есть прослушивается порт 443 на всех адресах;
   - **ASPNETCORE_Kestrel__Certificates__Default__Path** – путь до SSL-сертификата для приложения Directum Omni внутри контейнера. Пример: /certificates/omni-ssl.pfx;
   - **ASPNETCORE_Kestrel__Certificates__Default__Password** – пароль от сертификата.
4. Заполните адрес API-шлюза в параметрах:
   - **WebPartsProxy__Clusters__DirectumRXWebServer__Destinations__DirectumRX_WebClientServer__Address**

     Значение параметра формируется по шаблону: <Имя сервиса в Docker-сети>/static

     Пример настройки:

```
WebPartsProxy__Clusters__DirectumRXWebServer__Destinations__DirectumRX_WebClientServer_
_Address: 'http://apigateway/static'
```

   - **WebPartsProxy__Clusters__DirectumRXStaticWebServer__Destinations__DirectumRX_WebClientServer__Address**

     Значение параметра формируется по шаблону: <Имя сервиса в Docker-сети >/api/publicapi

     Пример настройки:

```
WebPartsProxy__Clusters__DirectumRXStaticWebServer__Destinations__DirectumRX_WebClientS
erver__Address: 'http://apigateway/api/publicapi'
```

5. Задайте настройки аутентификации:
   - **Authentication__ReturnUrl** – адрес для перенаправления после успешной аутентификации. Укажите адрес для доступа к приложению Directum Omni;
   - **Authentication__TrustedIssuers__0__Issuer** – имя издателя сертификата. По умолчанию в настройках сервиса идентификации задано значение OmniId;

- **Authentication__TrustedIssuers__0__SigningCertificatePath** – путь до открытого ключа сертификата внутри контейнера. Пример: /opt/certificates/ids-jwt.crt.

ВАЖНО. Имя издателя должно быть таким же, как в настройках сервиса идентификации.

6. Укажите адреса внешних хостов, которые должны быть доступны из супераппа. Например, провайдеры внешней аутентификации. Каждый адрес указывается в отдельном параметре **AllowedExternalHosts__<Порядковый номер>**. Пример:

```
AllowedExternalHosts__0: "https://rx.contoso.ru"
AllowedExternalHosts__1: "https://portal.contoso.ru"
AllowedExternalHosts__2: "https://lk.contoso.ru"
```

7. Разверните Docker-контейнеры с сервисами. Для этого из папки с файлом docker-compose.yml выполните команду:

```
sudo docker compose –f docker-compose.yml up -d saserver
```

Сервис супераппа (SuperAppService)

1. В конфигурационном файле docker-compose.yaml заполните секцию **saservice**:

```
  saservice:
    container_name: saservice
    image: <Имя и версия образа из шаблона конфигурационного файла>
    entrypoint: sh -c 'cp /certificates/*.crt /etc/pki/ca-trust/source/anchors/ &&
update-ca-trust && dotnet Directum.SuperApp.Service.dll'
    env_file:
      - saservice.env
    volumes:
      - /opt/certificates/ids:/certificates:ro
      - /opt/ids/logs:/app/Logs
```

2. В конфигурационном файле saservice.env задайте строки подключения:
   - **ConnectionStrings__Synapse** – к сервису мессенджера;
   - **ConnectionStrings__IdentityService** – к сервису идентификации;
   - **ConnectionStrings__SynapseDatabase** – к базе данных сервиса мессенджера;
   - **ConnectionStrings__RX1** – к Directum RX;
   - **ConnectionStrings__ChatBot** – к чат-боту.
3. Задайте настройки аутентификации:
   - **Authentication__TrustedIssuers__0__Issuer** – имя издателя сертификата. По умолчанию в настройках сервиса идентификации задано значение **OmniId**;
   - **Authentication__TrustedIssuers__0__SigningCertificatePath** – путь до открытого ключа сертификата внутри контейнера. Укажите значение в соответствии с созданной конфигурацией Docker. Пример: /opt/certificates/ids-jwt.crt.

ВАЖНО. Имя издателя должно быть таким же, как в настройках сервиса идентификации.

4. Если планируется использование плагина Контур.Толк для проведения видеоконференций, добавьте параметры:
   - **Plugins__0__Enabled** – признак использования плагина. Укажите значение **true**. Если параметр не задан, плагин не используется;
   - **Plugins__0__Args__talkUrl** – адрес для подключения к сервису Контур.Толк. Если параметр не задан, по умолчанию используется значение **https://ktalk.ru/**.

5. Укажите секреты, которые используются для взаимодействия сервисов:
   - **General__SynapseRegistrationSharedSecret** – секрет, который позволяет сервису супераппа создавать новых пользователей мессенджера. Укажите то же значение, что задано в автоматически сгенерированном конфигурационном файле сервиса мессенджера homeserver.yaml в параметре **registration_shared_secret**;
   - **General__ChatBotToken** – секрет для взаимодействия с сервисом-чат-бота. Это же значение нужно указать в настройках чат-бота.

     СОВЕТ. Сгенерировать секрет можно, например, с помощью библиотеки OpenSSL. Для этого выполните команду:

   ```
   openssl rand -base64 32
   ```
6. По умолчанию в конфигурационном файле в параметре **Applets** задан массив с настройками миниаппов HR Pro, Directum Portal и «Входящие». В каждом элементе массива задайте значения параметров, содержащих URL-адреса ресурсов миниаппов:
   - **IconUrl** – URL-адрес иконки миниаппа;
   - **resourceUrl** – URL-адрес продукта, для которого добавляется миниапп;
   - **hyperlinkServerUrl** – URL-адрес сервера Directum RX (для миниаппа «Входящие»).

   Укажите значения параметров на основе примеров ниже, при этом замените в них доменные имена на используемые в вашей компании. Менять значение остальных параметров не нужно.

   Если миниапп не используется, в параметре **Enabled** для него укажите значение **false**.

   Пример для миниаппа HR Pro

   Предположим, что в инфраструктуре компании сайт личного кабинета развернут по адресу **lk.contoso.ru**. В этом случае задайте настройки следующим образом:

   ```
   Applets__0__IconUrl: "https://lk.contoso.ru/logo.svg"
   Applets__0__Config__resourceUrl: https://lk.contoso.ru/authorize?
   returnUrl=https://lk.contoso.ru{route}
   ```

   Пример для миниаппа Directum Portal

   Предположим, что в инфраструктуре компании сайт Directum Portal развернут по адресу **portal.contoso.ru**. В этом случае задайте настройки следующим образом:

   ```
   Applets__1__IconUrl: "https://portal.contoso.ru/o/sungero-theme/images/favicon.ico"
   Applets__1__Config__resourceUrl: "https://portal.contoso.ru/c/portal/login"
   ```

   Пример для миниаппа «Входящие»

   Предположим, что в инфраструктуре компании Directum RX развернут по адресу **rx.contoso.ru**. В этом случае задайте настройки следующим образом:

   ```
   Applets__2__Config__hyperlinkServerUrl: "http://rx.contoso.ru/Sungero"
   ```
7. Разверните Docker-контейнеры с сервисами. Для этого из папки с файлом docker-compose.yml выполните команду:

   ```
   sudo docker compose -f docker-compose.yml up -d saservice
   ```

API-шлюз (ApiGateway)

1. В конфигурационном файле docker-compose.yaml заполните секцию **apigateway**:

   ```
   apigateway:
     container_name: apigateway
     image: <Имя и версия образа из шаблона конфигурационного файла>
   ```

```
    entrypoint: "sh -c 'cp /certificates/*.crt /etc/pki/ca-trust/source/anchors/ &&
update-ca-trust extract && dotnet Directum.Core.ApiGateway.dll'"
    env_file:
      - apigateway.env
    volumes:
      - /opt/certificates/ids:/certificates:ro
      - /opt/logs:/app/Logs
```

2. В конфигурационном файле apigateway.env в параметре **ConnectionStrings__IdentityService** задайте строку подключения к сервису идентификации.
3. Задайте настройки сертификата, которым сервис идентификации подписывает токены для аутентификации запросов к сервисам:
   - **Authentication__TrustedIssuers__0__Issuer** – имя издателя сертификата. По умолчанию в настройках сервиса идентификации задано значение **OmniId**;
   - **Authentication__TrustedIssuers__0__SigningCertificatePath** – путь до открытого ключа сертификата. Укажите значение в соответствии с созданной конфигурацией Docker. Пример: /opt/certificates/ids-jwt.crt.

   ВАЖНО. Имя издателя должно быть таким же, как в настройках сервиса идентификации.
4. Задайте адреса для перенаправления запросов к ресурсам. Каждому ресурсу соответствует группа параметров с префиксом **RoutingConfiguration__Routes__<Номер блока>__**. Название ресурса указывается в параметре **Key**:
   - сайт личного кабинета – **EmployeeSelfServiceOffice**;
   - предпросмотр в Directum RX – **RxPreviewApi**;
   - сервис PublicAPI Directum RX – **PublicApi**;
   - предпросмотр в папке Directum RX «Входящие» – **InboxPreview**;
   - папка Directum RX «Входящие» – **inbox**;
   - предпросмотр в модуле Directum RX «Корпоративные услуги» – **ESMPreviewApi**;
   - модуль Directum RX «Корпоративные услуги» – **ESM**;
   - сервис супераппа – **Directum.SuperApp**;
   - статичные ресурсы Directum RX – **DirectumRXStatic**.

   В группе каждого сервиса укажите значения параметров:
   - **DownstreamHostAndPorts__0__Host** – хост для подключения к сервису;
   - **DownstreamHostAndPorts__0__Port** – порт для подключения к сервису;
   - **DownstreamPathTemplate** – шаблон конечной точки, к которой перенаправляются запросы. В блоках для ресурсов Directum RX она формируется следующим образом:

     /<Относительный адрес веб-клиента Directum RX>/<конечная точка>

     По умолчанию используется относительный адрес **client**, т.е. веб-клиент доступен по адресу, например, https://rx.contoso.ru/client/. Если в компании используется другой относительный адрес, замените его в значениях параметров.

     В блоках для ресурсов, не относящихся к Directum RX, например для сервиса супераппа, менять значение параметра не нужно.

Пример. Настройки перенаправления для сервиса PublicAPI, если относительный адрес веб-клиента **WebClient**:

```
RoutingConfiguration__Routes__1__Key: "PublicApi"
RoutingConfiguration__Routes__1__DownstreamPathTemplate: "/WebClient/api/public/v2/
{endpoint}/{everything}"
RoutingConfiguration__Routes__1__DownstreamHostAndPorts__0__Host: "rx.contoso.ru"
RoutingConfiguration__Routes__1__DownstreamHostAndPorts__0__Port: "443"
```

В этом случае API-шлюз будет перенаправлять запросы к сервису PublicAPI на адрес https://rx.contoso.ru/WebClient/api/public/v2/<конечная точка>.

5. Разверните Docker-контейнер с сервисом. Для этого из папки с файлом docker-compose.yml выполните команду:

   ```
   sudo docker compose –f docker-compose.yml up -d apigateway
   ```

6. Проверьте работоспособность сервиса. Для этого выполните команду:

   ```
   docker exec -it apigateway curl http://localhost/ready
   ```

   Запрос возвращает информацию о состоянии сервиса в формате JSON. Убедитесь, что в каждой секции значение параметра **Status** равно **Healthy**.

Чат-бот (SuperAppChatBot)

1. Создайте учетную запись для чат-бота в сервисе мессенджера. Для этого перейдите в контейнер с его основным узлом с помощью команды:

   ```
   docker exec --it superapp-synapse-main
   ```

   Затем создайте учетную запись с помощью команды:

   ```
   register_new_matrix_user -c /data/homeserver.yaml --user <Имя учетной
   записи> --password <Пароль> --no-admin
   ```

   ВАЖНО. Укажите такое же имя (**OmniBot**), что у учетной записи чат-бота в сервисе идентификации.

   Подробнее см. в документации Synapse, статья «Configuration Manual», раздел «Registering a user».

2. В конфигурационном файле docker-compose.yaml заполните секцию **superapp-chatbot**:

   ```
     superapp-chatbot:
       container_name: superapp-chatbot
       image: <Имя и версия образа из шаблона конфигурационного файла>
         entrypoint: sh -c 'cp /certificates/*.crt /etc/pki/ca-trust/source/anchors/ &&
   update-ca-trust && dotnet Directum.SuperApp.Chatbot.dll'
       env_file:
         - superappchatbot.env
       volumes:
         - /opt/certificates:/certificates:ro
         - /opt/logs:/app/Logs
   ```

3. В конфигурационном файле superappchatbot.env в параметре **GeneralSettings__ChatBotSecurityToken** укажите секрет для взаимодействия с сервисом супераппа. Он должен совпадать со значением, указанным в настройках сервиса.

4. В параметре **GeneralSettings__SystemName** укажите значение **DirectumRX**.

5. Укажите параметры для подключения к сервису идентификации:

   - **GeneralSettings__IdentityServiceAddress** – URL-адрес для подключения;

- **GeneralSettings__IdentityServiceUser** – имя служебной учетной записи в сервисе идентификации. Укажите значение **OmniBot**;
- **GeneralSettings__IdentityServicePassword** – пароль служебной учетной записи.

6. Укажите параметры для подключения к сервису мессенджера:
   - **MatrixSettings__MatrixUrl** – адрес для подключения внутри сети Docker в формате http://<Имя контейнера с сервисом>:<Порт>. Например, http://superapp-chatbot:1443;
   - **MatrixSettings__MatrixAudience** – укажите значение **Directum Omni**;
   - **MatrixSettings__MatrixUser** – имя учетной записи чат-бота в сервисе мессенджера. Укажите значение **OmniBot**;
   - **MatrixSettings__MatrixPassword** – пароль учетной записи;
   - **MatrixSettings__MatrixHomeserverName** – имя сервера, указанное в настройках сервиса мессенджера.
7. Если планируется использование *помощника Ario*, укажите параметры для подключения к его сервису:
   - **AISettings__AIUrl** – URL-адрес для подключения;
   - **AISettings__AIToken** – токен доступа, который передается в запросах к сервису Directum Smart Agent. Укажите значение, заданное при установке сервиса в конфигурационном файле config.yml в параметре **ACCESS_TOKEN**.
8. В параметре **QueueSettings__0__ConnectionString** укажите строку подключения к RabbitMQ. Для этого в шаблоне строки укажите:
   - **hostName** – IP-адрес сервера;
   - **virtualhost** – имя виртуального хоста RabbitMQ;
   - **port** – порт для подключения;
   - **userName** и **password** – имя и пароль учетной записи;
   - **exchange** – точка обмена, к которой привязываются очереди сообщений веб-сервера. Также используется для наименования очередей сообщений. Укажите произвольное название, например **ChatBot**.

   ВАЖНО. Для всех параметров, кроме **exchange**, укажите те же значения, что были заданы при установке Directum RX. Подробнее см. в руководстве администратора Directum RX, раздел «Минимальные настройки» (Linux, Windows), пункт «Настройка подключения к RabbitMQ».
9. Разверните Docker-контейнеры с сервисами. Для этого из папки с файлом docker-compose.yml выполните команду:

   ```
   sudo docker compose -f docker-compose.yml pull
   ```
10. При необходимости настройте отображаемое имя и иконку чат-бота. Для этого выполните команду:

    ```
    docker exec -it superapp-chatbot sh /app/tools/bot-set-info.sh -n <Имя чат-бота> -f <Путь к файлу с иконкой>
    ```

    где:
    - **-n** – имя чат-бота, которое будет отображаться в мессенджере. Необязательный ключ. Если указана пустая строка, устанавливается значение по умолчанию **Интеллектуальный помощник**;

- **-f** – путь к файлу с иконкой внутри контейнера. Необязательный ключ. Поддерживаются файлы в форматах PNG и JPG. Если указана пустая строка, устанавливается иконка по умолчанию.

  Чтобы добавить файл внутрь контейнера, в конфигурационном файле docker-compose.yml в секции **superapp-chatbot** добавьте в параметр **volumes** строку в формате <Путь до файла вне контейнера>: <Путь до файла внутри контейнера>. Пример настройки:

  ```
  volumes:
    - /opt/certificates:/certificates:ro
    - /opt/logs:/app/Logs
    - /opt/icons/chatbot.png:/app/icons/chatbot.png # Файл с иконкой
  ```

Пример команды:

```
docker exec -it superapp-chatbot sh /app/tools/bot-set-info.sh -n "Ассистент Contoso" -f "/app/icons/chatbot.png"
```

Прокси-сервер HAProxy

Для перенаправления входящих запросов на сервере с основными сервисами Directum Omni необходимо настроить HAProxy в качестве обратного прокси-сервера. Для этого:

1. В конфигурационный файл docker-compose.yaml добавьте параметры для контейнера с HAProxy:

   ```
   haproxy:
     container_name: haproxy
     image: <Имя и версия образа из шаблона конфигурационного файла>
     volumes:
       - /opt/DirectumOmni/haproxy/haproxy.cfg:/usr/local/etc/haproxy/haproxy.cfg:ro
       - /opt/certificates:/certificates:ro
     networks:
       - sapp-network
   ```

2. В папке /opt/DirectumOmni/haproxy создайте конфигурационный файл haproxy.cfg и добавьте в него секции **global**, **defaults** и **resolver**. Рекомендуемый вариант настройки:

   ```
   global
       ssl-default-bind-options ssl-min-ver TLSv1.2

   defaults
       mode http
       timeout connect 5s
       timeout client 2m
       timeout server 15m
       option http-server-close
       option forwardfor except 127.0.0.0/8
       option redispatch
       option httplog
       default-server init-addr last,libc,none
       default-server check inter 10s resolvers docker_resolver
       retries 3
       maxconn 4000
       log stdout len 4096 format raw local0 debug

   resolvers docker_resolver
       nameserver dns 127.0.0.11:53
   ```

3. Добавьте секцию **frontend** с настройками, указанными ниже. Вместо выделенного жирным шрифтом фрагмента укажите название файла SSL-сертификата в формате PEM, подготовленного до начала установки.

   ```
   frontend directum_omni
       bind 0.0.0.0:80
       http-request set-header X-Forwarded-Host %[req.hdr(host)]
   ```

```
    bind 0.0.0.0:443 ssl crt /certificates/<Сертификат в формате PEM>
    redirect scheme https if !{ ssl_fc }
    http-request set-header X-Forwarded-Proto https
    #Правила обработки запросов к виртуальным каталогам
    acl tokens_refresh path_beg /tokens/refresh
    acl api_path path_beg /api/
    acl matrix_path path_beg /_matrix
    # Перенаправление запросов по указанным выше правилам
    use_backend saserver_https if tokens_refresh
    use_backend apigateway if api_path
    use_backend matrix_backend if matrix_path
    default_backend saserver_https
```

4. Вместо секции backend по умолчанию добавьте секции для сервисов Directum Omni:
   - для BFF-сервера:

```
backend saserver_https
    mode http
    balance roundrobin
    server saserver1 saserver:443 ssl verify none
    # Общие настройки заголовков
    http-request set-header Host %[req.hdr(Host)]
    http-request set-header X-Real-IP %[src]
    http-request set-header X-Forwarded-For %[src]
    http-request set-header X-Forwarded-Proto https if { ssl_fc }
    http-request set-header X-Forwarded-Proto http if !{ ssl_fc }
    # Настройки location /
    acl is_olm path_end /olm.wasm
    http-request set-path /olm.wasm if is_olm
    # Настройки времени ожидания для запросов
    timeout server 50000
    timeout connect 5000
```

   - для сервиса сообщений. Ниже приведена конфигурация с 4 служебными узлами. При необходимости увеличьте количество служебных узлов или уберите соответствующие настройки, если служебные узлы не используются:

```
# Основной узел
backend matrix_backend
    mode http
    balance roundrobin
    server synapse1 superapp-synapse-main:8008
    # Общие настройки заголовков
    http-request set-header Host %[req.hdr(Host)]
    http-request set-header X-Real-IP %[src]
    http-request set-header X-Forwarded-For %[src]
    http-request set-header X-Forwarded-Proto https if { ssl_fc }
    http-request set-header X-Forwarded-Proto http if !{ ssl_fc }
    #Настройки времени ожидания для запросов
    timeout server 50000
    timeout connect 5000
    # ACL для синхронных запросов, если используются служебные узлы
    acl sync_request path_reg ^/_matrix/client/(r0|v3)/sync
    use_backend sync_workers if sync_request
# Настройка синхронных запросов, если используются служебные узлы
backend sync_workers
    balance leastconn
    # Долгие таймауты для long-polling sync
    timeout server 90s
    timeout connect 10s
    # 5 равных служебных узлов для синхронных запросов
    server synapse-worker1 synapse-w1:8008 check weight 100 maxconn 1000
    server synapse-worker2 synapse-w2:8008 check weight 100 maxconn 1000
    server synapse-worker3 synapse-w3:8008 check weight 100 maxconn 1000
    server synapse-worker4 synapse-w4:8008 check weight 100 maxconn 1000
```

   - для API-шлюза. Вместо выделенного жирным шрифтом фрагмента укажите адрес для доступа к приложению Directum Omni. Например, https://omni.contoso.ru.

```
backend apigateway
    mode http
    balance roundrobin
    server apigateway1 apigateway:80
    # Настройки подключения
```

```
    fullconn 3000
    timeout server 50000
    timeout connect 5000
    # Удаление заголовка CORS
    http-response del-header Access-Control-Allow-Origin
    # Добавление CORS заголовков
    http-response set-header Access-Control-Allow-Origin "<Адрес для
доступа к Directum Omni>"
    http-response set-header Access-Control-Allow-Credentials "true"
    # Общие настройки заголовков
    http-request set-header Host %[req.hdr(Host)]
    http-request set-header X-Real-IP %[src]
    http-request set-header X-Forwarded-For %[src]
    http-request set-header X-Forwarded-Proto https if { ssl_fc }
    http-request set-header X-Forwarded-Proto http if !{ ssl_fc }
```

5. Сохраните изменения в конфигурационном файле, затем выполните команду:

```
sudo docker compose -f docker-compose.yml pull
```

6. Убедитесь, что все контейнеры с сервисами Directum Omni запущены. Для этого выполните команду:

```
docker compose ps
```

В результате выводится список контейнеров. Проверьте, что в нем есть все контейнеры из файла docker-compose.yaml.

Настройка подключения к сервисам в Directum RX

Настройки задаются в конфигурационном файле Directum RX config.yml. Подробнее см. в руководстве администратора, раздел «Конфигурирование настроек» (Linux, Windows).

1. На сервере с сервисами Directum RX разместите открытый ключ сертификата для подписания JWT-токенов в папке {{ home_path }}/data_protection. Подробнее см. в руководстве администратора Directum RX, раздел «Минимальные настройки» (Linux, Windows).
2. В конфигурационном файле в секцию **Общие настройки** (common_config) добавьте параметры для подключения к Directum Omni:
   - **OMNI_BASE_URL** – адрес для доступа к приложению Directum Omni. Укажите адрес, на который при подготовке к установке получен SSL-сертификат;
   - **OMNI_SERVICE_PATH** – конечная точка для доступа к сервисам Directum Omni. Укажите значение **/api**.

   Пример настройки:

```
common_config: &base
    <...>
    OMNI_BASE_URL: 'https://omni.contoso.ru'
    OMNI_SERVICE_PATH: '/api'
```

3. В секцию **Общие настройки** (common_config) добавьте настройки доверия для сертификатов:

```
common_config: &base
    <...>
    TOKEN_VALID_ISSUERS_LIST:
        string:
        -   "OmniId"
    TOKEN_ADDITIONAL_CERTIFICATES:
        TokenCertificatesSettings:
        -   "@CertificateFile": '{{ home_path }}/data_protection/ids-jwt.crt'
```

ВАЖНО. Убедитесь, что настройки также добавлены в секцию **Public API** (SungeroPublicApi) с помощью псевдонима ***base**:

```
SungeroPublicApi:
    <<: *base
```

4. Добавьте секцию **ids_config** с якорем **&ids** и укажите в ней значения параметров:
   - **IDENTITY_SERVICE_ADDRESS** – внешний адрес сервиса, указанный в его конфигурационном файле в параметре **General__ServiceEndpoint**;
   - **IDENTITY_SERVICE_SYSTEM_CODE** – код системы для сервиса идентификации. Укажите значение **DirectumRX**;
   - **IDENTITY_SERVICE_USER** – имя служебной учетной записи. Укажите данные созданной ранее учетной записи для сервисов Directum RX;
   - **IDENTITY_SERVICE_PASSWORD** – пароль служебного пользователя;
   - **IDENTITY_SERVICE_NAME** – имя контейнера с сервисом идентификации, указанное в файле docker-compose.yml в параметре **container-name**;
   - **IDENTITY_SERVICE_PLATFORM_TOKEN_LIFE_TIME** – время жизни токена сервиса идентификации. Не рекомендуется менять значение параметра;
   - **IDENTITY_SERVICE_TOKEN_CACHE_LIFE_TIME** – время кэширования токена сервиса идентификации. Не рекомендуется менять значение параметра;
   - **IDENTITY_SERVICE_APPLICATION** – приложения, с которыми работает сервис. Для Directum Omni в параметр **application** добавьте набор параметров:

```
-   '@name': 'Directum Omni'
    '@audience': 'Directum Omni'
    '@claimType': 'DirectumRX/Username'
```

Пример настройки:

```
ids_config: &ids
    IDENTITY_SERVICE_ADDRESS: 'https://id.contoso.ru'
    IDENTITY_SERVICE_SYSTEM_CODE: 'DirectumRX'
    IDENTITY_SERVICE_USER: <Имя служебной учетной записи>
    IDENTITY_SERVICE_PASSWORD: <Пароль служебной учетной записи>
    IDENTITY_SERVICE_NAME: 'identity-service'
    IDENTITY_SERVICE_PLATFORM_TOKEN_LIFE_TIME: '00:15:00'
    IDENTITY_SERVICE_TOKEN_CACHE_LIFE_TIME: '00:15:00'
    IDENTITY_SERVICE_APPLICATIONS:
        'application':
        -   '@name': 'Directum Omni'
            '@audience': 'Directum Omni'
            '@claimType': 'DirectumRX/Username'
```

5. Добавьте настройки из секции **ids_config** в секции **Сервис интеграции** (IntegrationService) и **Сервис асинхронных событий** (SungeroWorker) с помощью псевдонима ***ids**:

```
services_config:
    <...>
    SungeroWorker:
        <<: [*base, *ids]
    <...>
    IntegrationService:
        <<: [*base, *ids]
```

6. Чтобы в Directum Omni в заданиях и уведомлениях был доступен предпросмотр вложений, в секцию **Public API** (SungeroPublicApi) добавьте параметр:

```
PREVIEW_STORAGE_API_HOST_URL: '{{ protocol }}://{{ host_fqdn }}/Preview'
```

7. Для применения настроек перезапустите сервисы Directum RX с помощью команды:

```
./do.sh all up
```

Настройка совместной работы с сервисами HR Pro и Directum Portal

В состав HR Pro и Directum Portal входит собственный экземпляр сервиса идентификации. Если в компании используются эти продукты, то чтобы пользователям не нужно было проходить аутентификацию в каждом из них, между сервисами необходимо настроить доверие. Кроме того, для сайта личного кабинета HR Pro необходимо разрешить его встраивание в суперапп Directum Omni.

ПРИМЕЧАНИЕ. В разделе описан порядок настройки для сервисов HR Pro. Описание настроек для Directum Portal см. в документации продукта, раздел «Настройка совместной работы с Directum Omni».

Порядок настройки:

1. Убедитесь, что сервисы Directum Omni и HR Pro подключены к одной и той же инсталляции Directum.
2. Для сервиса идентификации HR Pro укажите Directum Omni в качестве доверенного издателя токенов. Для этого в конфигурационном файле config.yml в секцию **IdentityService** добавьте параметры:
   - **TrustedIssuers__0__Issuer** – имя издателя токенов. Укажите такое же значение, как для сервиса идентификации Directum Omni, по умолчанию это **OmniId**;
   - **TrustedIssuers__0__SigningCertificatePath** – путь до открытого ключа подготовленного сертификата. Укажите путь внутри контейнера с сервисом идентификации. Пример: /opt/certificates/ids-jwt.crt;
   - **TrustedIssuers__0__UserIdentifyingClaim** – название пакета с информацией о пользователе (claim). Укажите значение **DirectumRX/Username**.

   Если для сервиса идентификации HR Pro уже настроены другие доверенные издатели, добавьте в конфигурационный файл те же параметры, но вместо **0** укажите **1**, затем **2** и так далее. Например, **TrustedIssuers__1__Issuer**.

   ПРИМЕЧАНИЕ. Если сервис идентификации HR Pro установлен на сервере с Microsoft Windows, синтаксис настроек см. в руководстве администратора HR Pro (Windows), раздел «Сервис идентификации», пункт «Доверенные издатели токенов».
3. Настройте авторизацию запросов к компонентам HR Pro: сайту личного кабинета, API-шлюзу, сервису сообщений и серверу сеансов, если он установлен. Для этого на серверах с указанными сервисами в конфигурационный файл HR Pro config.yml в секции **Site**, **ApiGateway**, **MessageBroker** и **SessionServer** добавьте параметры:
   - **Authentication__TrustedIssuers__0__Issuer** – имя издателя сертификатов. Укажите то же имя, что задано для сервиса идентификации;
   - **Authentication__TrustedIssuers__0__SigningCertificatePath** – путь до открытого ключа сертификата, подготовленного до начала установки. Укажите путь внутри контейнера с сервисом идентификации. Пример: /opt/certificates/ids-jwt.crt.

Если для сервисов уже заданы другие настройки авторизации, добавьте в конфигурационный файл те же параметры, но вместо **0** укажите **1**, затем **2** и так далее. Например, **Authentication__TrustedIssuers__1__Issuer**.

Пример настройки:

```
Authentication__TrustedIssuers__0__Issuer: "HrProId"
Authentication__TrustedIssuers__0__SigningCertificatePath:      "/opt/certificates/ids-jwt.crt"
Authentication__TrustedIssuers__1__Issuer: "OmniId"
Authentication__TrustedIssuers__1__SigningCertificatePath:      "/opt/certificates/ids-jwt.crt"
```

ПРИМЕЧАНИЕ. Если сервисы установлены на сервере с Microsoft Windows, синтаксис настроек см. в руководстве администратора HR Pro (Windows), раздел «Общие настройки сервисов», пункт «Авторизация запросов».

4. Разрешите встраивание сайта личного кабинета в суперапп Directum Omni. Для этого в секцию **Site** добавьте параметр **Security__AllowedFrameAncestors** и укажите в нем URL-адрес, по которому будет доступен суперапп. Пример настройки:

   ```
   Security__AllowedFrameAncestors: "https://omni.contoso.ru"
   ```

5. С помощью утилиты id повторно зарегистрируйте в сервисе идентификации HR Pro:

   - сервис хранения BLOB-объектов:

   Linux

   ```
   ./do.sh id_cli update_resource --resource_name="Directum.Core.BlobStorageService" --resource_path="<Папка с Directum Launcher>/etc/_builds/Audiences/BlobStorageServiceAudience.json"
   ```

   Windows

   ```
   id update resource "Directum.Core.BlobStorageService" -c "<Путь к дистрибутиву сервиса>\BlobStorageServiceAudience.json"
   ```

   - сервис хранения файлов предпросмотра:

   Linux

   ```
   ./do.sh id_cli update_resource --resource_name="Directum.Core.PreviewStorage" --resource_path="<Папка с Directum Launcher>/etc/_builds/Audiences/PreviewStorageServiceAudience.json"
   ```

   Windows

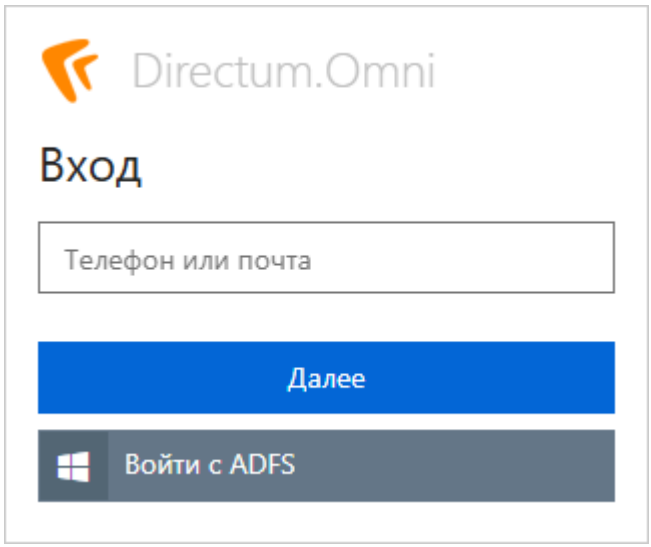
   ```
   id update resource "Directum.Core.PreviewStorage" -c "<Путь к дистрибутиву сервиса>\PreviewStorageServiceAudience.json"
   ```

   ВАЖНО. Файлы в формате JSON для повторной регистрации ресурсов включены в компонент для установки сервисов HR Pro с версии 2.7.252.22.

6. Перезапустите сервисы HR Pro для применения настроек.

Настройка внешней аутентификации

В Directum Omni поддерживается внутренняя аутентификация пользователей по логину и паролю и внешняя – с использованием внешних провайдеров. В зависимости от настроек, заданных администратором, пользователю доступны один или несколько вариантов аутентификации. Например, через Google и AD FS.



ВАЖНО. Настройка внешней аутентификации доступна только для пользователей, которые еще не подключены к Directum Omni. Если сотрудник планирует использовать внешнюю аутентификацию, настройте ее перед тем, как подключать его.

Для использования внешней аутентификации:

1. Ознакомьтесь с доступными типами аутентификации и определите, какие из них будут использоваться. В базовой поставке поддерживаются:
   - Google;
   - Active Directory Federation Services (AD FS);
   - Keycloak.

   Если используется внешняя аутентификация, то для входа в Directum Omni сервис идентификации перенаправляет пользователя на страницу используемого провайдера. После этого реквизиты пользователя сопоставляются с учетной записью в сервисе идентификации.
2. Настройте выбранные типы аутентификации.
3. Настройте интеграцию с нужными провайдерами:
   - AD FS по протоколу WS-Federation
   - AD FS по протоколу SAML
   - Google
   - Keycloak
4. Создайте учетные записи с заданной аутентификацией.
5. Если необходимо использовать двухфакторную внешнюю аутентификацию, настройте ее на стороне провайдера. Порядок настройки см. в документации выбранных провайдеров.

Настройка типов аутентификации

В конфигурационных файлах укажите тип аутентификации по умолчанию и настройте доступ к выбранным провайдерам аутентификации из мобильного приложения.

Тип аутентификации по умолчанию

В конфигурационном файле identity-service.env задайте тип аутентификации по умолчанию на случай, если при регистрации пользователя аутентификация не задана. В параметре **UserAccounts__DefaultAuthentication** укажите возможные значения:

- **Password** – внутренняя парольная аутентификация;
- **Adfs** – аутентификация в AD FS по протоколу WS-Federation;
- **SAML** – аутентификация в AD FS по протоколу SAML 2.0;
- **OpenId** – аутентификация в Keycloak по протоколу OpenID Connect 1.0;
- **Google** – аутентификация в Google.

Пример настройки см. в разделе «AD FS по протоколу WS-Federation».

Доступ к провайдерам из мобильного приложения

Чтобы сотрудники могли работать с мобильным приложением Directum Omni, настройте для него доступ к провайдерам аутентификации. Для этого в конфигурационный файл BFF-сервера saserver.env добавьте параметр **AllowedExternalHosts__<Порядковый номер провайдера>** и укажите в нем URL-адрес хоста провайдера.

Пример настройки:

```
AllowedExternalHosts__0: "https://adfs.contoso.ru/adfs/ls"
AllowedExternalHosts__1: "https://google.ru"
```

AD FS по протоколу WS-Federation

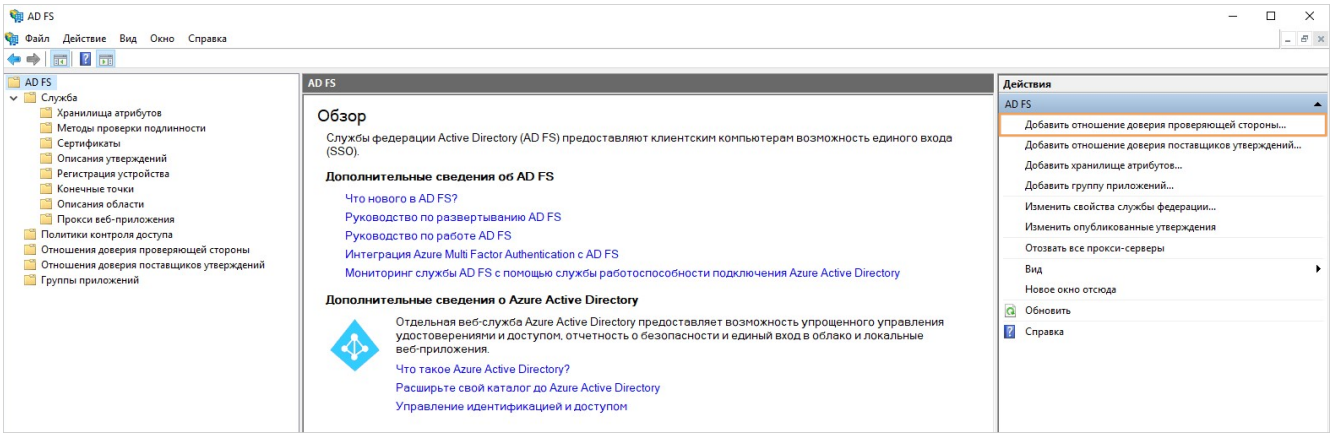
1. Для аутентификации в AD FS используется протокол WS-Federation. Перед началом настройки убедитесь, что он доступен в службе.
2. Настройте сервис AD FS.
3. Настройте плагин провайдера аутентификации в сервисе идентификации.
4. Проверьте корректность настройки.

В разделе приведен пример настройки при работе с Windows Server 2019.

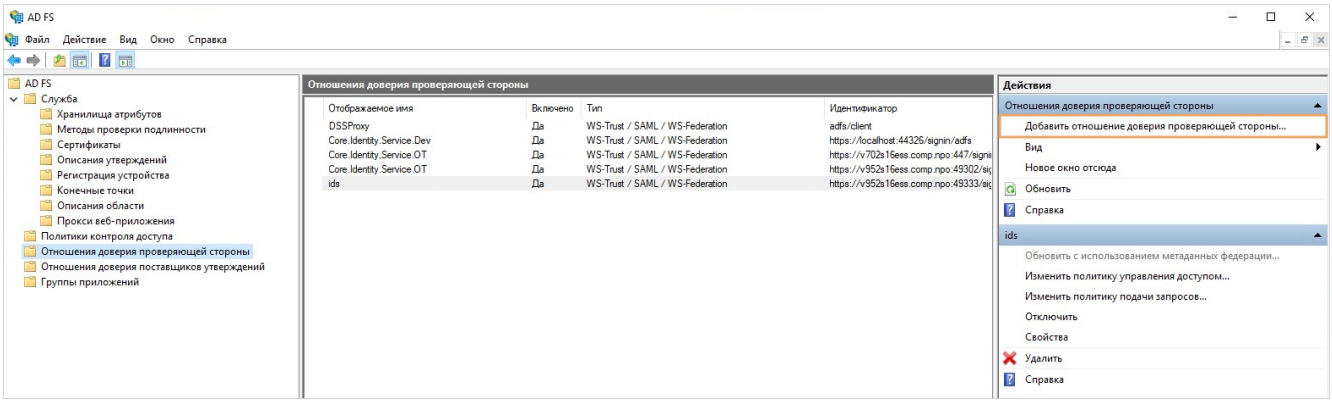
Настройка AD FS

1. Перед началом настройки сохраните в удобной форме адрес сервиса идентификации. Пример: **https://id.contoso.ru**.
2. В диспетчере серверов в выпадающем списке **Средства** выберите пункт **Управление AD FS**. Откроется окно утилиты.

3. Добавите отношение доверия проверяющей стороны одним из способов:
   - перейдите в узел **AD FS** и в разделе «Действия» выберите пункт **Добавить отношение доверия проверяющей стороны...**:

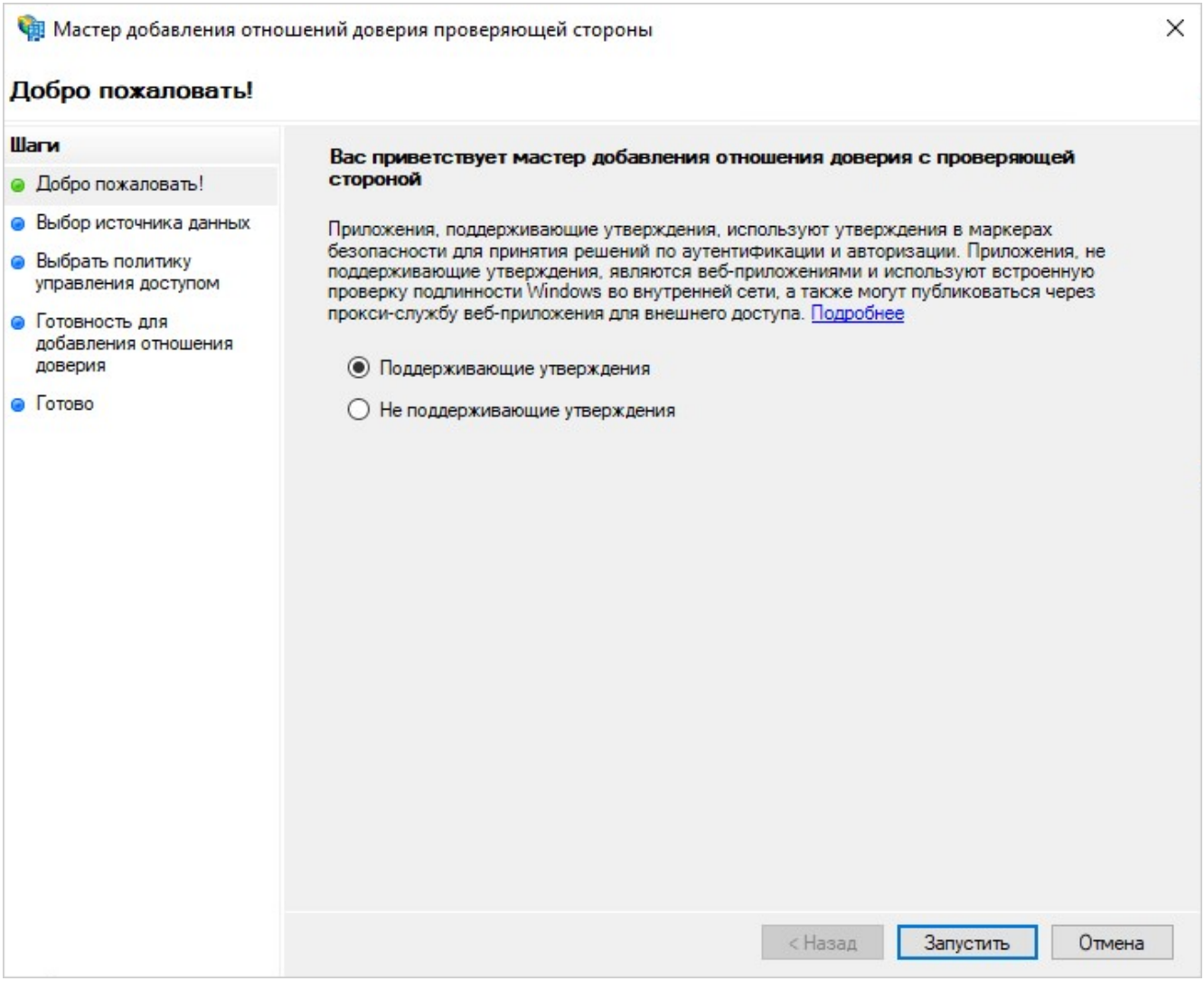


   - перейдите в узел **Отношения доверия проверяющей стороны** и в разделе «Действия»выберите пункт **Добавить отношение доверия проверяющей стороны..**:

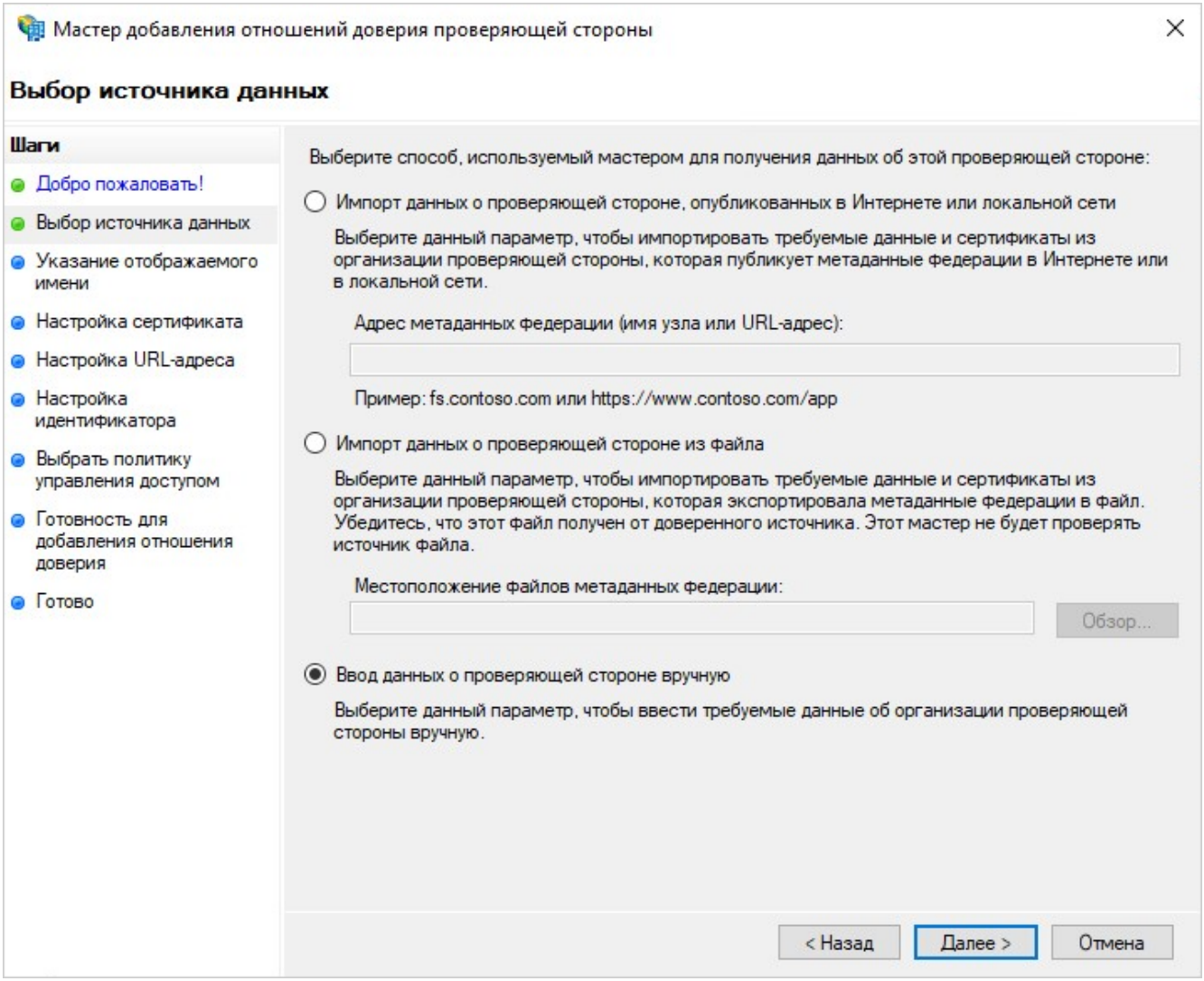


Откроется окно «Мастер добавления отношений проверяющей стороны».

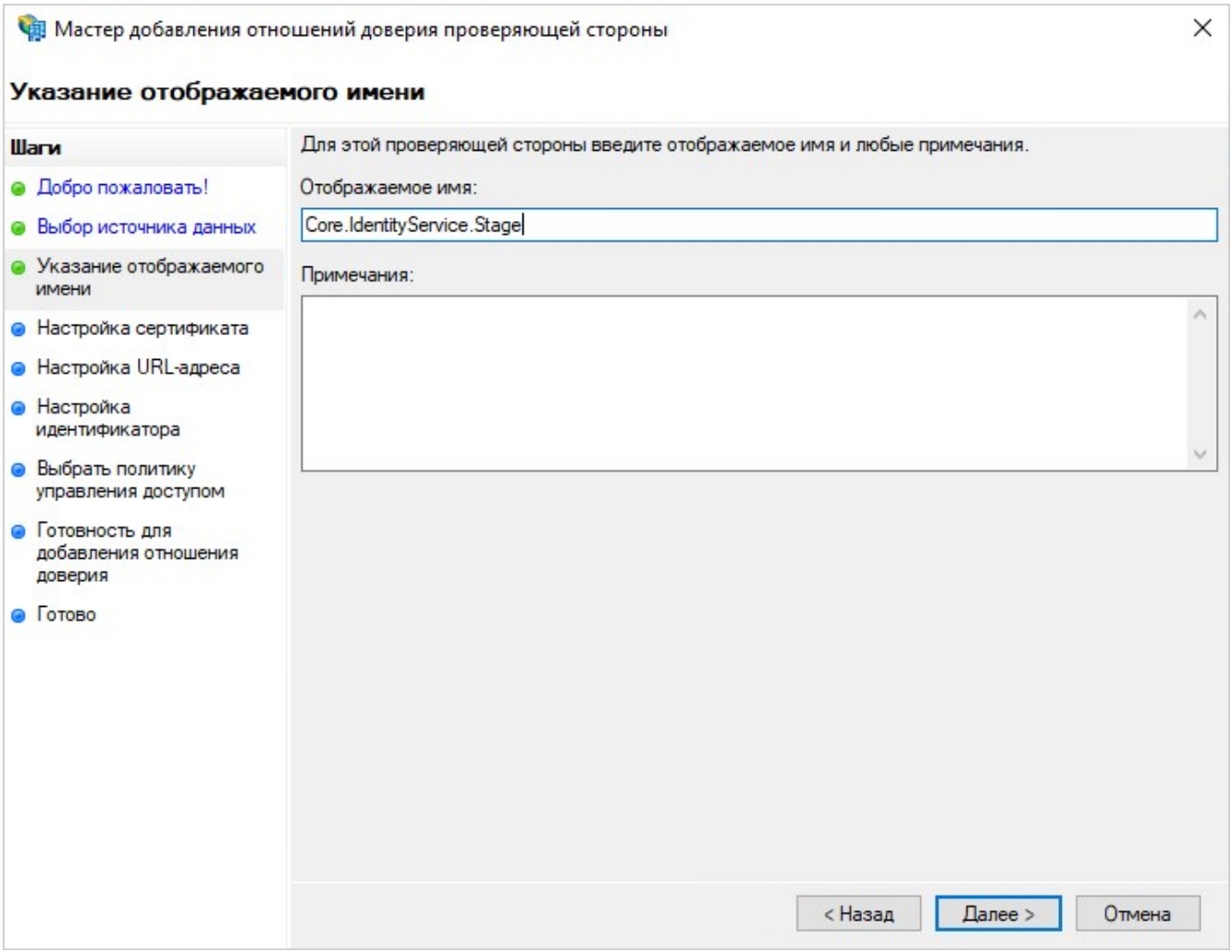
4. В окне «Добро пожаловать!» установите переключатель **Поддерживающие утверждения** и нажмите на кнопку **Запустить**:



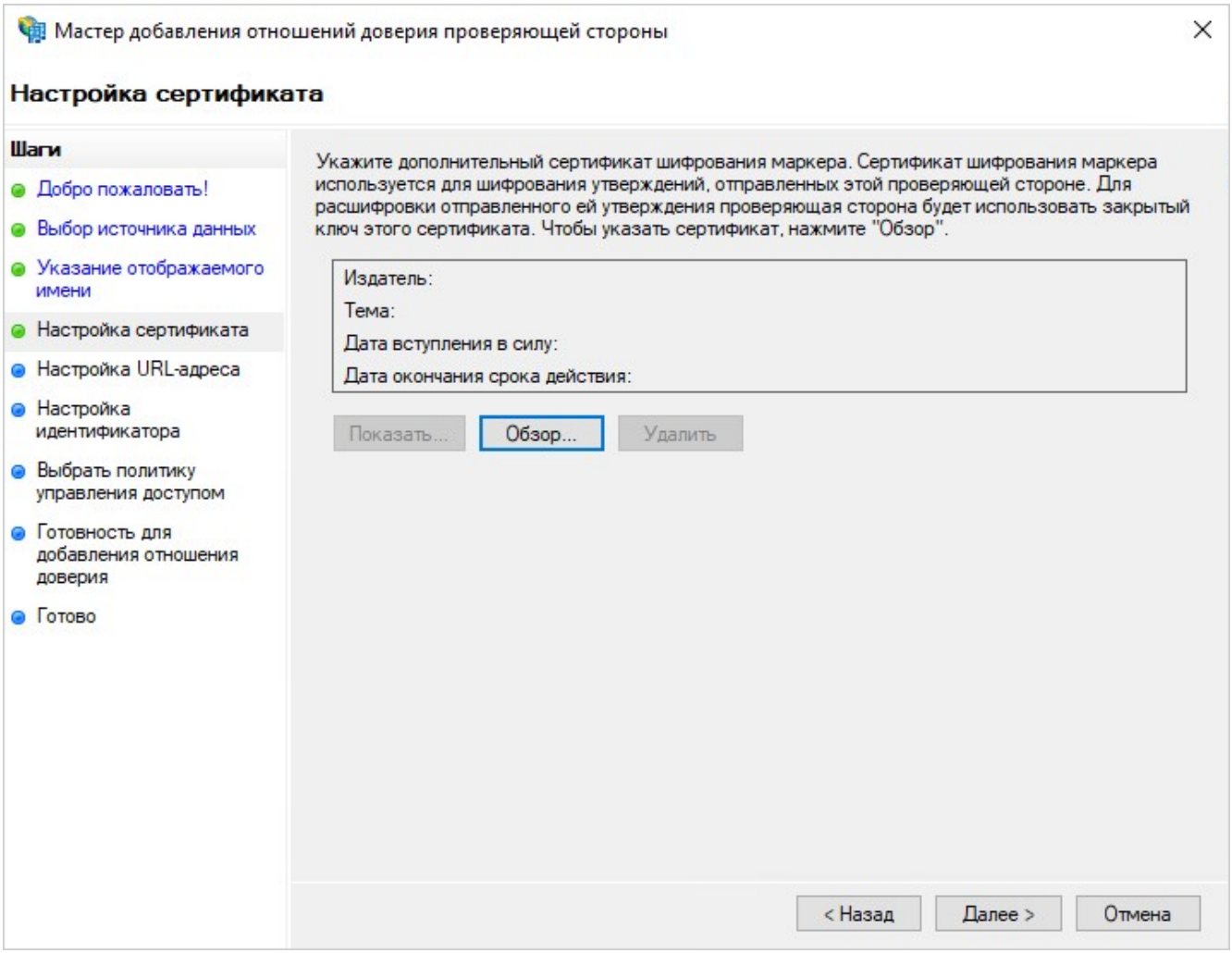
5. В окне «Выбор источника данных» установите переключатель **Ввод данных о проверяющей стороне вручную** и нажмите на кнопку **Далее>**:



6. В окне «Указание отображаемого имени» в поле **Отображаемое имя:** введите имя проверяющей стороны, которое будет отображаться в списке «Отношения доверия проверяющей стороны», например Core.IdentityService.Stage, и нажмите на кнопку **Далее>**:



7. В окне «Настройка сертификата»:
   - если необходимо указать дополнительный сертификат шифрования, нажмите на кнопку **Обзор**:

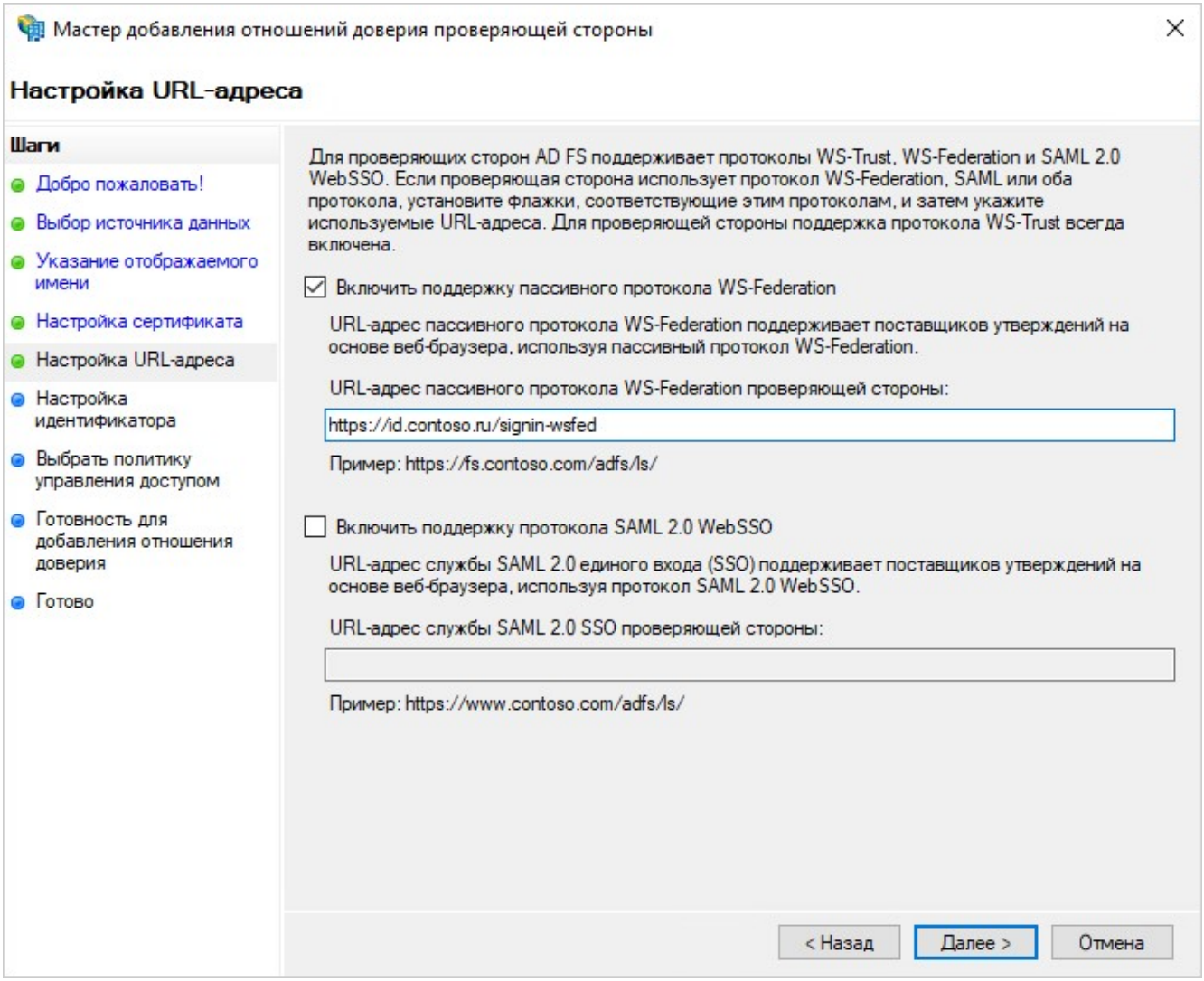


     В открывшемся окне выберите хранилище сертификата и укажите нужный сертификат. Нажмите на кнопку **Далее>**.
   - если указывать дополнительный сертификат шифрования не нужно, нажмите на кнопку **Далее>**.

8. В окне «Настройка URL-адреса» установите флажок **Включить поддержку пассивного протокола WS-Federation**.

   В качестве URL-адреса пассивного протокола WS-Federation проверяющей стороны укажите доверенный адрес с конечной точкой типа WS-Federation. Пример: **https://id.contoso.ru/signin-wsfed**.

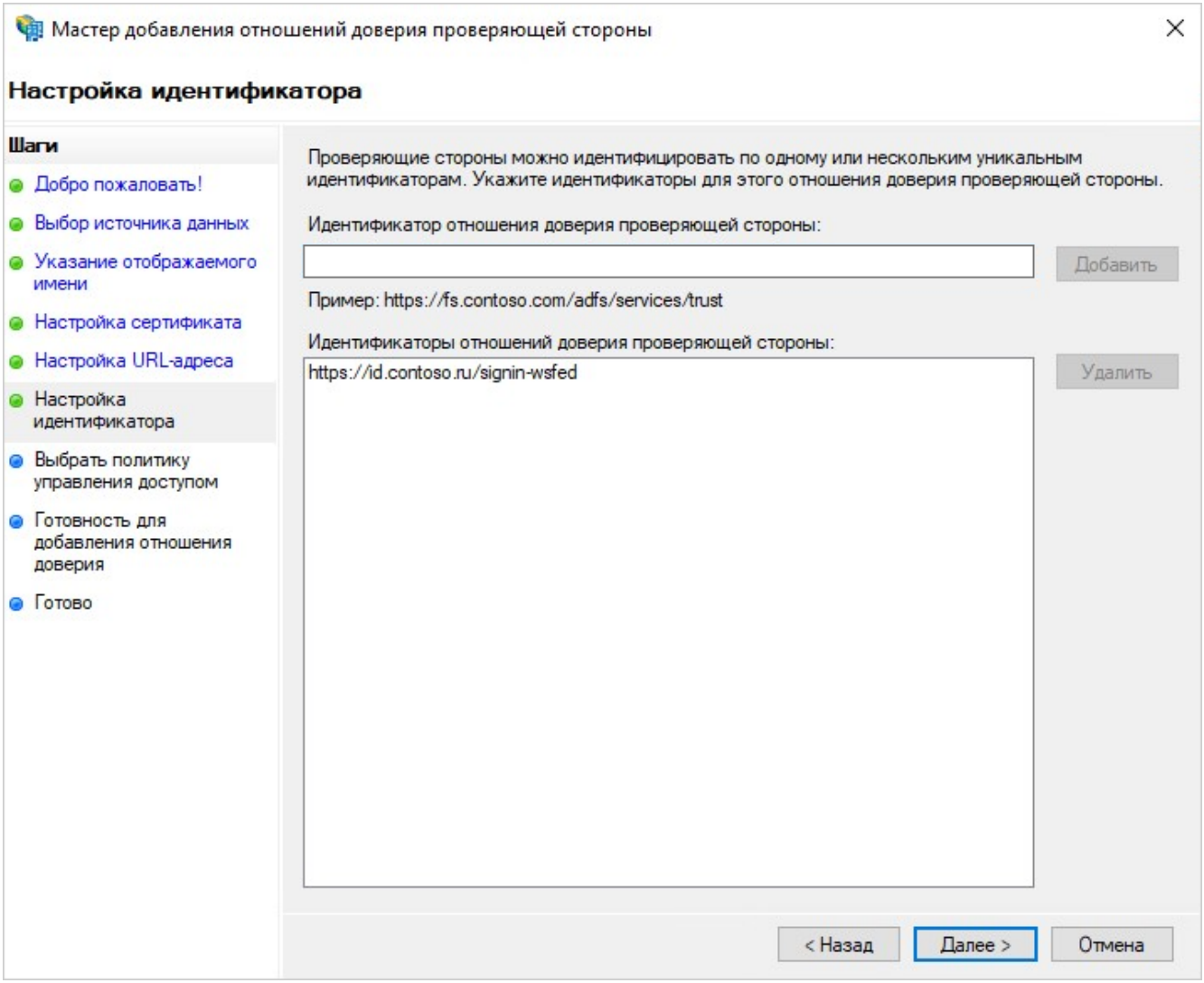
   Нажмите на кнопку **Далее>**:



9. В окне «Настройка идентификатора» в поле **Идентификатор отношения доверия проверяющей стороны:** укажите адрес идентификатора и нажмите на кнопку **Добавить**.

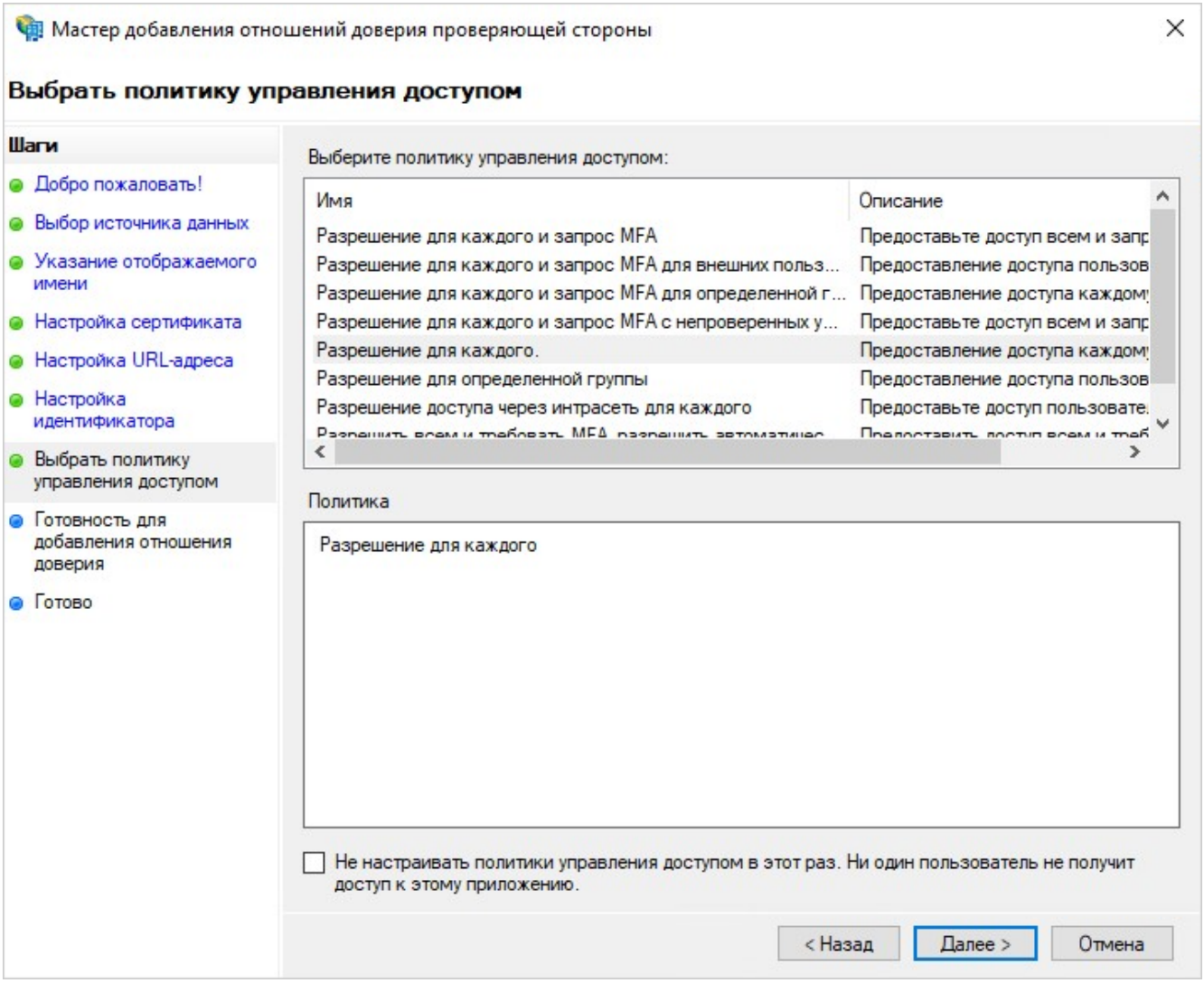
   Пример адреса: **https://id.contoso.ru/signin/adfs**. Адрес отображается в поле **Идентификаторы отношений доверия проверяющей стороны:**.

   Нажмите на кнопку **Далее>**:

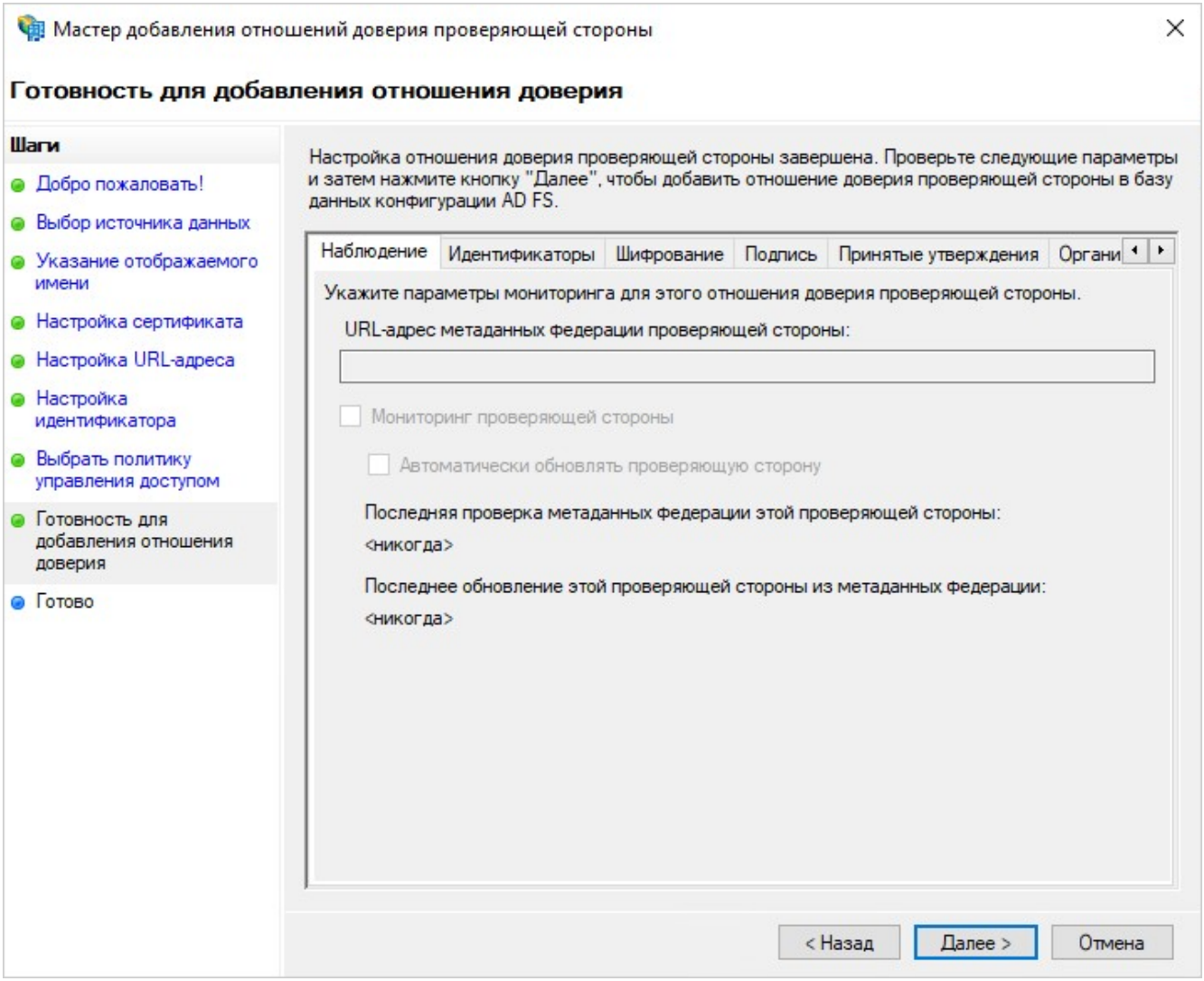


   Работа мастера завершается.

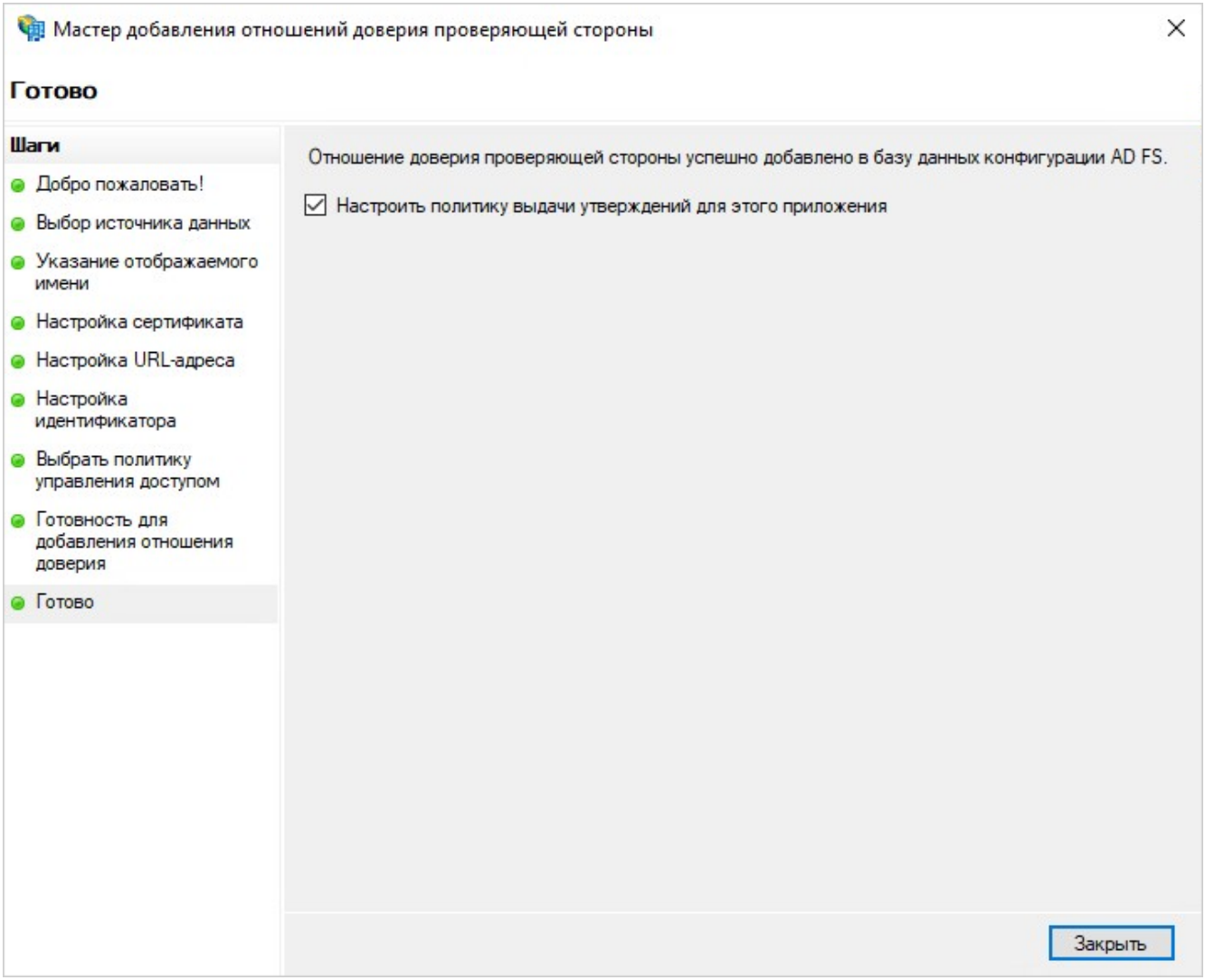
10. В окне «Выбрать политику управления доступом» выберите нужную политику из списка:



11. В окне «Готовность для добавления отношения доверия» проверьте заданные настройки:

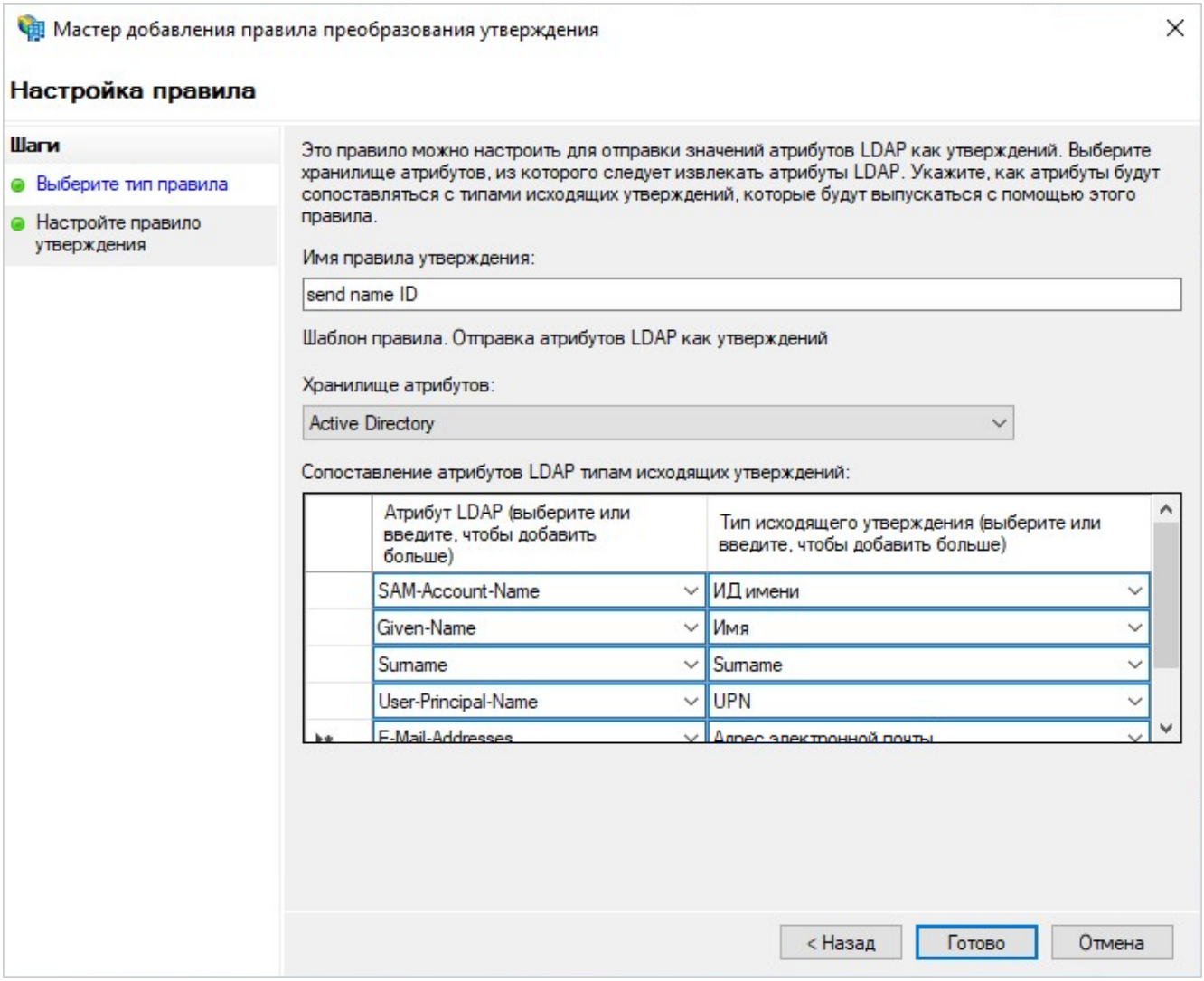


12. В окне «Готово» проверьте, что флажок **Настроить политику выдачи утверждений для этого приложения** установлен и нажмите на кнопку **Закрыть**:



13. После завершения работы мастера открывается окно «Выбор шаблона правила». Оставьте значение по умолчанию и нажмите на кнопку **Далее>**.

14. В окне «Изменение правил утверждений» добавьте новое правило:
    - в поле **Имя правила утверждения** укажите для него произвольное имя;
    - в поле **Хранилище атрибутов** укажите Active Directory;
    - добавьте сопоставления атрибутов LDAP и типов исходящих утверждений:



Значения зависят от используемой службы AD FS. Пример:

SAM-Account-Name – ИД имени;

Given-Name – Имя;

Surname – Surname;

User-Principal-Name – UPN;

Display-Name – Общее имя.

Нажмите на кнопку **Готово**.

Настройка плагина провайдера аутентификации

В конфигурационный файл identity-service.env добавьте параметры:

- **Authentication__WsFederation__MetadataAddress** – адрес для запроса метаданных сервиса AD FS.

  Пример: **https://adfs.contoso.ru/FederationMetadata/2007-06/FederationMetadata.xml**;

- **Authentication__WsFederation__Wtrealm** – идентификатор сервиса идентификации в AD FS.

  Пример: **https://id.contoso.ru/signin/adfs**.

  Пример настройки:

```
Authentication__WsFederation__MetadataAddress:
'https://adfs.contoso.ru/FederationMetadata/2007-06/FederationMetadata.xml'
Authentication__WsFederation__Wtrealm: 'https://id.contoso.ru/signin/adfs'
```

- добавьте блок параметров для подключения плагина **AdfsAuthenticationProvider** и заполните его как указано в примере.

  При необходимости в параметре **ChangePasswordPath** укажите путь к странице смены пароля пользователя на портале AD FS.

  Пример настройки:

```
Authentication__Providers__0__Name: 'AdfsAuthenticationProvider'
Authentication__Providers__0__AssemblyFileName:
'Directum.IdentityService.AuthenticationProviders.AdfsAuthenticationProvider.dll'
Authentication__Providers__0__Type:
'Directum.IdentityService.AuthenticationProviders.AdfsAuthenticationProvider.AdfsAuthenticatio
nProvider, Directum.IdentityService.AuthenticationProviders.AdfsAuthenticationProvider,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=427ba5252f628cb0'
Authentication__Providers__0__Audiences: 'Directum.Omni'
Authentication__Providers__0__Configuration__ChangePasswordPath:
'https://adfs.contoso.ru/adfs/portal/updatepassword/'
```

СОВЕТ. Чтобы настроить отображение провайдера, добавьте соответствующие настройки. Подробнее см. раздел «Настройка отображения внешних провайдеров аутентификации».

Проверка корректности настройки

1. Откройте сайт сервиса идентификации.
2. В конце адресной строки добавьте параметры **returnUrl** и **audience** и их значения. Пример: **returnUrl=https://test&audience=Directum.Omni**.
3. Обновите страницу.
4. В окне входа нажмите на кнопку **Войти с ADFS**. Открывается сайт AD FS для прохождения аутентификации.

   Если настройка корректна, после прохождения аутентификации открывается страница **https://test**.

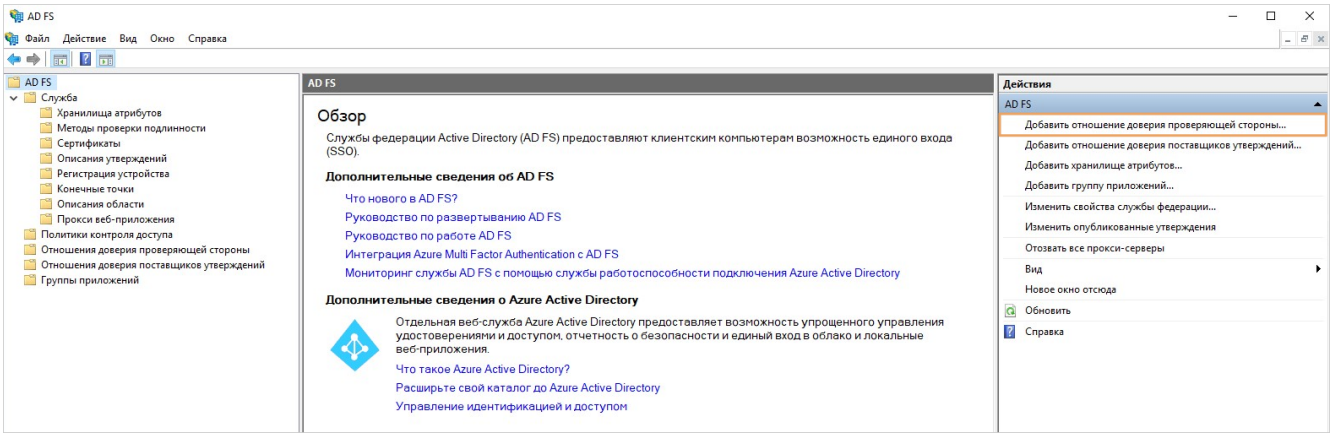
AD FS по протоколу SAML 2.0

1. Для аутентификации в AD FS используется протокол SAML 2.0. Перед началом настройки убедитесь, что он доступен в службе.
2. Настройте сервис AD FS.
3. Если используется Windows 2016, а также провайдер аутентификации как единая точка входа во все корпоративные приложения (IdP-initiated), то настройте страницу входа в провайдер аутентификации.
4. Настройте конечную точку проверочного удостоверения SAML.
5. Настройте плагин провайдера аутентификации в сервисе идентификации.

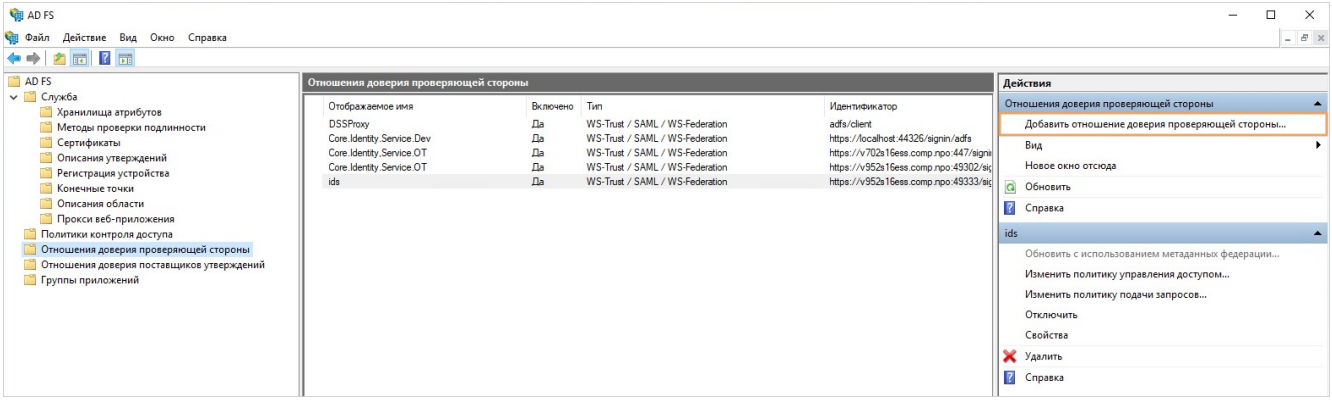
6. Проверьте корректность настройки.

Настройка AD FS

1. Перед началом настройки сохраните в удобной форме адрес сервиса идентификации. Пример: **https://id.contoso.ru**.
2. В диспетчере серверов в выпадающем списке **Средства** выберите пункт **Управление AD FS**. Откроется окно утилиты.
3. Добавьте отношение доверия проверяющей стороны одним из способов:
   - перейдите в узел **AD FS** и в разделе «Действия» выберите пункт **Добавить отношение доверия проверяющей стороны...**:

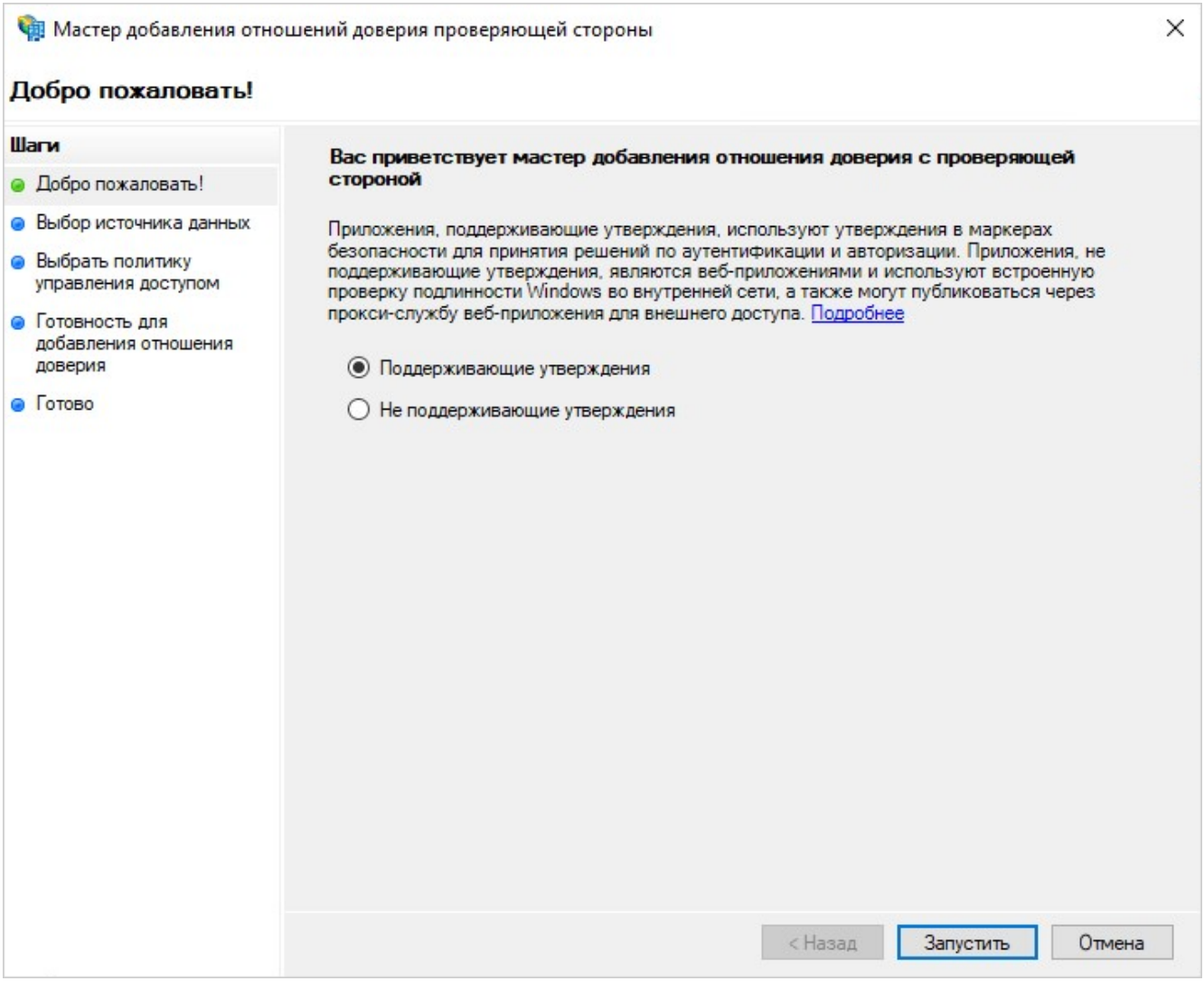


   - перейдите в узел **Отношения доверия проверяющей стороны** и в разделе «Действия» выберите пункт **Добавить отношение доверия проверяющей стороны...**:

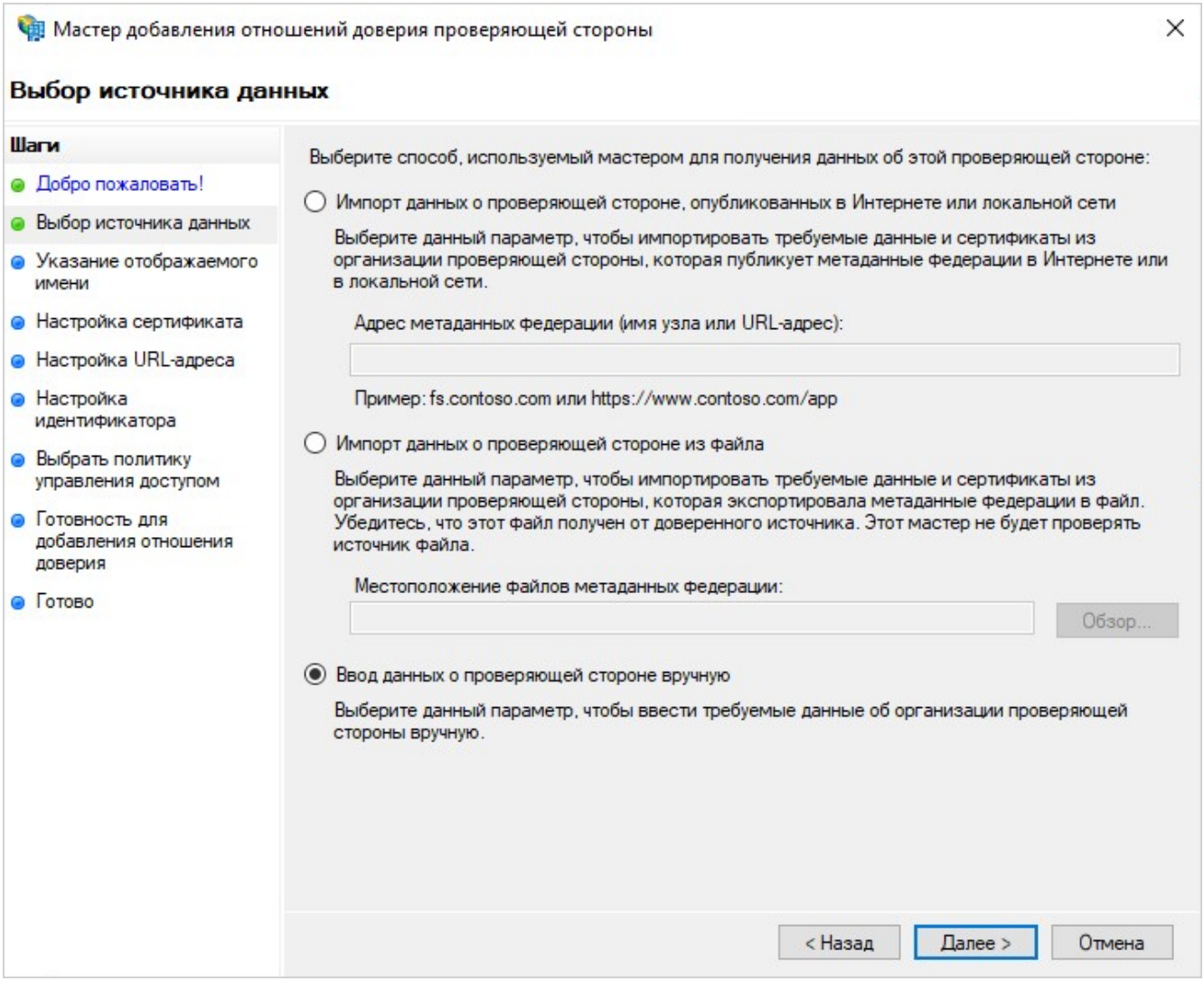


Откроется окно «Мастер добавления отношений проверяющей стороны».

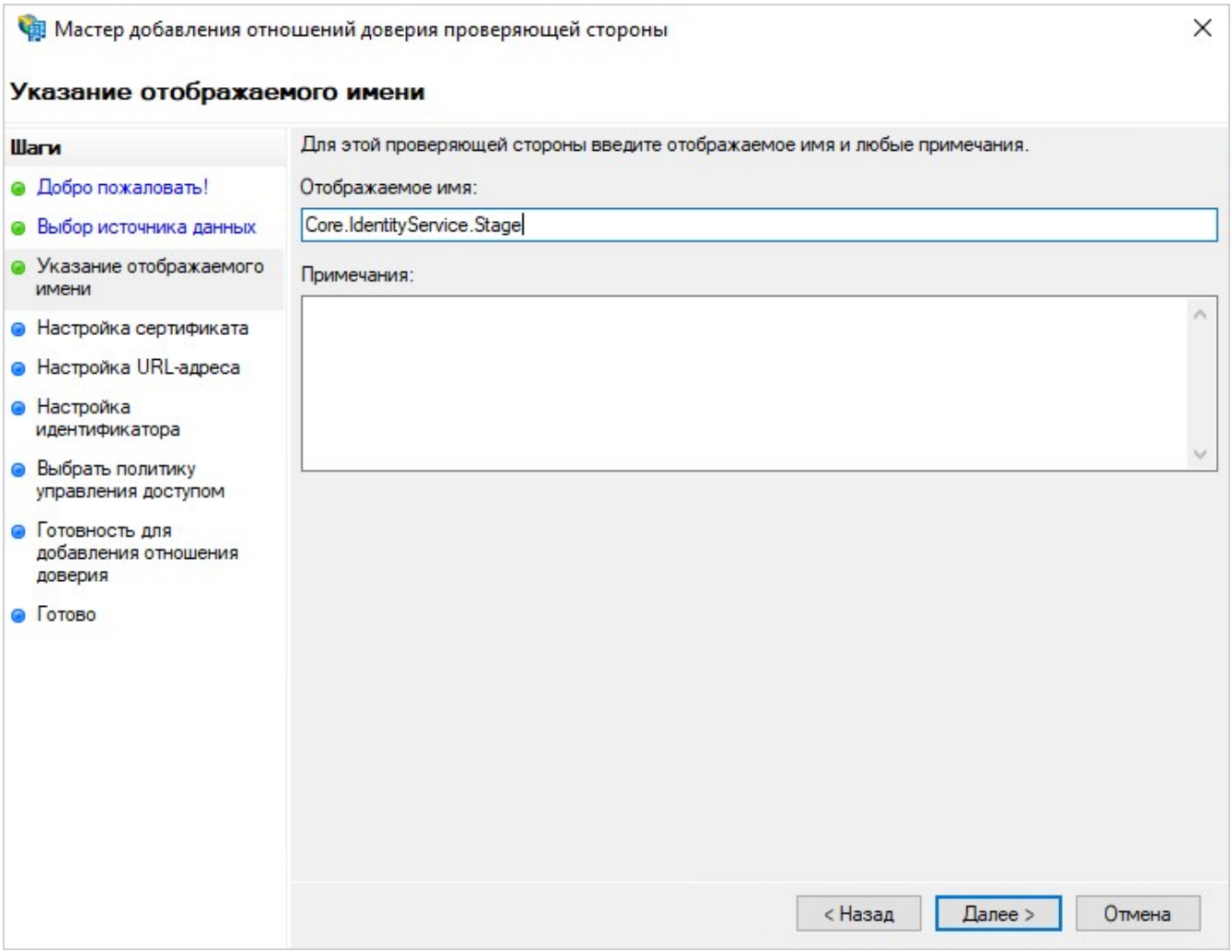
4. В окне «Добро пожаловать!» установите переключатель **Поддерживающие утверждения** и нажмите на кнопку **Запустить**:



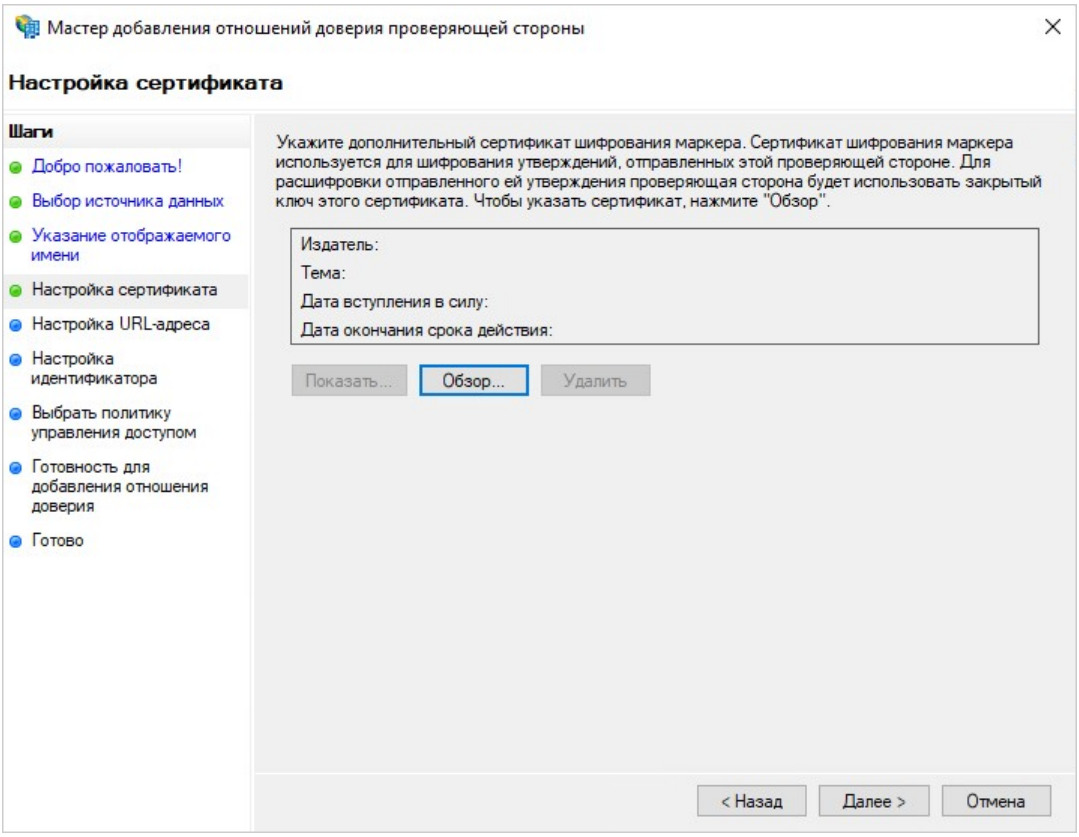
5. В окне «Выбор источника данных» установите переключатель **Ввод данных о проверяющей стороне вручную** и нажмите на кнопку **Далее>**:



6. В окне «Указание отображаемого имени» в поле **Отображаемое имя:** введите имя проверяющей стороны, которое будет отображаться в списке «Отношения доверия проверяющей стороны», например Core.IdentityService.Stage, и нажмите на кнопку **Далее>**:



7. В окне «Настройка сертификата»:
   - если необходимо указать дополнительный сертификат шифрования, нажмите на кнопку **Обзор**:

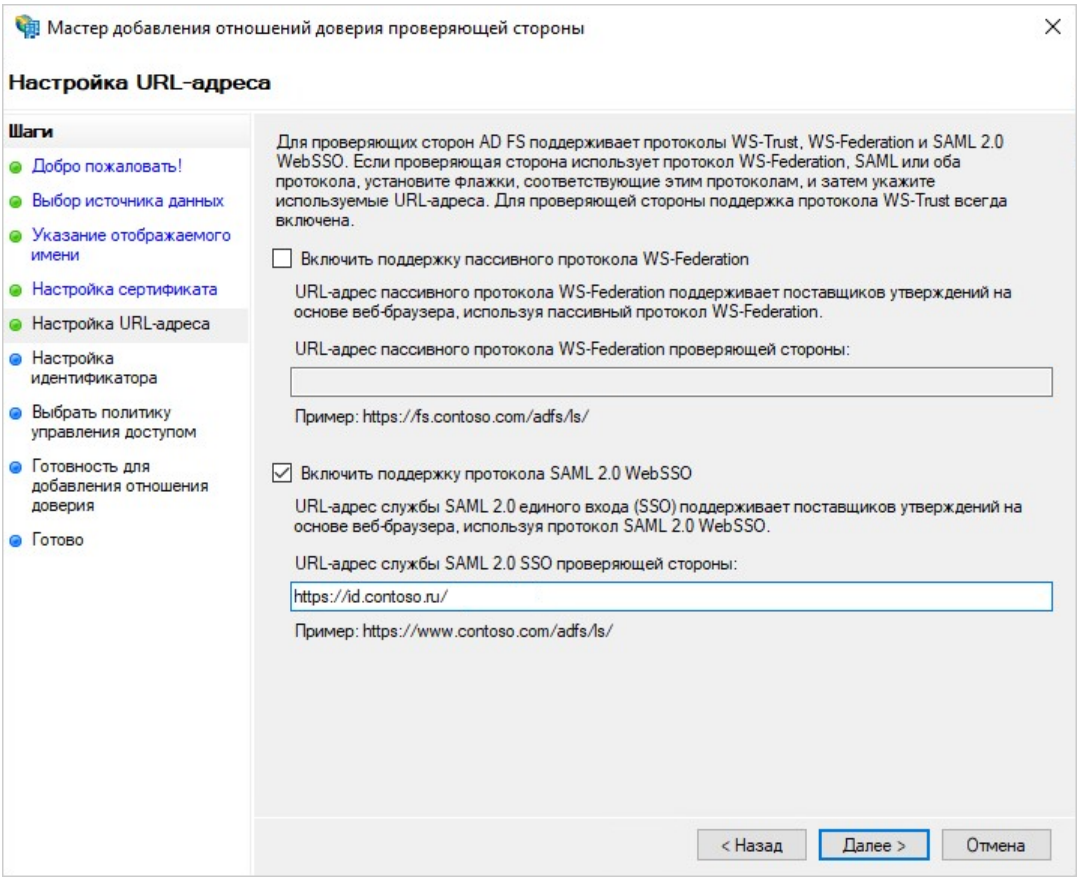


     В открывшемся окне выберите хранилище сертификата и укажите нужный сертификат. Нажмите на кнопку **Далее>**.
   - если указывать дополнительный сертификат шифрования не нужно, нажмите на кнопку **Далее>**.

8. В окне «Настройка URL-адреса» установите флажок **Включить поддержку протокола SAML 2.0 WebSSO**.

   В качестве URL-адреса службы SAML 2.0 SSO проверяющей стороны укажите доверенный адрес. Пример: **https://id.contoso.ru/**.

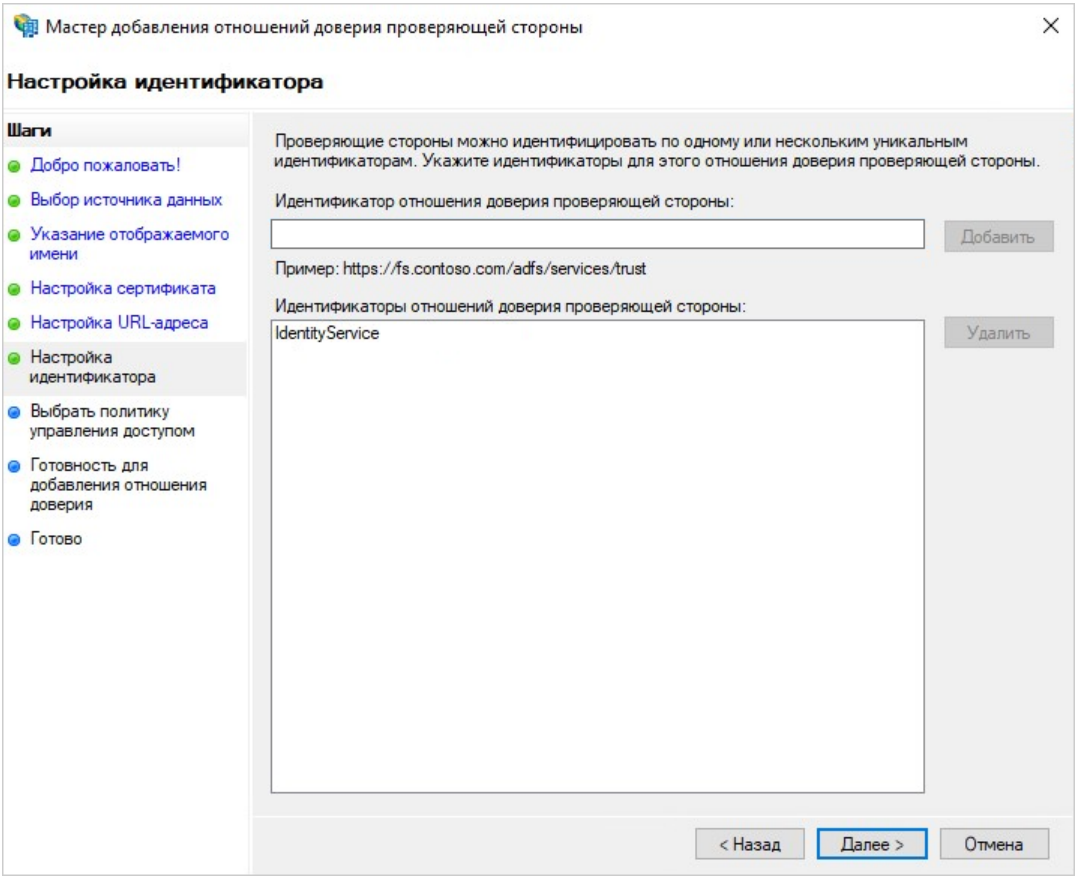
   Нажмите на кнопку **Далее>**:



9. В окне «Настройка идентификатора» в поле **Идентификатор отношения доверия проверяющей стороны:** укажите значение **IdentityService** и нажмите на кнопку **Добавить**.

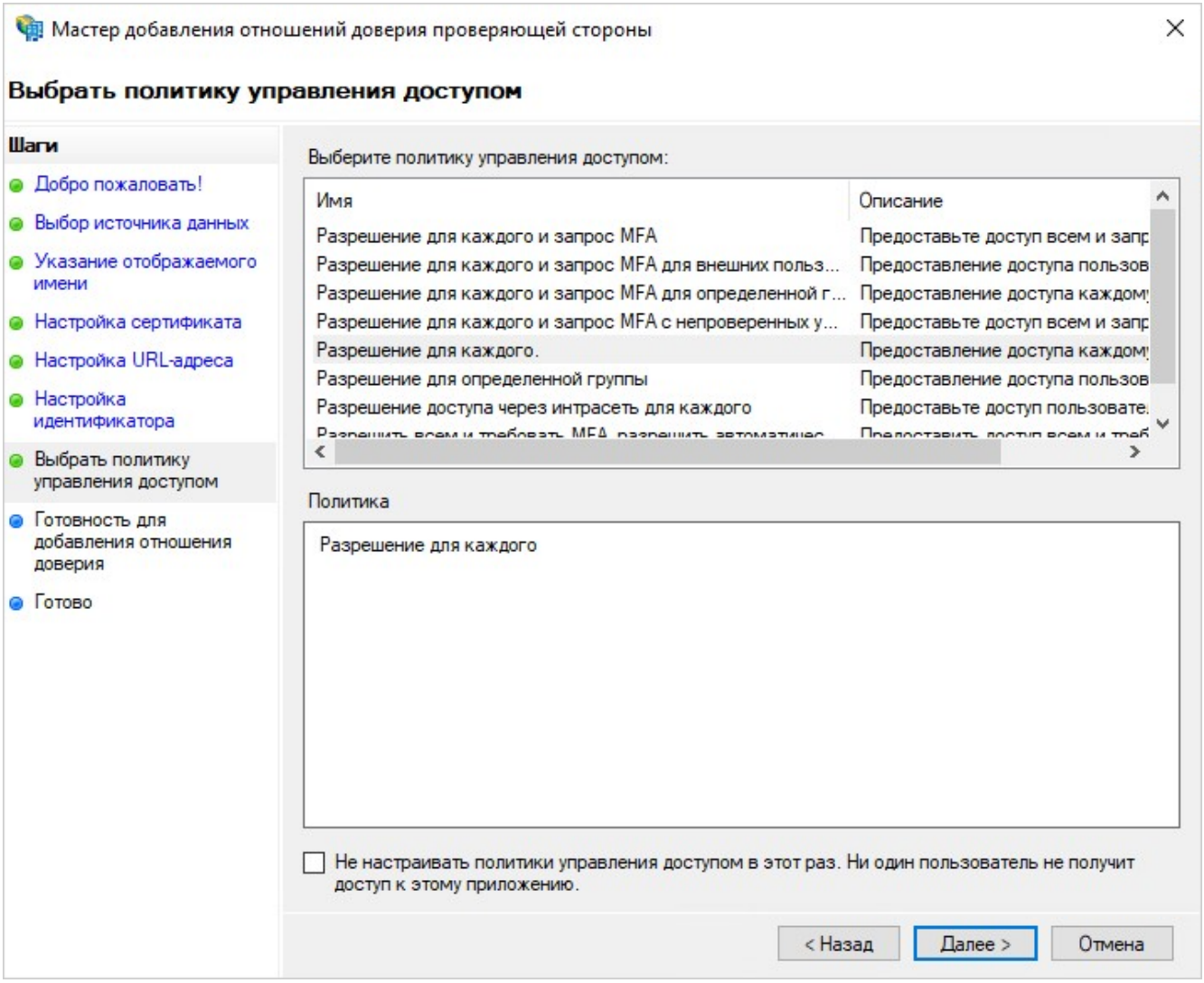
   Значение отображается в поле **Идентификаторы отношений доверия проверяющей стороны:**.

   Нажмите на кнопку **Далее>**:

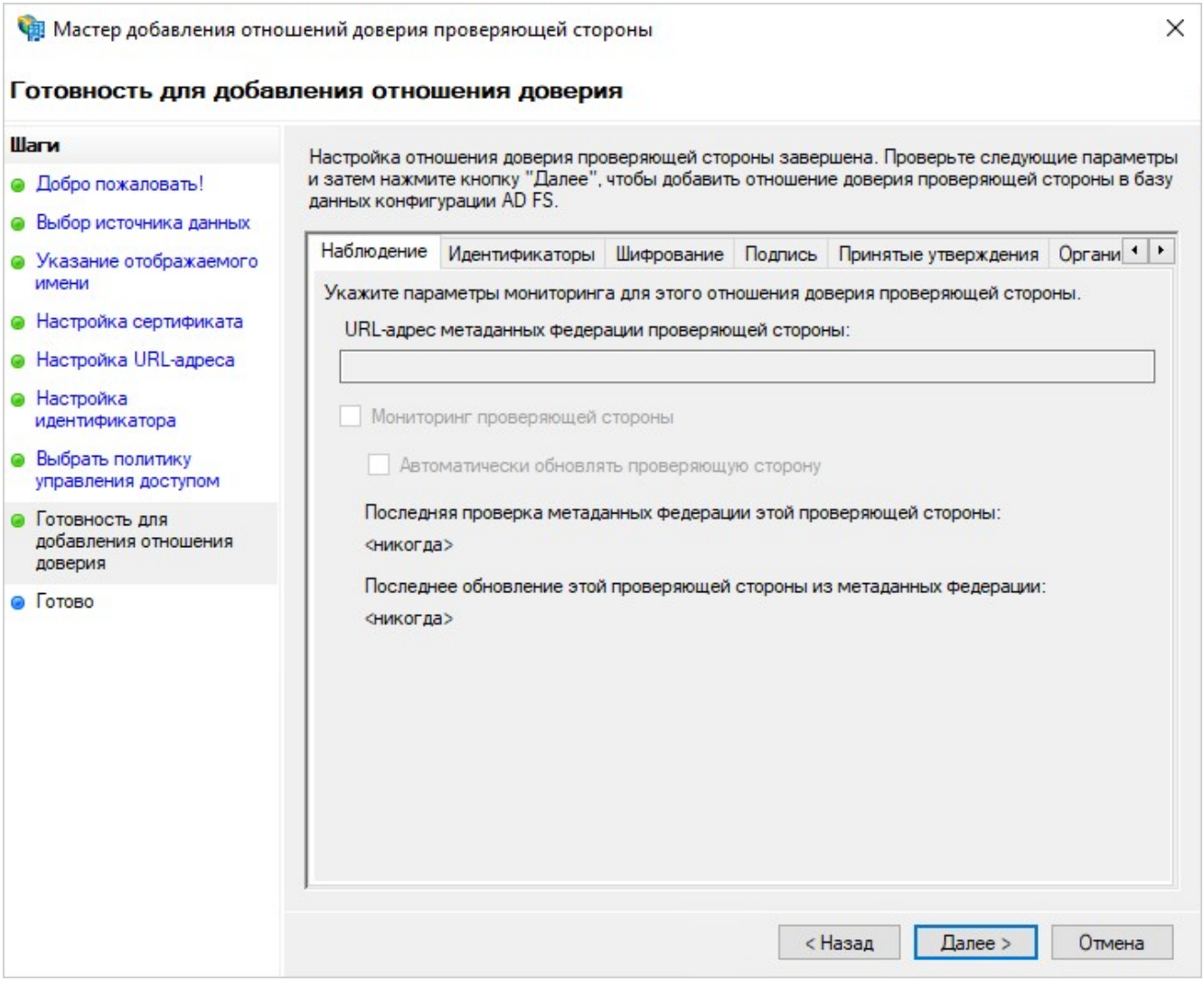


   Работа мастера завершается.

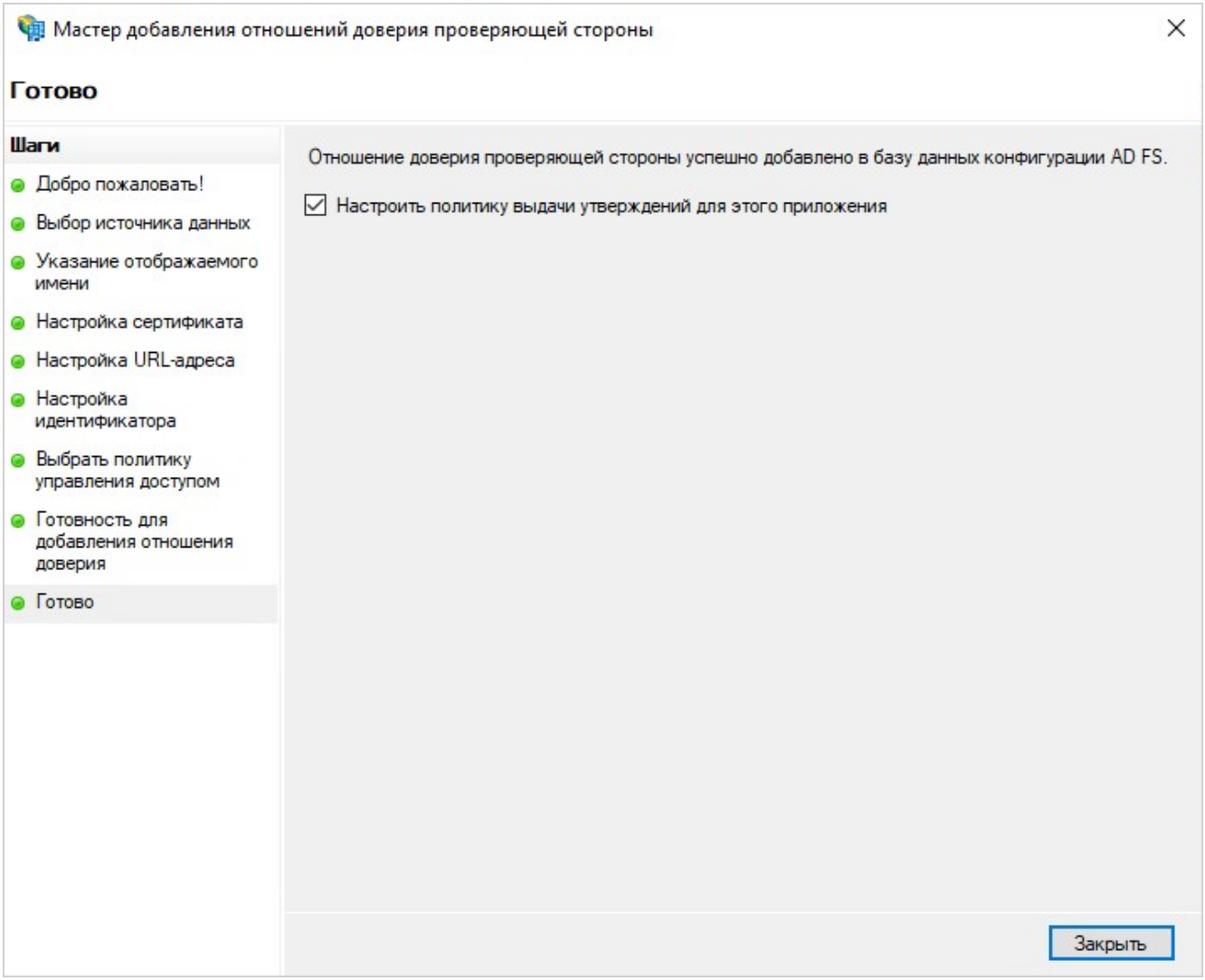
10. В окне «Выбрать политику управления доступом» выберите нужную политику из списка:



11. В окне «Готовность для добавления отношения доверия» проверьте заданные настройки:

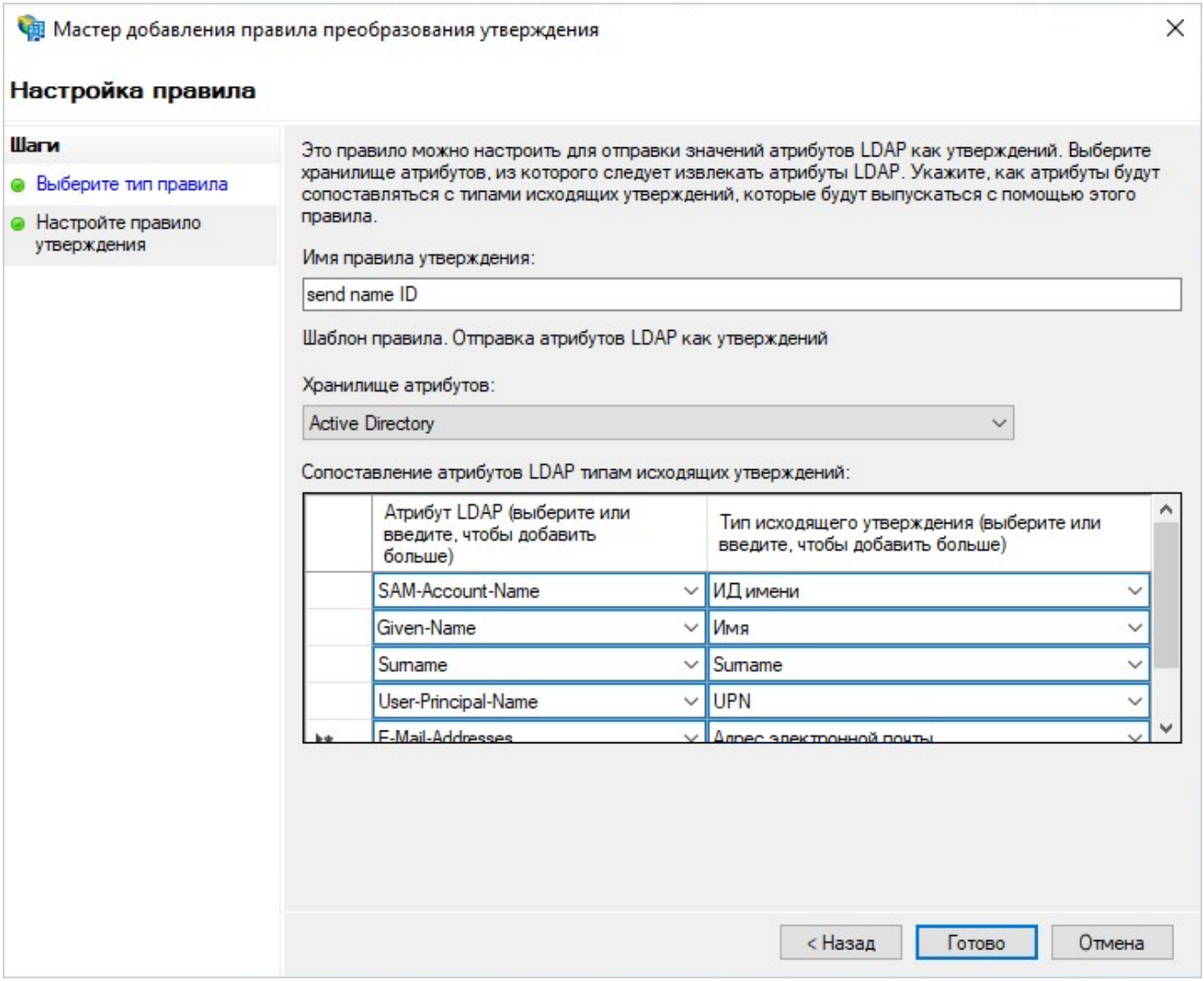


12. В окне «Готово» проверьте, что флажок **Настроить политику выдачи утверждений для этого приложения** установлен и нажмите на кнопку **Закрыть**:



13. После завершения работы мастера открывается окно «Выбор шаблона правила». Оставьте значение по умолчанию и нажмите на кнопку **Далее>**.

14. В окне «Изменение правил утверждений» добавьте новое правило:
    - в поле **Имя правила утверждения** укажите для него произвольное имя;
    - в поле **Хранилище атрибутов** укажите Active Directory;
    - добавьте сопоставления атрибутов LDAP и типов исходящих утверждений:



Значения зависят от используемой службы AD FS. Пример:

SAM-Account-Name – ИД имени;

Given-Name – Имя;

Surname – Surname;

User-Principal-Name – UPN;

Display-Name – Общее имя.

Нажмите на кнопку **Готово**.

ПРИМЕЧАНИЕ. Работа с узлом AD FS продолжается при настройке конечной точки проверочного удостоверения SAML. Закрывать его окно не обязательно.

Настройка страницы входа в провайдер аутентификации

ВАЖНО. Настройку необходимо выполнять, только если используется:

- провайдер аутентификации как единая точка входа во все корпоративные приложения (IdP-initiated);
- Windows 2016.

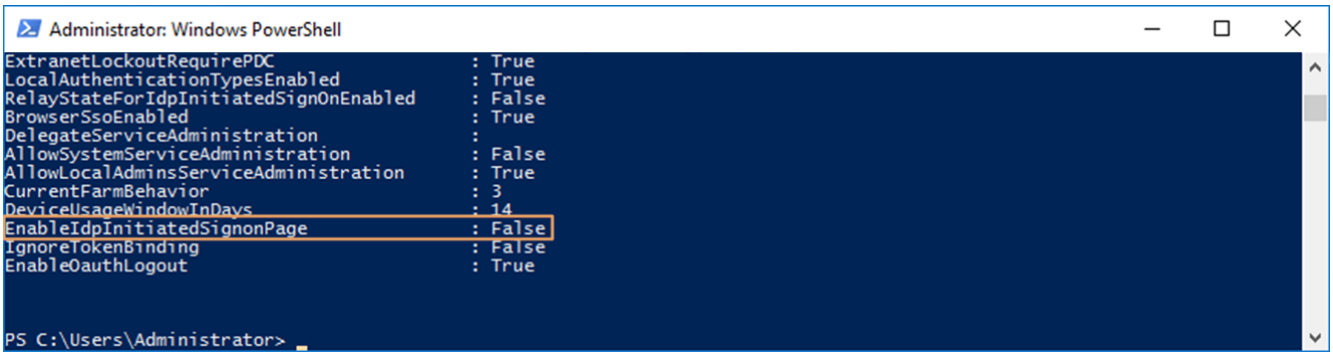
В других случаях перейдите к настройке конечной точки проверочного удостоверения SAML.

Настройте страницу входа в провайдер аутентификации:

1. Запустите утилиту Windows PowerShell от имени администратора.
2. В открывшемся окне выполните команду:

   ```
   Get-AdfsProperties
   ```

3. Убедитесь, что для свойства **EnableIdpInitiatedSignonPage** указано значение **false**:



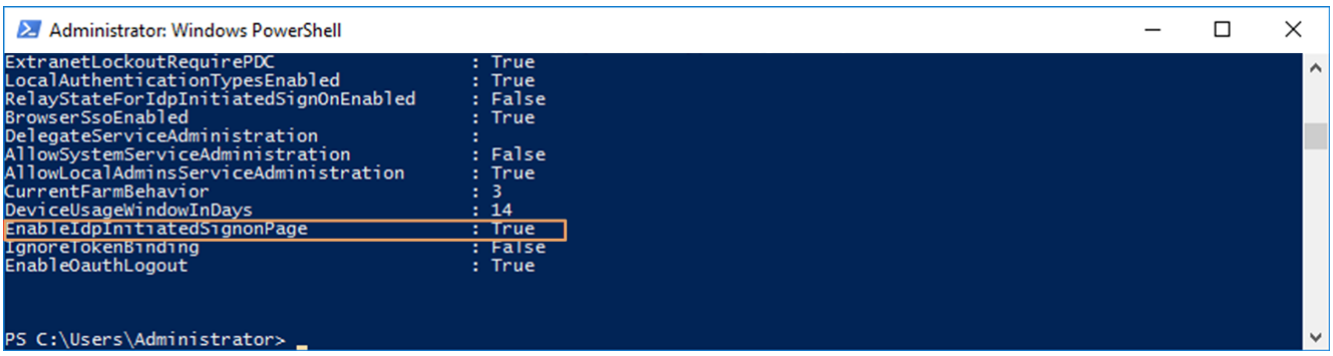
4. В Windows PowerShell выполните команду:

   ```
   Set-AdfsProperties -EnableIdpInitiatedSignonPage $true
   ```

5. Повторно выполните команду:

   ```
   Get-AdfsProperties
   ```

6. Проверьте, что для свойства **EnableIdpInitiatedSignonPage** указано значение **true**:



Настройка конечной точки проверочного удостоверения SAML

Настройка выполняется в окне «AD FS».

1. В узле **AD FS** откройте свойства созданного отношения доверия проверяющей стороны.

2. На вкладке **Конечные точки** нажмите на кнопку **Добавить SAML...**
3. В открывшемся окне «Добавить конечную» точку укажите параметры:

   Тип конечной точки – «Получатель проверочного удостоверения SAML»;

   Привязка – «POST»;

   Индекс – «0». Если ранее были заданы другие конечные точки, их индекс должен быть больше нуля;

   Доверенный URL-адрес – укажите значение в формате:

   <Адрес сервиса идентификации>/<Путь, по которому сервис идентификации принимает ответы по протоколу SAML}?audience=<Ресурс, в котором необходимо авторизоваться>&ReplyUrl=<Адрес для авторизации>

   Пример: **https://id.contoso.ru/signin-saml?audience=Directum.Omni&ReplyUrl=https://contoso.ru/Authorize**

Настройка плагина провайдера аутентификации

1. Создайте сущность config для сертификата, который предоставил SAML-провайдер. Для этого выполните команду:

   ```
   docker config create --label certificate_type=ssl <Название сущности>
   <Полный путь до сертификата>
   ```

   Пример команды:

   ```
   docker config create --label certificate_type=ssl
   saml.crt /opt/certificates/saml.crt
   ```

   В результате сертификат монтируется в Docker-контейнеры в папку /etc/pki/ca-trust/source/anchors/.
2. В конфигурационный файл identity-service.env добавьте параметры:
   - **Authentication__Saml__CertificatePath** – путь до сертификата внутри контейнера;
   - **Authentication__Saml__ProviderEndpoint** – адрес конечной точки SAML-провайдера. Пример: https://adfs.contoso.ru/adfs/ls;
   - **Authentication__Saml__ValidateResponsePath** – путь, по которому сервис идентификации принимает ответы по протоколу SAML. Пример: /signin-saml;
   - **Authentication__Saml__ClientId** – значение, указанное для идентификатора отношений доверия проверяющей стороны.

   Пример настройки:

   ```
   Authentication__Saml__CertificatePath: '/etc/pki/ca-trust/source/anchors/saml.cert'
   Authentication__Saml__ProviderEndpoint: 'https://adfs.contoso.ru/adfs/ls'
   Authentication__Saml__ValidateResponsePath: '/signin-saml'
   Authentication__Saml__ClientId: 'IdentityService'
   ```
3. Добавьте блок параметров для подключения плагина **SamlAuthenticationProvider** и заполните его как указано в примере.

   В параметре **ChangePasswordPath** укажите путь к странице смены пароля пользователя на портале AD FS. Необязательный параметр.

   Пример настройки:

   ```
   Authentication__Providers__0__Name: 'SamlAuthenticationProvider'
   Authentication__Providers__0__Type:
   'Directum.IdentityService.AuthenticationProviders.SamlAuthenticationProvider.SamlAuthen
   ticationProvider,
   ```

```
Directum.IdentityService.AuthenticationProviders.SamlAuthenticationProvider,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=427ba5252f628cb0'
Authentication__Providers__0__Audiences: 'Directum.Omni'
Authentication__Providers__0__Configuration__ChangePasswordPath:
'https://adfs.contoso.ru/adfs/portal/updatepassword/'
```

СОВЕТ. Чтобы настроить отображение провайдера, добавьте соответствующие настройки. Подробнее см. раздел «Настройка отображения внешних провайдеров аутентификации».

Проверка корректности настройки

1. Откройте сайт сервиса идентификации.
2. В конце адресной строки добавьте параметры **returnUrl** и **audience** и их значения. Пример: **returnUrl=https://test&audience=Directum.Omni**.
3. Обновите страницу.
4. В окне входа нажмите на кнопку **Войти с ADFS**. Открывается сайт AD FS для прохождения аутентификации.

   Если настройка корректна, после прохождения аутентификации открывается страница **https://test**.
5. Если настроена страница входа в провайдер аутентификации, то выполните дополнительную проверку, открыв эту страницу в браузере. Пример URL-адреса: https://sts.contoso.com/adfs/ls/idpinitiatedsignon.aspx. Если настройка корректна, то после аутентификации в AD FS открывается страница провайдера.

Google

Для аутентификации в Google используется протокол OAuth 2.0.

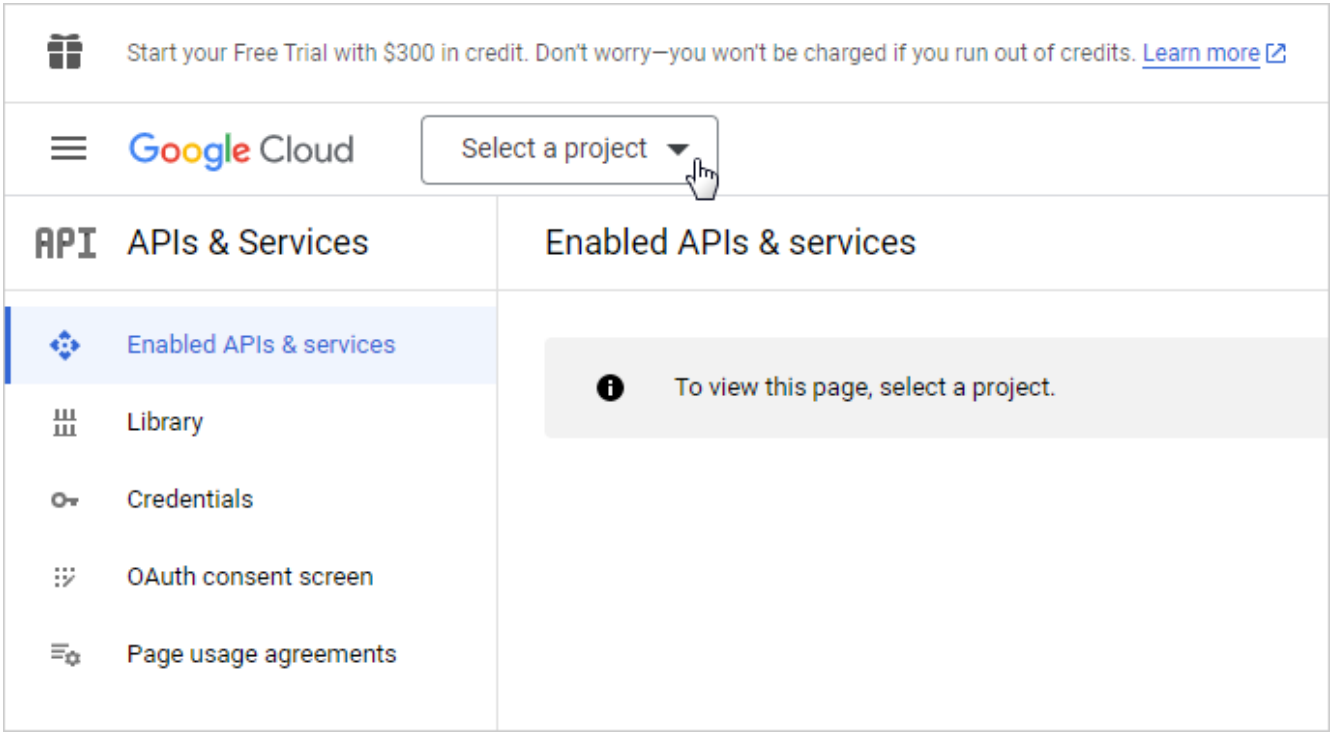
1. Зарегистрируйте приложение в Google OAuth 2.0.
2. Настройте плагин провайдера аутентификации в сервисе идентификации.
3. Проверьте корректность настройки.

Регистрация приложения в Google OAuth 2.0

В разделе приведен общий порядок создания клиента:

1. Перед началом настройки сохраните в удобной форме адрес сервиса идентификации. Пример: **https://id.contoso.ru**.
2. Откройте консоль разработчиков Google API & Services.

3. В верхнем левом меню выберите пункт **Select a project**:



Открывается окно создания проекта. Создайте новый проект или выберите существующий.

4. Перейдите на вкладку **Dashboard** и откройте раздел с настройками экрана согласия **OAuth consent screen**.
5. В настройке **User Type** укажите **External** и нажмите на кнопку **CREATE**.
6. В диалоговом окне придумайте и укажите имя приложения, адрес электронной почты технической поддержки и контактные данные разработчиков.
7. Пропустите шаги **Scopes** и **Test users**.
8. Проверьте, что параметры на экране **OAuth consent screen** указаны верно, и вернитесь на вкладку **Dashboard**.
9. В разделе **Credentials** последовательно выберите **CREATE CREDENTIALS** и **OAuth client ID**.
10. В поле **Application type** укажите значение **Web application** и введите указанное ранее имя приложения.
11. Для установки адреса перенаправления в настройке **Authorized redirect URIs** добавьте URI-адрес по кнопке **ADD URI**. Пример адреса: **https://id.contoso.ru/signin-google**.
12. Нажмите кнопку **CREATE**.
13. Сохраните значения параметров **Client ID** и **Client Secret** для настройки сервиса идентификации.

Настройка плагина провайдера аутентификации

В конфигурационный файл identity-service.env добавьте параметры:

- **Authentication__Google__ClientId** – идентификатор клиентского приложения в Google;

- **Authentication__Google__ClientSecret** – пароль клиентского приложения в Google.

  Пример настройки:

  ```
  Authentication__Google__ClientId: '123456768-abcdef.apps.googleusercontent.com'
  Authentication__Google__ClientSecret: 'ABCDEF-12345678ABCDEFTHS'
  ```

- добавьте блок параметров для подключения плагина GoogleAuthenticationProvider и заполните его как указано в примере:

  ```
  Authentication__Providers__0__Name: 'GoogleAuthenticationProvider'
  Authentication__Providers__0__AssemblyFileName:
  'Directum.IdentityService.AuthenticationProviders.AdfsAuthenticationProvider.dll'
  Authentication__Providers__0__Type:
  'Directum.IdentityService.AuthenticationProviders.GoogleAuthenticationProvider.GoogleAuthen
  ticationProvider,
  Directum.IdentityService.AuthenticationProviders.GoogleAuthenticationProvider,
  Version=1.0.0.0, Culture=neutral, PublicKeyToken=427ba5252f628cb0'
  Authentication__Providers__0__Audiences: 'Directum.Omni'
  Authentication__Providers__0__Configuration: ''
  ```

СОВЕТ. Чтобы настроить отображение провайдера, добавьте соответствующие настройки. Подробнее см. раздел «Настройка отображения внешних провайдеров аутентификации».

Проверка корректности настройки

1. Откройте сайт сервиса идентификации.
2. В конце адресной строки добавьте параметры **returnUrl** и **audience** и их значения. Пример: **returnUrl=https://test&audience=Directum.Omni**.
3. Обновите страницу.
4. В окне входа нажмите на кнопку **Войти с Google**. Открывается сайт Google для прохождения аутентификации:

   Если настройка корректна, после прохождения аутентификации открывается страница **https://test**.

Keycloak

1. Для аутентификации в Keycloak используется протокол OpenID Connect 1.0. Перед началом настройки убедитесь, что он доступен в используемой службе.
2. Настройте сервис Keycloak.
3. Настройте плагин провайдера аутентификации в сервисе идентификации.
4. Проверьте корректность настройки.

Настройка Keycloak

В инструкции приведен порядок настройки сервиса Keycloak на примере провайдера Cloud-IAM.

1. Перед началом настройки сохраните в удобной форме адрес сервиса идентификации. Пример: **https://id.contoso.ru**.

2. В консоли Keycloak создайте нового клиента. Для этого в разделе **Clients** на вкладке **Settings** заполните поля:
   - **Client ID** – ИД клиента. Придумайте значение, подходящее для организации. Пример: omniid;
   - **Valid redirect URLs** – адрес перенаправления на конечную точку signin-oidc в сервисе идентификации. Пример: **https://id.contoso.ru/signin-oidc**.
3. Включите использование **Client Secret**. Для этого в группе **Capability config**:
   - в поле **Client Authentication** установите значение **On**;
   - в группе **Authorization Flow** установите флажки **Standard flow** и **Implicit flow**:
4. С вкладки **Credentials** сохраните в удобной форме значения параметров **Client ID** и **Client Secret** для последующей настройки сервиса идентификации.

   Из раздела **Realm settings** с вкладки **General** из поля **Endpoints** сохраните в удобной форме адрес метаданных по ссылке **OpenID Endpoint Configuration**.

Настройка плагина провайдера аутентификации

В конфигурационный файл identity-service.env добавьте параметры:

- **Authentication__OpenId__MetadataAddress** – адрес для запроса метаданных сервиса Keycloak. Пример: **https://lemur-2.cloud-iam.com/auth/realms/omni-test/.well-known/openid-configuration**;
- **Authentication__OpenId__ClientId** – идентификатор клиентского приложения в Keycloak;
- **Authentication__OpenId__ClientSecret** – пароль клиентского приложения в Keycloak.

  Пример настройки:

```
Authentication__OpenId__MetadataAddress: 'https://lemur-2.cloud-iam.com/auth/realms/hrpro-test/.well-known/openid-configuration'
Authentication__OpenId__ClientId: 'omniid'
Authentication__OpenId__ClientSecret: '1234567890abcdefghijk'
```

- добавьте блок параметров для подключения плагина OpenIdAuthenticationProvider и заполните его как указано в примере.

  При необходимости в параметре **ChangePasswordPath** укажите путь к странице смены пароля пользователя на портале Keycloak.

  Пример настройки:

```
Authentication__Providers__0__Name: 'OpenIdAuthenticationProvider'
Authentication__Providers__0__AssemblyFileName:
'Directum.IdentityService.AuthenticationProviders.OpenIdAuthenticationProvider.dll'
Authentication__Providers__0__Type:
'Directum.IdentityService.AuthenticationProviders.OpenIdAuthenticationProvider.OpenIdAuthenticationProvider,
Directum.IdentityService.AuthenticationProviders.OpenIdAuthenticationProvider,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=427ba5252f628cb0'
Authentication__Providers__0__Audiences: 'Directum.Omni'
Authentication__Providers__0__Configuration__ChangePasswordPath: '<Адрес страницы смены пароля в keycloak>'
```

СОВЕТ. Чтобы настроить отображение провайдера, добавьте соответствующие настройки. Подробнее см. раздел «Настройка отображения внешних провайдеров аутентификации».

Проверка корректности настройки

1. Откройте сайт сервиса идентификации.
2. В конце адресной строки добавьте параметры **returnUrl** и **audience** и их значения. Пример: **returnUrl=https://test&audience=Directum.Omni**.
3. Обновите страницу.
4. В окне входа нажмите на кнопку **Войти с OpenID**. Открывается сайт Keycloak и прохождения аутентификации:

   Если настройка корректна, после прохождения аутентификации открывается страница **https://test**.

Создание учетной записи с заданной аутентификацией

В сервисе идентификации создайте учетные записи пользователей с установленной внешней аутентификацией. Учетная запись пользователя в сервисе идентификации создается с помощью Web API с использованием конечной точки /api/users. При создании в параметрах указываются используемый вид аутентификации и дополнительные реквизиты, например внешний логин. По ним учетная запись пользователя, прошедшего внешнюю аутентификацию, сопоставляется с данными в сервисе идентификации. Используемые для аутентификации реквизиты индивидуальны для каждого провайдера.

Чтобы создать учетную запись, выполните запрос к сервису. В запросе передайте вид аутентификации и реквизиты пользователя, которые сервис идентификации использует как внешний логин.

Пример запроса для создания пользователя с аутентификацией в AD FS. Внешний логин, выданный провайдером AD FS для аутентификации, указывается в параметре UPN.

```
POST /api/Users
{
  "name": "Petrov.IV@contoso.ru",
  "givenName": "Петров",
  "surname": "Иван",
  "patronym": "Васильевич",
  "phoneNumber": "8(900)8714123",
  "authentication": [
    {
      "provider": "adfs",
      "credentials": {
        "upn": "petrov_iv@contoso"
      }
    }
  ]
}
```

ПРИМЕЧАНИЕ. Если параметры аутентификации не заданы, то пользователь создается с типом аутентификации, который указан по умолчанию в параметре **DefaultAuthentication**.

Настройка отображения внешних провайдеров аутентификации

Если в конфигурационном файле сервиса идентификации заданы настройки плагина для провайдера внешней аутентификации, то для этого провайдера отображается:

- кнопка входа в личный кабинет с заголовком **Войти с <Название провайдера>**;
- кнопка изменения пароля с заголовком **Изменить пароль в <Название провайдера>**.

Для обеих кнопок используется стандартная иконка провайдера.

При необходимости отображение кнопок можно изменить. Для этого в конфигурационный файл identity-service.env добавьте параметры для настройки отображения провайдера:

- **Authentication__Providers__0__Configuration__Title** – название провайдера. Можно указать строковую константу или строки локализации;
- **Authentication__Providers__0__Configuration__SignInCaption** – заголовок для действия входа пользователя. Можно указать строковую константу или строки локализации;
- **Authentication__Providers__0__Configuration__ChangePasswordCaption** – заголовок действия смены пароля пользователя. Можно указать строковую константу или строки локализации;
- **Authentication__Providers__0__Configuration__Icon** – иконка провайдера. Можно указать ссылку на иконку или одно из значений для предопределенных иконок: **openid**, **windows**, **google** или **vk**.

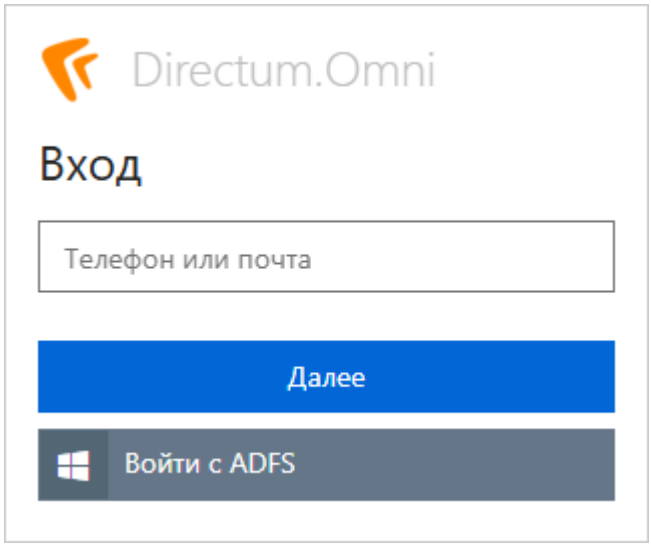
Пример настройки:

```
Authentication__Providers__0__Configuration__Title: 'ADFS-аутентификация'
Authentication__Providers__0__Configuration__SignInCaption: 'Войти под корпоративной учетной записью'
Authentication__Providers__0__Configuration__ChangePasswordCaption: 'Изменить пароль корпоративной учетной записи'
Authentication__Providers__0__Configuration__Icon: 'windows'
```

Вход пользователя с внешней аутентификацией

На странице входа в Directum Omni отображаются доступные типы аутентификации. Они определяются на основе информационного ресурса, который запросил аутентификацию. В настройках каждого провайдера аутентификации в параметре **Audiences** содержится список ресурсов, для которых применяется указанный вид аутентификации.

Для парольной аутентификации логин и пароль вводятся в окне входа в Directum Omni. Для внешней аутентификации отображаются кнопки входа со способами аутентификации:



Когда пользователь входит в Directum Omni с использованием внешней аутентификации, в браузере открывается страница используемого провайдера. Там указываются учетные данные для входа.

После входа открывается страница сервиса идентификации. Далее сервис:

1. Получает от внешнего провайдера токен для идентификации пользователя.
2. Определяет реквизит, который сопоставляется со списком внешних логинов в таблице сервиса.
3. Определяет, какая из учетных записей Directum Omni соответствует пользователю.
4. Сравнивает вид аутентификации пользователя с тем, который был использован при входе.
5. Аутентифицирует пользователя, если полученные данные совпадают.

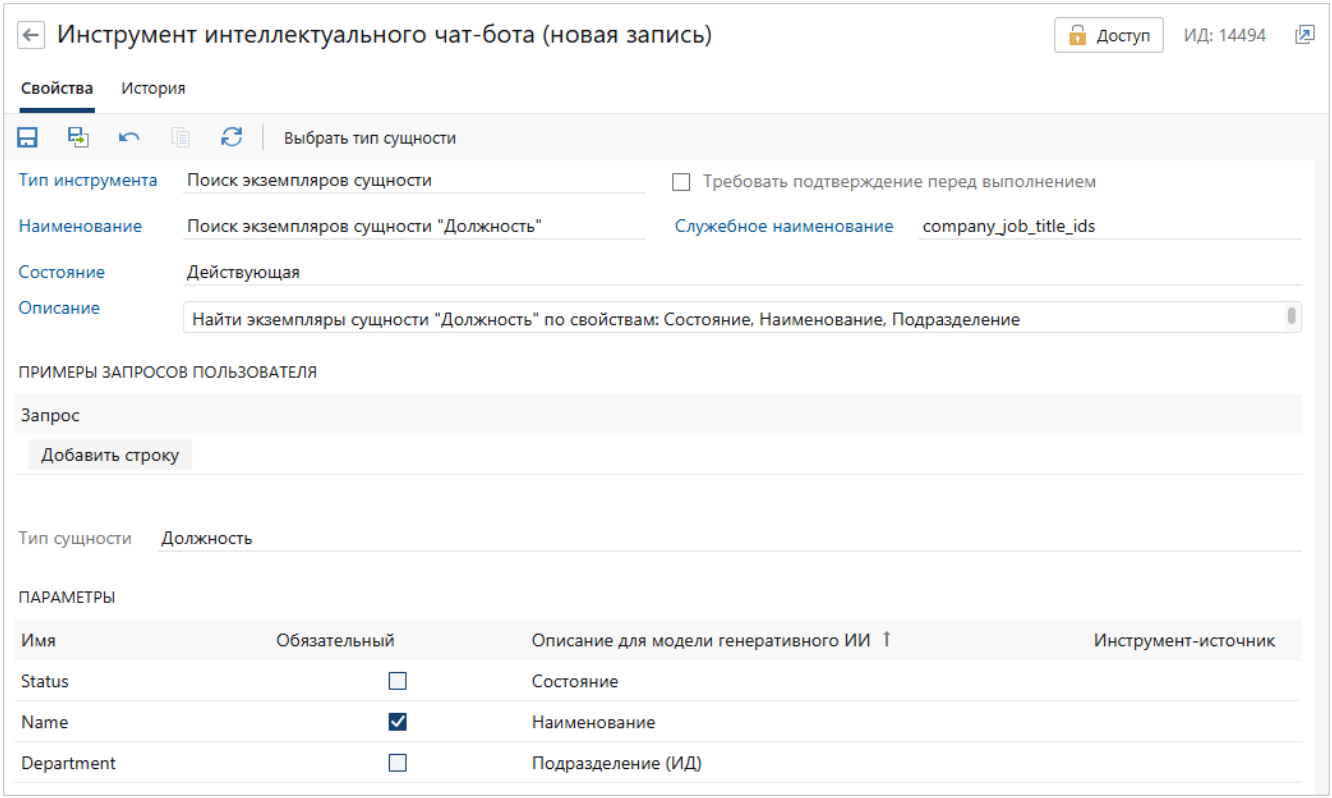
Настройка помощника Ario

Чтобы *помощник Ario* мог обрабатывать запросы пользователя, в Directum RX должны быть настроены специальные *инструменты*. Они задаются в справочнике **Инструменты интеллектуального чат-бота**.

Если установлено решение «Directum ESM», в системе по умолчанию доступны специальные инструменты. Если решение не установлено, инструменты нужно создать вручную. Для этого:

1. С помощью расширенного поиска перейдите к справочнику **Инструменты интеллектуального чат-бота**.
2. Нажмите на кнопку + Создать и в выпадающем списке выберите пункт **Инструмент интеллектуального чат-бота**.

   В результате открывается карточка инструмента:



3. Определите, какой тип действий должен выполнять инструмент, и в выпадающем списке **Тип инструмента** выберите значение:

   Поиск экземпляров сущности – инструмент находит нужную сущность в системе по информации из запроса и передает идентификатор сущности пользователю или другому инструменту;

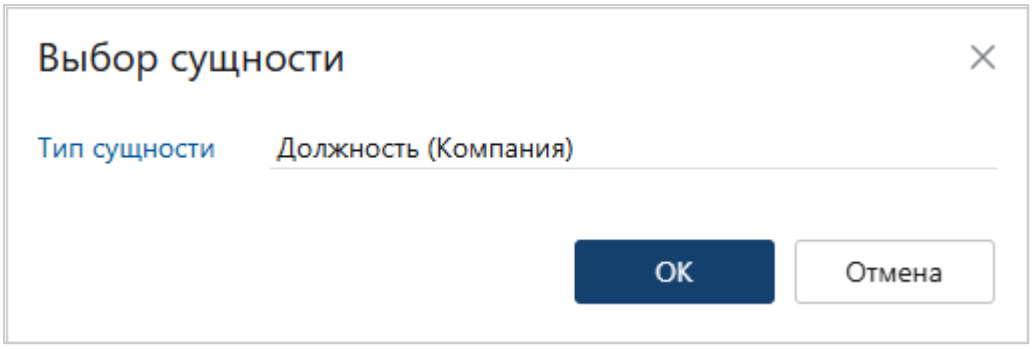
   Получение свойств сущности – инструмент находит нужную сущность в системе по идентификатору и передает информацию из свойств сущности пользователю или другому инструменту;

   Открытие карточки сущности – инструмент находит нужную сущность в системе по идентификатору и отображает карточку сущности в чате.

   ВАЖНО. Для работы инструментов получения свойств сущности или открытия ее карточки нужно предварительно создать инструмент, который ищет экземпляр сущности.

Также в среде разработки можно создать инструмент, который вызывает прикладные функции системы.

4. Выберите тип сущности системы, над которой выполняет действие инструмент. Для этого на панели действий нажмите на кнопку **Выбрать тип сущности** и в открывшемся окне укажите нужную запись:



Затем нажмите на кнопку **OK**.

5. Заполните другие поля карточки:

Требовать подтверждение перед выполнением. Признак того, что интеллектуальный помощник просит пользователя подтвердить выполнение действий инструмента. По умолчанию флажок снят и недоступен для редактирования.

ПРИМЕЧАНИЕ. Флажок может быть установлен для инструментов, которые вызывают прикладные функции. Разработчик настраивает это при создании инструмента.

***Наименование** инструмента. Поле заполняется автоматически при выборе типа инструмента и сущности в формате: <тип инструмента> <тип сущности>. Например, для инструмента, который получает свойства карточки валюты, в поле указывается значение Получение свойства сущности Валюта. При необходимости измените его вручную.

***Служебное наименование**. Уникальный идентификатор инструмента. Поле заполняется автоматически при выборе типа инструмента и сущности в формате:

<имя модуля>_<сущность>_<постфикс типа инструмента>

Например, для инструмента, который получает свойства карточки валюты, в поле указывается значение **commons_currency_info**. При необходимости измените его вручную.

ВАЖНО. Наименование должно быть написано латиницей без пробелов и не превышать 256 символов. Также можно использовать цифры и знаки подчеркивания.

***Описание** назначения инструмента. Оно передается в модель генеративного ИИ и используется при подборе нужного инструмента. Поле заполняется автоматически при выборе типа инструмента и сущности. При необходимости измените значение, учитывая рекомендации.

Примеры запросов пользователя для выполнения действий текущего инструмента. Примеры используются при предварительном подборе нужного инструмента. Чтобы добавить пример, в табличной части нажмите на кнопку **Добавить строку** и укажите значение. Например, для инструмента, который ищет контакты сотрудника, можно указать «Какой номер телефона у сотрудника?». При заполнении учитывайте рекомендации.

6. Табличная часть «Параметры» заполняется автоматически в зависимости от выбранных типа инструмента и сущности. В таблице отображаются свойства сущности,

которые требуются инструменту для выполнения действия. Для каждого свойства отображается основная информация о нем. При необходимости измените ее:

Имя свойства сущности.

***Описание для модели генеративного ИИ**. Описание свойства, по которому модель генеративного ИИ находит в запросе пользователя информацию о нужном свойстве сущности. При необходимости измените значение, учитывая рекомендации.

Обязательный. Установите флажок, если свойство нужно обязательно передавать для выполнения действия инструментом. Если в запросе такое свойство не указано, интеллектуальный помощник просит пользователя дополнить информацию о нем.

Инструмент-источник. Инструмент, который вызывается для получения параметра. Например, для инструмента, который получает информацию о сотруднике, нужно создать инструмент, который ищет идентификатор сотрудника по критериям, введенным пользователем в чате. Выберите нужный инструмент в выпадающем списке.

7. Сохраните карточку инструмента.

Системные инструменты продуктов

По умолчанию в Directum RX нет преднастроенных инструментов, но, если установлено решение «Directum ESM», в чат-боте с помощью интеллектуального помощника пользователь может создавать запросы, для которых созданы *услуги* в решении.

При создании услуги автоматически создаются инструменты для создания соответствующих запросов и инструкции по услугам. Подробнее об услуге см. в документации решения «Directum ESM».

Особенности инструментов по услуге:

- инструменты вызывают функции:
  - **GetInstructionPromptText** – формирует текст запроса к модели генеративного ИИ на подготовку инструкции по обращению пользователя. В текст запроса добавляется текст статей, которые связаны с услугой и используются для подготовки инструкции;
  - **CreateRequestFromTool** – создает карточку обращение по услуге.
- наименование инструмента и описание его назначения формируется из наименования услуги;
- на вход каждому инструменту передается идентификатор услуги;
- при изменении услуги соответствующие инструменты автоматически актуализируются.

Рекомендации по заполнению полей для модели генеративного ИИ

При подборе нужного *инструмента* для выполнения запроса пользователя в карточках инструментов модель генеративного ИИ учитывает значения:

- поля **Описание**;
- строк табличной части «Примеры запросов пользователей»;

- столбца **Описание для модели генеративного ИИ** в табличной части «Параметры».

Подробнее о полях см. в разделе «Настройка помощника Ario».

В разделе описаны рекомендации по их заполнению, чтобы *помощник Ario* выполнял запросы пользователя точнее.

Описание инструмента

1. Для инструментов поиска сущности или получения ее свойств указывайте, какие именно свойства нужно получить или используются для их поиска. Например:
   - вместо «Получение информации о сотруднике» укажите «Получение **телефона**, **почты** и **подразделения** сотрудника по **ИД** сотрудника»;
   - вместо «Получение идентификатора сотрудника» укажите «Получение **ИД** пользователя по **его фамилии**».
2. Если в компании используются синонимичные понятия или используются внутренние аббревиатуры, указывайте все варианты. Например:
   - «Оформление заявления для перевода в другой **отдел (подразделение)**»;
   - «Получение списка сотрудников **БЕ (бизнес-единицы)**».
3. Если инструмент имеет слишком общее описание, рекомендуется привести 1-2 примера его действия. Например:
   - вместо «Заявка на настройку оборудования на складе» укажите «Заявка на настройку оборудования на складе **(сигнализация, детекторы дыма, автоматические ворота)**»;
   - вместо «Заявка на подключение интернета» укажите «Заявка на подключение интернета **(установка роутера, прокладка кабеля, настройка wi-fi)**».

Примеры запросов пользователя

1. Иногда для выполнения действий основного инструмента требуется выполнение действий другого инструмента-источника. В этом случае примеры запросов нужно заполнять только для основного инструмента.

   Например, для получения информации о сотруднике нужно сначала найти идентификатор сотрудника по критериям, введенным пользователем в чате. В этом случае примеры запросов нужно заполнять только для инструмента, который ищет информацию о сотруднике.
2. В примерах запросов не нужно указывать определенных сущностей, например имен сотрудников или наименований организаций. Например, вместо «Какой телефон у Иванова Ивана» укажите «Какой телефон у **сотрудника**».
3. В одном примере запроса должно быть описано только одно действие или свойство. Например, вместо «Скажи телефон и почту сотрудника» укажите два примера запроса: «Скажи телефон сотрудника» и «Скажи почту **сотрудника**».

Описание параметра для модели генеративного ИИ

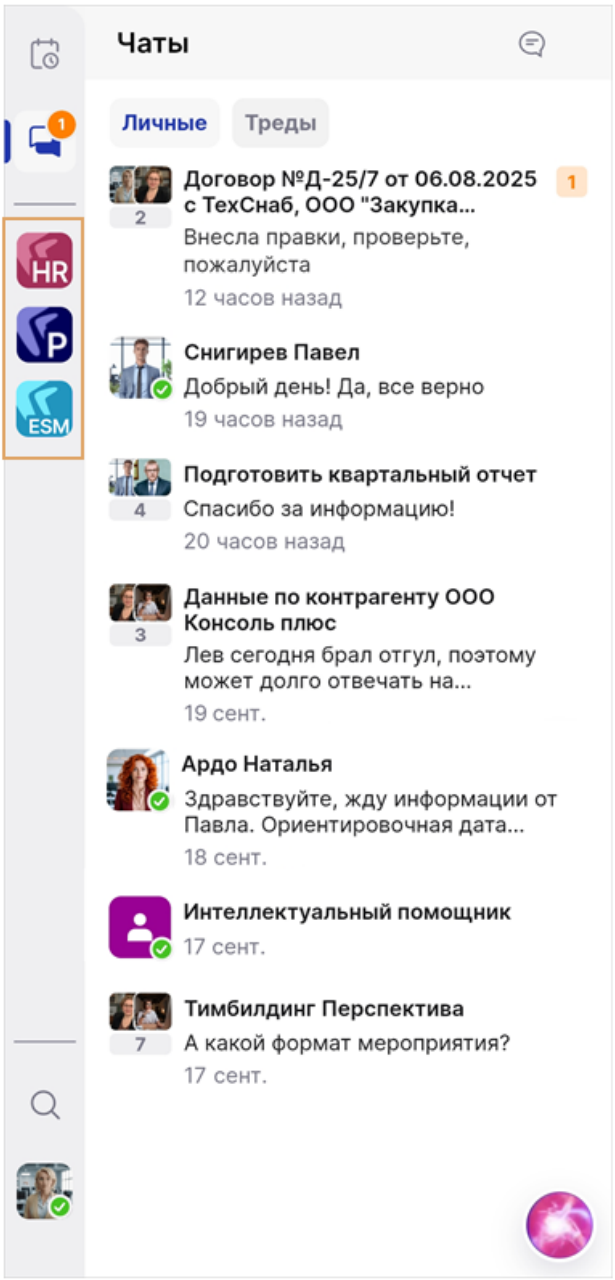
1. Учитывайте формат параметра, заданный в Directum RX. Если формат есть, то его нужно указать. Например:
   - «Дата оформления заявления **в формате YYYY-MM-DD**»;

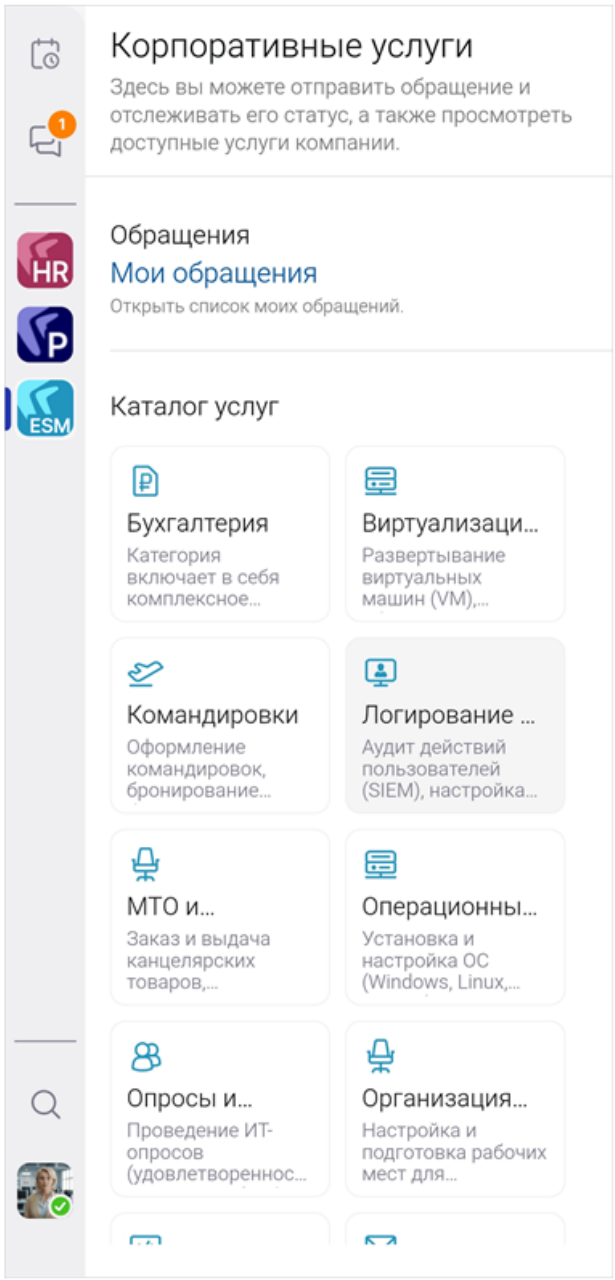
- «**Фамилия и имя** пользователя».

2. Если у параметра есть значение по умолчанию, указывайте его. Например:
   - «Дата оформления заявления в формате YYYY-MM-DD. **Если не указано, используй текущую дату**»;
   - «ИД заявителя. **Если не указан, используй ИД текущего пользователя**».
3. Если в параметре содержится текстовое поле и нужно избежать переписывания текста моделью генеративного ИИ, указывайте это в описании параметра. Например:
   - «**Полный текст** запроса пользователя»;
   - «**Полный текст** уведомления»;
4. Если нужно получить текст в определенном виде, указывайте это в описании параметра. Например, «Запрос пользователя от его лица **со словом «Прошу**».

Настройка миниаппов

В суперапп Directum Omni встраиваются *миниаппы* – с их помощью сотрудники получают доступ к сервисам компании прямо из приложения:





Настройки для работы миниаппов Directum HR Pro и «Корпоративный портал» задаются в процессе установки и настройки сервиса суперапп, а настройки no-code для работы миниаппа Directum ESM автоматически импортируются при установке решения. Эти миниаппы доступны сотрудникам, если есть лицензия на соответствующий продукт.

Если сотрудники регулярно используют другие сервисы и ресурсы, добавьте их в суперапп:

- чтобы добавить миниапп на базе цифровой платформы Directum RX, задайте соответствующие настройки no-code;
- чтобы подключить к Directum Omni сторонние веб-приложения, например портал для онлайн-обучения, укажите настройки в конфигурационном файле сервиса супераппа.

Настройка миниаппа на базе Directum RX

1. Создайте новое представление модуля или скопируйте существующее. В карточке представления заполните поля:
   - ***Имя** – укажите название миниаппа в Directum Omni;
   - **Состояние** – оставьте значение **Черновик**. Переведите представление в действующее состояние после окончания настройки;
   - **Контекст клиента** – выберите значение **Внешний клиент**;
   - ***Кто видит** и **Кроме** – укажите, кому должен быть доступен миниапп в Directum Omni;
   - **Описание** – при необходимости добавьте описание миниаппа, которое будет отображаться в Directum Omni.

   ПРИМЕЧАНИЕ. В карточке представления не нужно настраивать элементы модуля, так как они не отображаются в Directum Omni.

   Подробнее см. в книге «No-code: бизнес-процессы и интерфейс» раздел «Создание представления модуля».
2. Настройте обложку модуля. Обложка отображается при выборе миниаппа в Directum Omni. Добавьте на нее действия, которые сотрудники будут выполнять в приложении. При этом:
   - в настройках верхнего колонтитула в поле **Иконка** укажите иконку, которая будет использоваться в качестве значка миниаппа в Directum Omni;
   - учитывайте, что, если на обложку добавлено действие для формирования отчета или запуска диалога, оно не будет отображаться в Directum Omni;
   - используйте рекомендации по оформлению обложки. Обратите внимание, что при открытии миниаппа в мобильном приложении Directum Omni группы с действиями автоматически располагаются друг под другом.

   Подробнее см. в книге «No-code: бизнес-процессы и интерфейс» раздел «Настройка обложки».
3. Если необходимо, дополнительно настройте представления форм и списков, которые сотрудники будут открывать в миниаппе. Например, добавьте или скройте некоторые действия. При настройке в карточке представления формы или списка в поле **Контекст клиента** выберите значение **Внешний клиент**.

   Дополнительно учитывайте ограничения:
   - в супераппе доступно обновление списков и удаление ссылок из папок. Сортировка, фильтрация и выполнение прочих действий не поддерживается;
   - на формах карточек вместо табличных частей отображается сообщение «Не поддерживается».

   Подробнее см. в книге «No-code: бизнес-процессы и интерфейс» разделы «Настройка форм карточек» и «Настройка списков записей».
4. В карточке представления модуля в поле **Состояние** выберите значение **Действующая**.

В результате миниапп появляется в Directum Omni и становится доступен сотрудникам.

Подключение миниаппа через iframe

ВАЖНО. Веб-приложение, которое добавляется в суперапп, должно работать по протоколу HTTPS. Также в приложении должны быть корректно заполнены заголовки X-Frame-Options и Content Security Policy для встраивания через iframe. Подробнее см. на сайте MDN Web Docs разделы X-Frame-Options header и Content Security Policy (CSP).

Чтобы подключить веб-приложение к супераппу через iframe:

1. Откройте конфигурационный файл сервиса супераппа saservice.env.
2. В конфигурационный файл добавьте настройки миниаппа в следующем формате:

```
Applets__<Порядковый номер элемента в массиве Applets>__<Имя параметра>: "<Значение параметра>"
```

ВАЖНО. Порядковый номер указывается в соответствии с другими блоками настроек Applets, добавленными в конфигурационный файл. Например, если в файле уже есть блоки настроек Applets__0 и Applets__1, для добавляемого миниаппа укажите следующий порядковый номер Applets__2. Убедитесь, что номер уникален в рамках конфигурационного файла. Далее приведены примеры названия параметров с порядковым номером 2.

Укажите значения параметров:

- **Applets__2__Name** – имя миниаппа, которое используется для формирования URL. Имя должно быть кратким, состоять из латинских букв и не содержать пробелов;
- **Applets__2__Title** – отображаемое название миниаппа в Directum Omni;
- **Applets__2__Description** – отображаемое описание миниаппа в Directum Omni;
- **Applets__2__Mount** – признак добавления миниаппа в панель доступа Directum Omni. Укажите значение **true**;
- **Applets__2__MountGroup** – группа элементов, в которую добавляется миниапп. Укажите значение **MiniApps**;
- **Applets__2__IconUrl** – путь до значка миниаппа в формате SVG, PNG или JPG. При отображении в супераппе значок автоматически подгоняется к размерам 36x36 пикселей;
- **Applets__2__WebPartUrl** – способ встраивания миниаппа. Укажите значение **/webparts/embed**;
- **Applets__2__Config__requestMethod** – метод запроса контента встраиваемого веб-приложения. Укажите значение **GET**;
- **Applets__2__Config__resourceUrl** – URL-адрес встраиваемого веб-приложения;
- **Applets__2__Config__defaultRoute** – дополнительные параметры URL. Необязательный параметр. Например, его можно использовать, чтобы при открытии миниаппа сотрудник сразу попадал на определенный раздел веб-сайта;
- **Applets__2__Enabled** – признак отображения миниаппа в Directum Omni. Чтобы миниапп был доступен сотрудникам, укажите значение **true**, иначе укажите значение **false**.

Пример настройки:

```
Applets__2__Name: "learn"
Applets__2__Title: "Курсы обучения"
Applets__2__Description: "Онлайн-курсы и запись на обучение"
Applets__2__Mount: "true"
```

```
Applets__2__MountGroup: "MiniApps"
Applets__2__IconUrl: "https://images.company.ru/logo_32.svg"
Applets__2__WebPartUrl: "/webparts/embed"
Applets__2__Config__requestMethod: "GET"
Applets__2__Config__resourceUrl: "https://learn.company.ru"
Applets__2__Config__defaultRoute: "/courses/favorites"
Applets__2__Enabled: "true"
```

3. Перезапустите сервис супераппа.

В результате миниапп появляется в Directum Omni и становится доступен сотрудникам.

Логирование Directum Omni

Сообщения об ошибках, предупреждения и информационные сообщения, которые появляются во время работы Directum Omni, записываются в логи. Администратор системы или специалист службы безопасности использует их для отслеживания состояния приложения, устранения ошибок и расследования инцидентов.

Directum RX

Логи сервисов и компонентов Directum RX записываются в файлы с расширением *.log в формате JSON. Подробнее о расположении файлов и рекомендациях по работе с ними см. в руководстве администратора Directum RX в разделе «Лог-файлы во время работы».

Сервисы Directum Omni

Для сервисов Directum Omni записываются:

- основные логи. По умолчанию записываются в файлы с расширением *.log в формате JSON;
- журнал событий аудита. Записывается в файлы с расширением *.json;
- профайлинг входящих запросов. Записывается в файлы с расширением *.json;
- журнал библиотеки NLog. Записывается в файлы <Название компонента>.logger.log.

Лог-файлы каждого сервиса располагаются в папке, указанной в конфигурационном файле docker-compose.yml в секции этого сервиса в параметре **volumes**. Например, /opt/omni/logs. При необходимости укажите другую папку, при этом путь до папки с логами внутри контейнера (/app/Logs) оставьте неизменным:

```
volumes:
  - /opt/omni/certificates/:/certificates:ro
  - /opt/logs:/app/Logs
```

Информация в записях лога для удобства чтения разделена на атрибуты. Перечень и описание атрибутов см. в описаниях типов логов.

При необходимости для каждого сервиса можно изменить настройки логирования.

Клиентские приложения Directum Omni

В лог-файлы записываются ошибки клиентских приложений и статистика времени выполнения операций. Лог-файлы автоматически передаются в BFF-сервер (SuperAppBFF). Они ведутся в формате JSON и имеют расширение .*log.

Лог-файлы располагаются в папке с другими логами BFF-сервера и именуются по формату:

<IP-адрес устройства пользователя>.SuperApp.<Логин учетной записи пользователя в сервисе идентификации>.<Дата>.log

Основные логи

В зависимости от настроек компонента основные логи могут записываться в файл, в индекс Elasticsearch или в консоль сервера. Также в настройках задается:

- формат названий для лог-файлов или индекса;
- уровень логирования;
- логирование текущих настроек сервиса. По умолчанию логирование настроек включено.

Подробнее см. в разделе «Настройка логирования».

Структура лога

Атрибут	Описание атрибута
t	Дата и время события в часовом поясе сервера
l	Уровень логирования
lg	Имя логгера, с помощью которого добавлена запись
mt	Шаблон сообщения. Передается в формате строки или в структурированном формате JSON. Структурированный формат выделяется символами { и }
ex	Исключение. По сравнению с mt содержит более детальную информацию. Стандартные атрибуты: • type – тип исключения; • m – сообщение об ошибке; • stack – стек вызовов. Содержит информацию о последовательности вызова функций и помогает найти изначальную функцию, в которой произошла ошибка
pid	Идентификатор процесса и потока
un	Идентификатор пользователя
ur	Роль пользователя
cip	IP-адрес клиента
tr	Идентификатор трассировки
tn	Идентификатор тенанта
h	Имя хоста сервера
s	Имя компонента, для которого записывается лог
v	Версия компонента

Журнал событий аудита

События аудита записываются в лог-файл, имя которого формируется по шаблону:

<Имя хоста>.<Имя сервиса>.Audit.<Дата и время>.json

Структура лог-файла

Атрибут	Описание атрибута
timestamp	Дата и время события
event	Информация о событии аудита в виде массива в формате JSON
Action	Информация о действии, в результате которого произошло событие, в виде массива в формате JSON
TraceId	Идентификатор трассы
HttpMethod	Метод HTTP-запроса
ControllerName	Название контроллера, к которому обращается запрос
ActionName	Имя действия
ActionParameters	Параметры действия
RequestUrl	Адрес, по которому отправлен запрос к компоненту
IpAddress	IP-адрес клиента
ResponseStatus	Состояние HTTP-запроса
ResponseStatusCode	Код состояния HTTP-запроса
RequestBody	Тело запроса
EventType	Тип события
Environment	Сведения об окружении, в котором произошло событие, в виде массива в формате JSON
UserName	Имя пользователя, от имени которого отправлен запрос
MachineName	Имя физического сервера или виртуальной машины, где произошло событие
DomainName	Имя домена, где произошло событие
CallingMethodName	Имя вызываемого метода
AssemblyName	Название сборки
Culture	Культура, заданная для компонента
StartDate	Дата и время начала выполнения запроса с указанием часового пояса
EndDate	Дата и время окончания выполнения запроса с указанием часового пояса
Duration	Время выполнения запроса в мс
EventId	ИД события
ip	IP-адрес клиента

Профайлинг входящих запросов

Если для сервиса включен профайлинг, в отдельный лог-файл записываются данные обо всех входящих запросах к сервису. Имя лог-файла формируется по шаблону:

<Имя хоста>.<Имя сервиса>.Profile.<Дата и время>.json

Структура лога

Атрибут	Описание атрибута
timestamp	Дата и время события
requestId	Идентификатор запроса
requestMethod	Метод HTTP-запроса
requestUrl	Адрес, по которому отправлен запрос к компоненту
responseStatus	Код состояния HTTP-запроса
duration	Время выполнения запроса в мс

Настройка логирования

Настройки задаются индивидуально для каждого сервиса в его конфигурационном файле.

В логи записываются данные:

Данные	По умолчанию	Настройка
Журнал событий сервиса (основные логи)	Включено	Для основных логов можно настроить: • формат записи; • уровни логирования в параметре **Logging__LogLevel__Default**; • дополнительные параметры в формате Logging__<Название свойства>__<Название категории>: "Значение" Подробнее см. в документации Microsoft в статье «Ведение журнала в .NET Core и ASP.NET Core»
Журнал событий аудита	Включено	Чтобы отключить, укажите значение **false** в параметре **Diagnostics__EnableAuditLogging**. Для журнала можно настроить маскировку конфиденциальных данных в параметре **Security__MaskSensitiveDataInAuditLog**
Текущие настройки сервиса	Включено	Записываются в основные логи сервиса. Чтобы отключить, укажите значение **false** в параметре **Diagnostics__EnableSettingsLogging**
Профайлинг всех входящих запросов	Отключено	Содержит запрос, код ответа и время выполнения. Чтобы включить, укажите значение **true** в параметре **Diagnostics__EnableRequestProfiling**
Расширенные данные сервиса	Отключено	Содержат исходящие HTTP-запросы, дополнительную информацию о подписании и т.д. Записываются в основные логи сервиса. Чтобы включить, укажите значение **true** в параметре **Diagnostics__EnableExtendedLogOutput**. ВАЖНО. Для записи расширенных данных должен быть установлен уровень логирования **Information** или ниже

Формат записи логов

Настройки задаются индивидуально для каждого сервиса в его конфигурационном файле.

Для основных логов можно изменить формат записи. Для этого добавьте параметр **Diagnostics__LogOutputs** и укажите в нем одно или несколько значений:

- **File** – текстовый файл. Дополнительно укажите значения параметров:
  - **Diagnostics__FileLogOutput__File** – имя для файла логов. В качестве имени можно использовать шаблоны. Полный список доступных шаблонов см. в документации библиотеки NLog. Доступные расширения файлов: *.log, *.json, *.txt, *.doc.

    Значение по умолчанию ${hostname}.SuperApp.All.${shortdate:universalTime=true}.log;
  - **Diagnostics__FileLogOutput__Format** – формат лог-файлов. Возможные значения: **Text**, **Json**. Значение по умолчанию **Text**;
- **ElasticSearch** – поисковая система Elasticsearch. Дополнительно укажите значения параметров:
  - **Diagnostics__ElasticSearchLogOutput__ServiceAddress** – адрес сервиса Elasticsearch;
  - **Diagnostics__ElasticSearchLogOutput__Index** – имя индекса для лога. Можно использовать шаблоны, например logs-${date:format=yyyy.MM.dd}. Все доступные шаблоны см. в документации библиотеки NLog;
  - **Diagnostics__ElasticSearchLogOutput__ApiKeyId** – идентификатор API-ключа;
  - **Diagnostics__ElasticSearchLogOutput__ApiKey** – API-ключ;
- **Console** – консоль сервера;
- **Null** – не сохранять логи.

Пример настройки:

```
Diagnostics__LogOutputs: "ElasticSearch, File"
Diagnostics__ElasticSearchLogOutput__ServiceAddress: "http://contoso.ru/elasticsearch:6800"
Diagnostics__ElasticSearchLogOutput__Index: "logs${date:format=yyyy.MM.dd}"
Diagnostics__ElasticSearchLogOutput__ApiKeyId: "L41sa4cBihGiY8agV6UM"
Diagnostics__ElasticSearchLogOutput__ApiKey: "cl08RGc3QfGt4rygaYmXog"
Diagnostics__FileLogOutput__Directory: "Logs"
Diagnostics__FileLogOutput__File:
"${hostname}.SuperApp.All.${shortdate:universalTime=true}.log"
Diagnostics__FileLogOutput__Format: "Json"
```

Уровень логирования

Настройки задаются индивидуально для каждого сервиса в его конфигурационном файле.

Параметры логирования в конфигурационном файле задаются в формате:
Logging__<Название свойства>__<Название категории>: "Значение"

Минимальный уровень логирования для основных лог-файлов задается в формате:
Logging__LogLevel__Default: "Значение"

Возможные значения:

- **Trace** – записываются все сообщения, включая сообщения трассировки выполнения запросов;

- **Debug** – записываются все сообщения, включая отладочные, кроме сообщений трассировки;
- **Information** – записываются все сообщения, включая информационные, кроме сообщений трассировки и отладочных;
- **Warning** – записываются только предупреждения и ошибки;
- **Error** – записываются только ошибки;
- **Critical** – записываются только критичные ошибки;
- **None** – запись отключена.

По умолчанию в лог-файл записываются сообщения с уровнем Information и выше: Warning, Error, Critical.

Если необходимо, чтобы в лог-файл записывались сообщения с уровнем не ниже Warning, в параметре **Logging__LogLevel__Default** укажите значение **Warning**:

```
environment:
  Logging__LogLevel__Default: "Warning"
```

Маскировка конфиденциальных данных в журнале событий аудита сервиса идентификации

Настройки задаются в конфигурационном файле сервиса идентификации identity-service.env.

В журналы событий аудита включается информация о пользователях: ФИО, номер телефона и адрес электронной почты. Чтобы защитить такие данные, их можно маскировать при записи в журнал. Для этого в конфигурационный файл добавьте параметр **Security__MaskSensitiveDataInAuditLog** и укажите в нем значение **true**. В результате при записи конфиденциальных данных в лог часть символов заменяются на знаки *****:

- в фамилии, имени и отчестве отображаются первые буквы;
- в номере телефона – первые и последние три цифры;
- в адресе электронной почты – первый символ и домен.

Пример записи данных в журнале событий аудита:

```
"Person": {
    "GivenName":"A***",
    "Surname":"H******",
    "Patronym":"A*********",
    "Email":"a******@contoso.ru",
    "Phone":"8999****123"
}
```

По умолчанию маскировка отключена.