Н·МЕДИАТОР

медиатор бизнеса

информационные системы и
электронные компоненты

Разворачивание платформы Сигма 4.0

Версия: 1.0

Содержание

1. Назначение документа

Данный документ предназначен для системного администратора/администратора по программному обеспечению (DevOps), перед которым поставлена задача развернуть платформу на ресурсах организации самостоятельно при помощи поставляемых инструментов для ознакомительных целей, обучения, разработки модулей или небольшой организации. Для промышленного разворачивания платформы требуется, на основании ниже изложенной информации, потребностей организации, особенностей аппаратного обеспечения, требований безопасности, нагрузки, произвести настройку среды исполнения разработав свой сценарий, в том числе учитывая механизмы динамического масштабирования с использованием, например, Docker SWARM или Kubernetes.

2. Среда исполнения

Для более точного определения потребности в программных и аппаратных средствах, требуемых для запуска платформы, рекомендуется ознакомиться с документом «Эксплуатационная документация: технические требования».

В качестве среды исполнения предполагается использование Debian-подобной операционной системы, как-то: Astra, Ubuntu, Debian, RedHat и т.д. В случае использования других UNIX-подобных операционных систем может потребоваться работа с установочными скриптами.

Технические требования:

Параметр	Значение	Примечание
ОЗУ	16Гб	
Диск	45Гб	
Процессор	4 ядра	9000 passmark единиц или лучше

Для данного типа развертывания предполагается работа всех компонент на одной машине.

3. Процедура разворачивания

Для начала работы, вы должны получить доступ к репозиторию Docker-объектов – логин и пароль, которые позволят загружать последние версии модулей платформы.

Скачайте установочный скрипт и запустите его. Убедитесь, что ваша учетная запись входит в группу sudo.

В процессе установки будут загружены компоненты из репозитория, установлены программы, будет создана учетная запись user. В случае, если учетная запись user существует, добавьте ее в группу sudo.

Внимание: установка происходит на ваш риск, никак не гарантирует результат и никак не гарантирует сохранность ваших данных. Может быть использована в целях, описанных в данном документе в разделе №1.

3.1. Получение установочного скрипта

Скачайте из репозитория установочный скрипт, установите разрешения на запуск и выполните скрипт. Внимание: работоспособность скрипта проверена на Ubuntu Ubuntu 24.04.1 LTS. Процесс установки и работоспособность скрипта может отличаться для других версий или типов ОС, возможно потребуется корректировка.

```
cd ~
wget https://nexus.nmediator.ru/repository/miscelaneous/sigma/install/install.sh
chmod 755 install.sh
./install.sh repouser repopassword
```

3.2. Установка

В процессе установки будут запрошены пароли от учетной записи. На машину будет установлен nginx, созданы сертификаты, статические компоненты Angular, создан сервис, созданы базы данных и наполнены тестовыми данными. После успешной установки сервис запуска будет автоматически запускать платформу. Ручной запуск и останов может быть выполнен следующим образом:

```
sudo systemctl stop sigma.service
sudo systemctl start sigma.service
```

3.3. Установка разрешений

В процессе установки скрипт будет анализировать IP адрес машины, на которую устанавливается платформа, причем префикс адреса зафиксирован на 192…. Если требуется другой адрес, то требуется корректировка установочного скрипта – задание фиксированного адреса через переменную IPADDRESS или установка маски в выражении `grep -m1 '^192.*$'` основного скрипта. На основании адреса будет настроен прокси сервер nginx. На изображении ниже определен адрес 192.168.110.197.

```
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
Generating locales (this might take a while)...
  ru_RU.ISO-8859-5... done
Generation complete.
Generating locales (this might take a while)...
  ru_RU.UTF-8... done
Generation complete.
Для установки будет использован адрес 192.168.110.197 в качестве внешнего адреса сетевого адаптера.
Если это неверно, то определине адрес в переменной IPADDRESS
Запускаю основную инсталляцию. Введите пароль для user
```

На компьютере, на котором будет осуществляться работа с платформой, требуется принудительно установить доверие для адресов https://192.168.110.197 и https://192.168.110.197:444, установив исключения для проверки сертификатов (желательно открыть в браузере оба адреса и проверить, что исключения работают)

4. Проверка

4.1. Открыть портал платформы

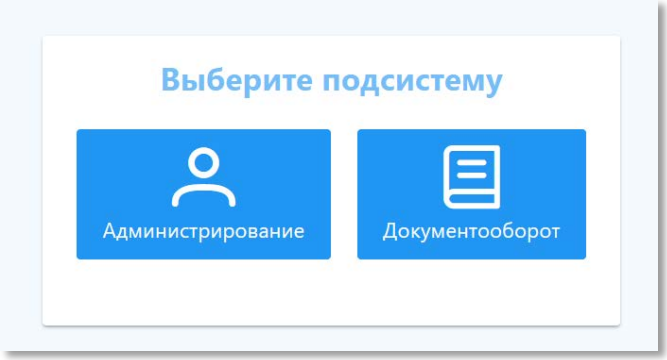
Для этого в браузере наберите адрес машины, на которую была произведена установка, например: https://192.168.110.197

4.2. Выполнить вход

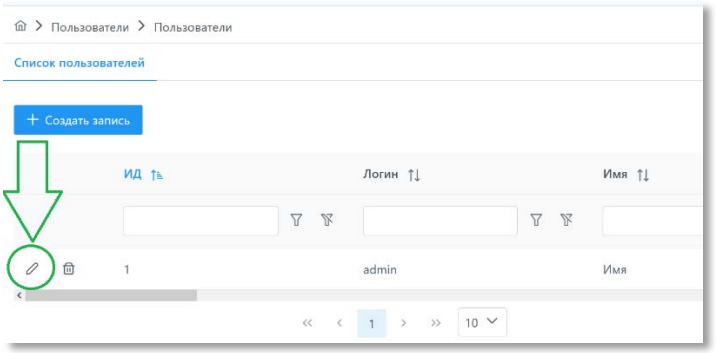
В появившейся форме ввода логина и пароля ввести admin и password.

4.3. Добавить права на работу с модулем НСИ

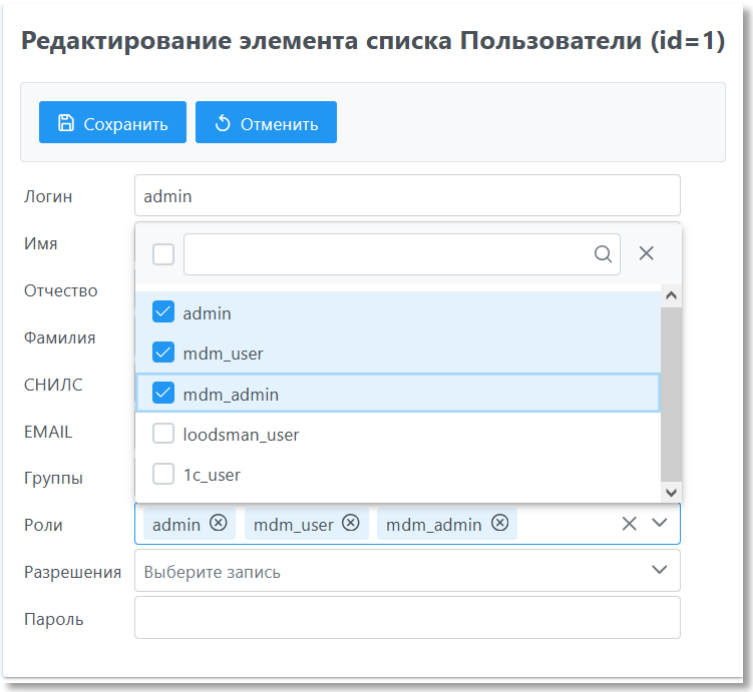
4.3.1. Нажать на вкладку «Администрирование»

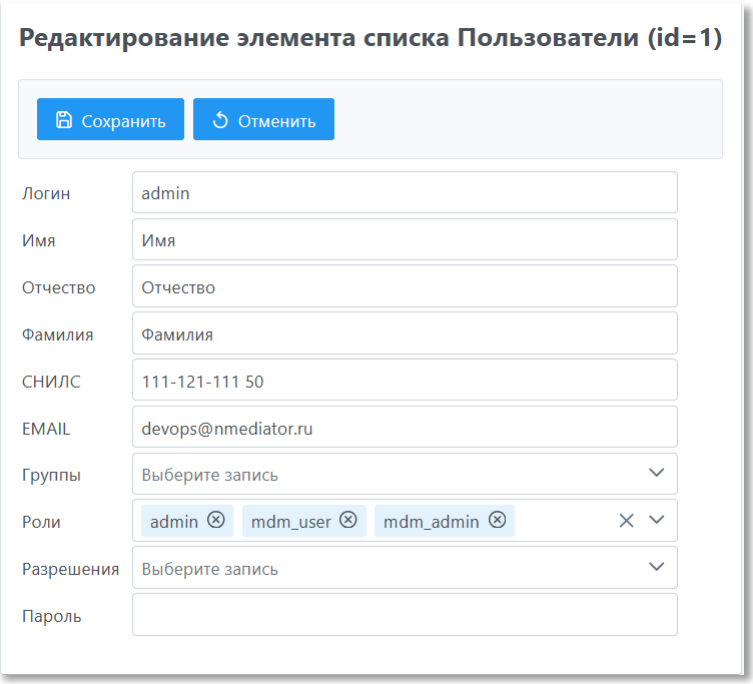


4.3.2. Выбрать пользователя admin и нажать на кнопку редактирования в виде карандаша



4.3.3. Добавить роли пользователя и администратора НСИ





4.3.4. Сохранить изменения

Далее провести работу в соответствии с документацией.

5. Содержание установочного скрипта

Основной:

```
#!/bin/bash

REPONAME=nexus.nmediator.ru
USERNAME=user
NEXUSLOGIN=
NEXUSPASSWORD=
IPADDRESS=
set -e

echo "Скрипт для установки SIGMA. Нажмите любую клавишу для
продолжения, для прекращения нажмите Ctrl-C"
read -s -N 1
if [ "$#" -ne 2 ]; then
    echo "Использование: install.sh <repologin> <repopassword>"
    exit 1;
fi

sudo apt-get -y install net-tools locales
sudo locale-gen ru_RU
sudo locale-gen ru_RU.UTF-8
if [ "" == "$IPADDRESS" ]; then
IPADDRESS=$(ifconfig | grep -Eo 'inet (addr:)?([0-9]*\.){3}[0-
9]*' | grep -Eo '([0-9]*\.){3}[0-9]*' | grep -v '127.0.0.1' |
grep -m1 '^192.*$')
echo "Для установки будет использован адрес $IPADDRESS в
качестве внешнего адреса сетевого адаптера."
echo "Если это неверно, то определине адрес в переменной
IPADDRESS. Нажмите любую клавишу для продолжения, для
прекращения нажмите Ctrl-C"
read -s -N 1
fi

NEXUSLOGIN=$1
NEXUSPASSWORD=$2

if [ "1" == $(id -u $USERNAME >/dev/null 2>&1; echo $?) ]; then
echo Создаю пользователя $USERNAME
sudo useradd -m -s /bin/bash -d /home/$USERNAME $USERNAME
sudo usermod -aG sudo $USERNAME
echo Введите новый пароль для $USERNAME
sudo passwd $USERNAME
fi

sudo wget -q
https://nexus.nmediator.ru/repository/miscelaneous/sigma/install
/dowork.sh -O /home/$USERNAME/dowork.sh
sudo chown $USERNAME:$USERNAME /home/$USERNAME/dowork.sh
sudo chmod 777 /home/$USERNAME/dowork.sh
export REPONAME
export USERNAME
export NEXUSLOGIN
export NEXUSPASSWORD
export IPADDRESS
echo Запускаю основную инсталляцию. Введите пароль для $USERNAME
```

```
su $USERNAME -m -c /home/$USERNAME/dowork.sh

exit 0
```

Вспомогательный:

```
#!/bin/bash

#Qs? Ask info@nmediator.ru

#
if [ ! -d "/IO" ]; then
sudo -S mkdir /IO
fi

#################################################################
###############
#docker
sudo -S apt-get -y update && sudo -S apt-get -y upgrade && sudo
-S apt-get -y install git maven wget curl rar unzip maven
default-jdk net-tools openssl
for pkg in docker.io docker-doc docker-compose docker-compose-v2
podman-docker containerd runc; do sudo -S apt-get -y remove
$pkg; done;
sudo -S apt-get -y install ca-certificates curl
sudo -S install -m 0755 -d /etc/apt/keyrings
sudo -S curl -fsSL https://download.docker.com/linux/ubuntu/gpg
-o /etc/apt/keyrings/docker.asc
sudo -S chmod a+r /etc/apt/keyrings/docker.asc
echo \
  "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo -S tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo -S apt-get -y update
sudo -S apt-get -y install docker-ce docker-ce-cli containerd.io
docker-buildx-plugin docker-compose-plugin
sudo -S usermod -aG docker $USERNAME
sudo -S systemctl restart docker.service
sudo -S systemctl restart docker.socket
# will be fixed after reboot
sudo -S chmod o+rw /var/run/docker.sock
docker login -u ${NEXUSLOGIN} -p ${NEXUSPASSWORD} $REPONAME
#docker
#################################################################
###############

#################################################################
###############
#nginx
sudo -S apt-get -y install nginx
sudo -S rm /etc/nginx/sites-enabled/*
if [ ! -d "/etc/nginx/certs" ]; then
sudo -S mkdir /etc/nginx/certs
fi
sudo -S rm /etc/nginx/certs/*
```

```
cd /home/$USERNAME

read -r -d '' VAR << EOF
basicConstraints = critical,CA:FALSE\n
subjectKeyIdentifier = hash\n
keyUsage = critical, digitalSignature, keyEncipherment\n
extendedKeyUsage = critical, serverAuth, clientAuth\n
subjectAltName = DNS:$IPADDRESS, DNS:$IPADDRESS:444\n
EOF

echo -e "$VAR" >/home/$USERNAME/v3.ext

sudo -S openssl req -x509 -nodes -days 500 -newkey rsa:2048 \
-keyout /etc/nginx/certs/server_private.key \
-out /etc/nginx/certs/server_public.cer \
-subj
"/C=RU/ST=Capital/L=Moscow/O=NMediator/OU=Dev/CN=NMediator" \
-addext
"keyUsage=digitalSignature,keyEncipherment,dataEncipherment,cRLS
ign,keyCertSign" \
-addext "basicConstraints=critical,CA:TRUE,pathlen:1"

sudo -S openssl genrsa -out /etc/nginx/certs/client_private.key
2048

sudo -S openssl req -new -nodes \
-key /etc/nginx/certs/client_private.key \
-out /etc/nginx/certs/client.csr \
-subj "/C=RU/ST=SPB/L=SPB/O=NMediator/OU=Dev/CN=$IPADDRESS/" \
-addext "subjectAltName = DNS:$IPADDRESS, DNS:$IPADDRESS:81" \
-addext "extendedKeyUsage=critical,serverAuth,clientAuth" \
-addext "basicConstraints=CA:FALSE"

sudo -S openssl x509 -req \
-in /etc/nginx/certs/client.csr \
-CA /etc/nginx/certs/server_public.cer \
-CAkey /etc/nginx/certs/server_private.key \
-CAcreateserial \
-out /etc/nginx/certs/client_public.cer \
-days 500 -sha256 \
-extfile v3.ext

sudo -S bash -c "cat /etc/nginx/certs/client_public.cer >
/etc/nginx/certs/public.cer"
sudo -S bash -c "cat /etc/nginx/certs/server_public.cer >>
/etc/nginx/certs/public.cer"
sudo -S bash -c "cp /etc/nginx/certs/client_private.key
/etc/nginx/certs/private.key"

sudo chmod 700 /etc/nginx/certs


read -r -d '' VAR <<-'EOF'
server {
    listen            443 ssl;
    ssl_certificate        /etc/nginx/certs/public.cer;
    ssl_certificate_key    /etc/nginx/certs/private.key;
```

```
    location ~ ^/mdm-ui(/*$) {
      rewrite                    ^(.*)$ http://\$host/mdm-ui/ui
permanent;
    }
    location ~ ^/sigma-ui(/*$) {
      rewrite                    ^(.*)$ http://\$host/sigma-
ui/ui permanent;
    }

    location ~ ^/mdm-ui(.*)$ {
      client_max_body_size            500M;
      proxy_buffering       off;
      proxy_redirect                  off;
      proxy_read_timeout               10m;
      proxy_connect_timeout           1m;
      proxy_set_header                 X-Forwarded-For
\$proxy_add_x_forwarded_for;
      proxy_set_header      X-Forwarded-Host \$host;
      proxy_set_header                 Host \$host;
      proxy_set_header                 X-Forwarded-Proto
\$scheme;
      proxy_pass                       http://127.0.0.1:8089/mdm-
ui\$1\$is_args\$args;
    }

    location ~ ^/sigma-ui(.*)$ {
      client_max_body_size            500M;
      proxy_buffering            off;
      proxy_redirect                  off;
      proxy_read_timeout               10m;
      proxy_connect_timeout           1m;
      proxy_set_header                 X-Forwarded-For
\$proxy_add_x_forwarded_for;
      proxy_set_header      X-Forwarded-Host \$host;
      proxy_set_header                 Host \$host;
      proxy_set_header                 X-Forwarded-Proto
\$scheme;
      proxy_pass
      http://127.0.0.1:8090/sigma-ui\$1\$is_args\$args;
    }


    root /IO/sigma-static;
    index index.html;

    location ~ ^(\/|^\/\\\?|^\\\?)(((?!\.).)+?)(\\\?|$)(.*?)$
    {
      proxy_buffering off;
      autoindex  on;
      set \$spath \$2;
      if (\$args ~ (.*)(?:&|^)path=[^&]*(.*))
      {
            set \$args \$1\$2;
        }
      if (\$is_args)
      {
```

```
          set \$args \$args&path=\$spath;
      }
      if (\$is_args = \"\")
      {
          set \$args path=\$spath;
      }

      if (\$request_uri !~ '^(\/|(.+\..+))$')
      {
          rewrite ^(.*)$ http://\$host/ permanent;
      }
    }
}
server {
    listen 80;
    return 301 https://\$host\$request_uri;
    }
EOF

sudo -S bash -c "printf \"$VAR\" > /etc/nginx/sites-
available/sigma"

read -r -d '' VAR <<-'EOF'
server {
    listen                  444 ssl;
    ssl_certificate             /etc/nginx/certs/public.cer;
    ssl_certificate_key         /etc/nginx/certs/private.key;

    location ~ ^/(.*) {
    proxy_redirect              off;
    proxy_read_timeout                  10m;
    proxy_send_timeout                  10m;
    fastcgi_send_timeout        10m;
    fastcgi_read_timeout        10m;
    proxy_connect_timeout       1m;
    keepalive_timeout           10m;
    proxy_set_header            X-Forwarded-For
\$proxy_add_x_forwarded_for;
    proxy_set_header            Host \$http_host;
    proxy_set_header            X-Forwarded-Proto \$scheme;
    proxy_pass
      http://127.0.0.1:8081/\$1\$is_args\$args;
    client_max_body_size        500M;
    proxy_buffering             off;
    }
}
EOF

sudo -S bash -c "printf \"$VAR\" > /etc/nginx/sites-
available/gateway"

sudo -S ln -s /etc/nginx/sites-available/gateway
/etc/nginx/sites-enabled/gateway
sudo -S ln -s /etc/nginx/sites-available/sigma /etc/nginx/sites-
enabled/sigma
sudo -S systemctl restart nginx
#nginx
```

```
#####################################################################
###############

#####################################################################
###############
#static
wget -q
https://nexus.nmediator.ru/repository/miscelaneous/sigma/install
/sigma-static.tar.gz -O /home/$USERNAME/sigma-static.tar.gz
sudo -S rm -r /IO/sigma-static
sudo -S tar -xvf /home/$USERNAME/sigma-static.tar.gz -C /IO/

sudo -S sed -i 's/\(baseApiUrl:[
\t]*'\''\).*\('\''.*\)/\1https:\/\/'"$IPADDRESS"':444\/test\2/1'
/IO/sigma-static/main.js
sudo -S sed -i 's/\(baseApiUrl:[
\t]*'\''\).*\('\''.*\)/\1https:\/\/'"$IPADDRESS"':444\/test\2/1'
/IO/sigma-static/main.js.map

#static
#####################################################################
###############

#####################################################################
###############
#sigma
wget -q
https://nexus.nmediator.ru/repository/miscelaneous/sigma/install
/sigma.tar.gz -O /home/$USERNAME/sigma.tar.gz
tar -xvf /home/$USERNAME/sigma.tar.gz
find  /home/$USERNAME/sigma -type f -name "*.sh" -exec chmod 777
{} \;
if [ ! -d "/home/$USERNAME/pgdata" ]; then
mkdir /home/$USERNAME/pgdata
fi
#Фиксим параметры
sed -i "s/--volume \/home\/.*\/pgdata:\/postgres.data/--volume
\/home\/$USERNAME\/pgdata:\/postgres.data/1"
/home/$USERNAME/sigma/components/modules/postgres/start.sh
#Запускаем бд на инициализацию
cd /home/$USERNAME/sigma/components
./deploy.sh postgres
echo "Немного подождем запуска"
sleep 10
./deploy.sh mdm
echo "Немного подождем запуска"
sleep 10
cd /home/$USERNAME/sigma/database
mvn install -Ddatabase.update.skip=false
cd /home/$USERNAME/sigma/mdm-database
mvn install -Ddatabase.update.skip=false

#Обновляем json
nextupdate=1
foundnext="false"
while [ "$foundnext" != "true" ]
do
```

```
if $(ls -d /home/$USERNAME/sigma/mdm-update/${nextupdate}
1>/dev/null 2>&1); then
echo Обрабатываю каталог ${nextupdate}
cd /home/$USERNAME/sigma/mdm-update/${nextupdate}
./updatemdm.sh
nextupdate=$((nextupdate+1));
else
foundnext="true";
fi;
done;

cd /home/$USERNAME/sigma/components
./shutdown.sh mdm
./shutdown.sh postgres

sed -i "s/^docker container prune -f$/#docker container prune -
f/1" /home/$USERNAME/sigma/components/shutdown.sh
sed -i "s/^docker image prune -f$/#docker image prune -f/1"
/home/$USERNAME/sigma/components/shutdown.sh
sed -i "s/^docker builder prune -f$/#docker builder prune -f/1"
/home/$USERNAME/sigma/components/shutdown.sh
#sigma
#####################################################################
###############

#####################################################################
###############
#service

read -r -d '' VAR << EOF
[Unit]\n
Description=sigma\n
\n
[Service]\n
Type=oneshot\n
RemainAfterExit=yes\n
User=$USERNAME\n
WorkingDirectory=/home/$USERNAME/sigma/components\n
ExecStart=/home/$USERNAME/sigma/components/deploy.sh\n
ExecStop=/home/$USERNAME/sigma/components/shutdown.sh\n
\n
[Install]\n
WantedBy=multi-user.target\n
EOF

sudo -S bash -c "echo -e \"$VAR\" >
/etc/systemd/system/sigma.service"
sudo -S systemctl enable sigma.service
echo Запускаю сервис sigma
sudo -S systemctl start sigma.service
#service
#####################################################################
###############


#####################################################################
###############
```

```
#sanation
rm /home/$USERNAME/sigma-static.tar.gz
rm /home/$USERNAME/sigma.tar.gz
rm /home/$USERNAME/dowork.sh
rm /home/$USERNAME/v3.ext
#sanation
#####################################################################
###############

exit 0
```