

Документация JumpServer

- [Руководство по быстрому старту](#)
- [Установка](#)
 - [Установка JumpServer Enterprise Edition](#)
 - [Установка JumpServer Community Edition](#)
 - [Эксплуатация и обслуживание с помощью командной строки - jmsctl](#)
 - [Описание сетевых портов](#)
 - [Настройка HTTPS и обратного прокси для веб-интерфейса JumpServer](#)
 - [Настройка HAProxy для HA-кластера JumpServer](#)
 - [Описание и пример настройки HA-кластера JumpServer](#)
- [Системные настройки](#)
 - [Интеграция с Active Directory\(LDAP\) и синхронизация с группами AD](#)
 - [Включение встроенной 2-факторной авторизации\(TOTP\)](#)
 - [Настройка отправки событий по Syslog](#)
 - [Настройка RDS \(RemoteApp\) для публикации приложений](#)
 - [Установка OpenSSH на Windows для управления УЗ Windows](#)
 - [Как установить верные дату и время в JumpServer?](#)
 - [Настройка Panda для публикации приложений](#)
- [Администрирование системы](#)
 - [Настройка блокировки команд SSH и запросов СУБД](#)
 - [Настройка подключения к целевым системам по HTTP\(веб-интерфейсы приложений\)](#)
- [Решение проблем](#)
 - [Проверка работы контейнеров и журналы ошибок](#)
 - [Решение проблем с публикацией приложений RemoteApp](#)
- [Дополнительные возможности](#)
 - [Структура апплетов RemoteApp и создание своего апплета](#)

- Разработка собственных приложений для Panda
- Руководство по проведению пилотного проекта

Руководство по быстрому старту

Добавление устройств.

1. Подготовка

Приготовьте 2 устройства (для SSH и RDP) и одну СУБД для проверки функций продукта.

Например:

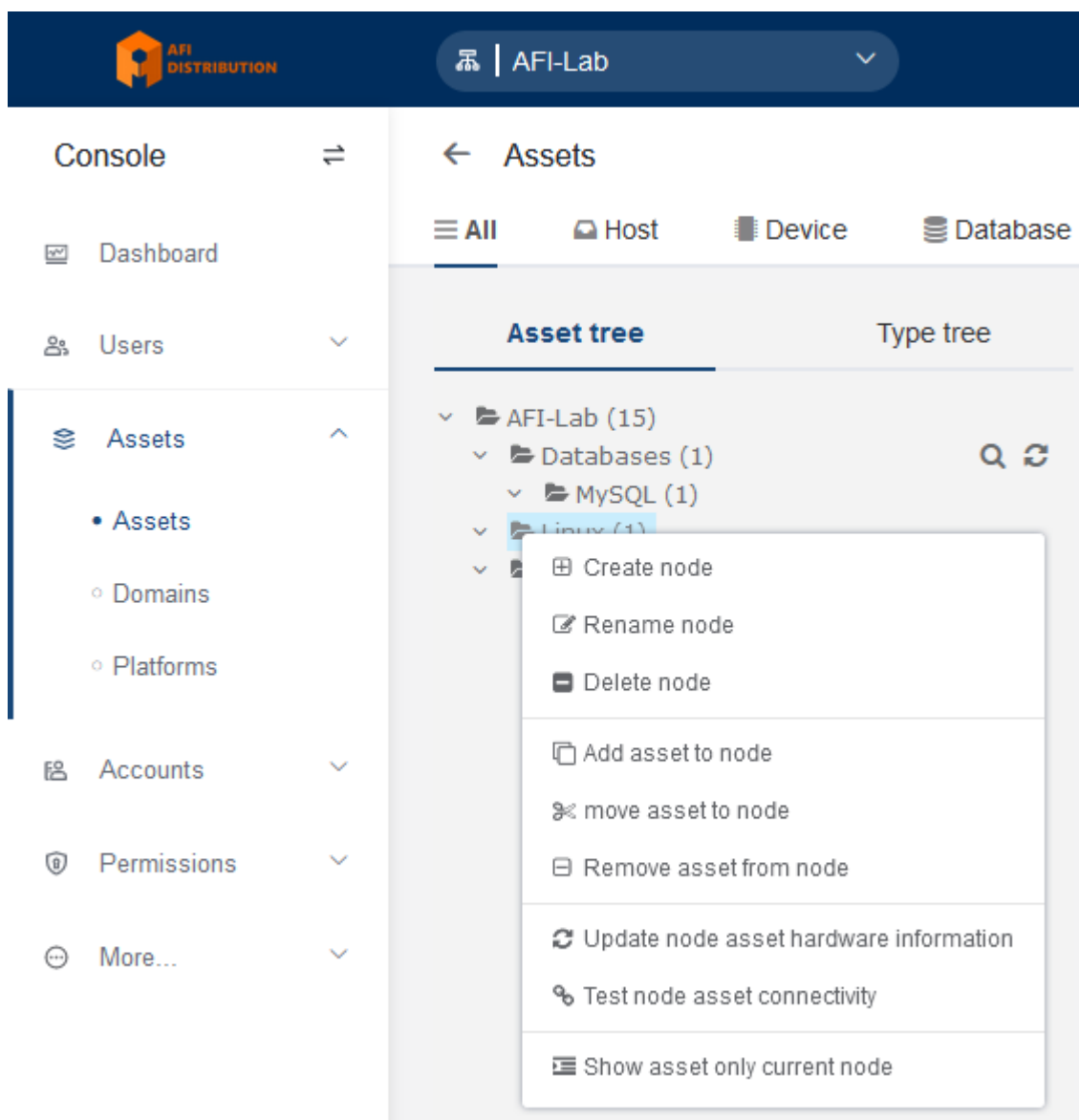
IP\Имя хоста	Порт	Тип	Учетная запись
afidc.afi.local	3389	Windows	testadmin
10.10.53.210	22	Linux	sergg
10.10.53.3	3306	MySQL	user

Если нужно, чтобы с **Windows** собиралась информация об устройстве, выполнялась смена паролей локальных УЗ, создание новых и прочие задачи, для этого необходимо настроить **SSH для Windows**.

Для подключения по **RDP** этого делать **не нужно**.

2. Редактирование дерева активов.

Перейдите в раздел **Console - Assets - Assets**, вкладка **Asset Tree**. Нажмите правую кнопку мышки в этом разделе и откроется меню редактирования дерева устройств.



Здесь вы можете создать папки(**Node**) и под-папки для ваших устройств (пункт **Create node**), все устройства можно сортировать по этим папкам.

- Одно и то же устройство может находиться одновременно в разных папках.
- Эти папки могут быть использованы при настройке политик доступа, например дать доступ ко всем устройствам в соответствующей папке.

3. Добавление активов в систему

Добавление устройства на базе Linux. Устройства Windows, MySQL и другие добавляются аналогичным образом

Нажмите кнопку **Create**, выберите тип устройства **Linux**(в разделе **Host**), заполните поля:

Name: любое понятное вам имя

IP\Host: IP адрес или DNS имя устройства

Platform: Linux

Node: папка или папки, куда будет помещено устройство при создании

Измените номера портов, если для подключения нужно использовать нестандартные порты.

The screenshot shows the APT-Infrastructure AFILab web interface. The left sidebar contains a navigation menu with the following items: Console, Dashboard, Users, Assets (expanded), Domains, Platforms, Accounts, Permissions, and More... The main content area is titled 'Basic' and contains the following sections:

- Basic:** Fields for Name (Lab_Rabbit), IP/Host (10.10.53.210), Platform (Linux), and Node (/AFILabLinux).
- Protocol:** A table with two rows. The first row has 'ssh' in the 'Protocols' column and '22' in the 'Port' column. The second row has 'sftp' in the 'Protocols' column and '22' in the 'Port' column. Below the table is a note: 'Asset support protocol is limited by platform, click the Settings button to view the protocol settings. If you need to update, please update the platform'.
- Account:** A table with the following columns: Name, Username, Privileged, Template add, and Actions. The table is currently empty, showing 'No Data'. Below the table are two buttons: 'Add' and 'Template add'.
- Other:** Fields for Domain (Select) and Label (Select).

Далее в разделе **Account**, ниже окна **Account List** нажмите кнопку **Add** для добавления учетной записи, с которой вы будете подключаться к нему. Введите имя пользователя и пароль и нажмите **Submit** для сохранения учетной записи.

Учетную запись можно будет добавить и позже или вообще не добавлять, если вы не планируете хранить пароли в PAM.

Add account



Basic

* Name

* Username

Tip: If no username is required for authentication, fill in `null`, If AD account, like `username@domain`

Privileged



Secret

Secret type



Password



SSH Key

Password



Other

Push now



Windows assets do not support pushing keys

Is active



Comment

Reset

Submit

4. Настройка разрешения доступа к устройству

Перейдите в раздел **Console - Policies - Authorization** и нажмите кнопку **Create**. Заполните нужные параметры доступа.

Console | AFI Demo

Update the asset authorization rules

Basic

Name: full access

User

Users: TST_User(TST_User) x Administrator(admin) x test-nlo(test-nlo) x testnlo.s(testnlo.s) x

Groups: AD Group_test x

Asset

Assets: Select

Nodes: /AFI Demo x

Account

Accounts: ☒ All existing accounts ☐ Specified accounts ☒ Virtual accounts

Virtual accounts:

☒ Manual account ☒ Same account ☐ Anonymous account

Protocol

Protocols: ☒ All protocol ☐ Specific protocol

Name: любое удобно название для группы доступа

Users: пользователь или пользователи PAM, которые получат доступ к устройствам

Groups: группа или группы пользователей, которые получают доступ к устройствам

Assets: устройство или устройства, к которым получают доступ

Nodes: папка или папки с устройствами, к которым пользователи получают доступ

Account:

- **All existing:** разрешить подключение с любой существующей УЗ для каждого устройства
- **Specified accounts:** указать конкретные УЗ, которые можно использовать для подключения
- **Virtual Accounts:** включение дополнительных параметров авторизации
- **Manual account:** возможность ввести логин и пароль УЗ вручную, без добавления УЗ в систему
- **Same account:** подключение к целевой системе с той же УЗ, с которой пользователь авторизован в PAM (только для LDAP авторизации)
- **Anonymous account:** подключение без УЗ, обычно для подключения к веб-интерфейсам, в которых пользователи сами вводят логин и пароль

Protocol: можно ограничить протокол для подключения

Actions: если тип подключения поддерживает передачу файлов, буфер обмена или совместное использование сессии доступа, то вы можете включить или отключить эти разрешения.

Actions

Actions

- ☒ All
 - ☒ Connect ?
- ☒ Transfer
 - ☒ Upload ?
 - ☒ Download ?
 - ☒ Delete ?
- ☒ Clipboard
 - ☒ Copy ?
 - ☒ Paste ?
- ☒ Share ?

The effects of each permission vary, click the icon next to the permission to view.

Далее можно включить или выключить политику доступа и задать начало и окончание ее работы:

Other

Active ☒

Date start

Date expired

Comment

Нажмите кнопку **Submit** для сохранения настроек

5. Подключение к устройствам

Перейдите в Веб-терминал, нажав соответствующую кнопку справа сверху:



В терминале слева каждый пользователь видит только те устройства, к которым ему разрешено подключаться. Кликните на нужное устройство в списке слева, на экране появится окно выбора учетной записи и типа подключения к устройству:

Connect - MySQL afisql.afi.local

MySQL

Select account

user

Connect Method

Web

Native

Applet

☒ Web CLI

☐ Web GUI

Advanced option

Remember selected

☐ Automatic Login next time (right click asset Connection to re-select)

Connect

Без дополнительных настроек вы сможете подключаться через веб-интерфейс по протоколам **SSH**, **RDP** и **SFTP**, а также к **MySQL** с помощью **Web CLI** или **Web GUI**. Подключения к Kubernetes, веб-интерфейсам, приложениям RemoteApp, СУБД с помощью соответствующих клиентов и другие способы будут описаны в других статьях.

Установка

Установка

Установка JumpServer Enterprise Edition

Оффлайн установка JumpServer:

Для начала нужно запросить у нас актуальный файл дистрибутива по почте support@afi-d.ru или в тг: [@mapceaheh](https://t.me/mapceaheh)

1. Системные требования:

ОС: Linux/AMD64 (arm64) x86_64(aarch64) ядро версии 4.0 или выше (желательно семейства Redhat, Debian, Ubuntu)

CPU: 4 ядра

RAM: 8 ГБ

HDD: 60 ГБ

2. Поместить скачанный файл в директорию

```
/opt
```

3. Выполнить команды(имена файлов могут отличаться с выходом новых версий):

```
$ cd /opt
$ tar -xf jumpserver-offline-installer-v3.10.3-amd64.tar.gz
cd jumpserver-offline-installer-v3.10.3-amd64
```

Далее вы можете отредактировать файл конфигурации, для изменения параметров установки, например для использования внешней СУБД MySQL или изменению папки установки продукта.

```
# nano /opt/jumpserver-offline-installer-v3.10.3-amd64/config-example.txt
```

Запускаем установку:

```
# cd jumpserver-offline-installer-v3.10.3-amd64
# ./jmsctl.sh install
```

Во время установки вам нужно будет подтвердить введенные в файле конфигурации данные или указать другие данные, если вы не заполняли файл конфигурации заранее.

Если вы планируете использовать внешний сервер СУБД, создайте БД по инструкции (<https://t.me/JumpServerPAM/57>).

Запускаем приложение:

```
# ./jmsctl.sh start
```

После завершения установки

перейти в папку с продуктом (имя папки может меняться с выходом новых версий)

```
# cd /opt/jumpserver-installer-v3.10.3
```

запустить приложение

```
# ./jmsctl.sh start
```

4. После этого можно будет зайти в веб-интерфейс по адресу:

http://IP/

логин: **admin**

пароль: **ChangeMe**

и приступить к настройке системы.

Установка JumpServer Community Edition

Внимание: Если вы планируете пилотировать/использовать Enterprise редакцию с расширенными функциями, то нужно сразу ставить ее по этой инструкции ([Установка JumpServer Enterprise Edition](#)), так как переход из Community в Enterprise невозможен без переустановки.

1. Системные требования:

ОС: Linux/AMD64 (arm64) x86_64(aarch64) ядро версии 4.0 или выше

CPU: 4 ядра

RAM: 8 ГБ

HDD: 60 ГБ

2. Установка дополнительных компонентов на примере Debian\Ubuntu:

```
# apt-get update
# apt-get install -y wget curl tar gettext iptables
```

3. Установка JumpServer

Быстрая онлайн установка JumpServer:

В этом случае JumpServer установится с параметрами по-умолчанию, СУБД MySQL и Redis будут установлены в контейнеры на том же сервере.

Выполняем команду:

```
# curl -sSL https://github.com/jumpserver/jumpserver/releases/latest/download/quick_start.sh | bash
```

Дожидаемся процесса выполнения скрипта.

Стандартная онлайн установка:

В этом случае вы можете изменить конфигурацию установки перед развертыванием.

Скачайте последний установщик с GitHub <https://github.com/jumpserver/installer/releases/>

Ниже пример команд для версии 3.10.3

```
# cd /opt/
# wget https://github.com/jumpserver/installer/releases/download/v3.10.3/jumpserver-installer-
```

```
v3.10.3.tar.gz
# tar -xf jumpserver-installer-v3.10.3.tar.gz
```

Далее вы можете отредактировать файл конфигурации, для изменения параметров установки.

Если вы планируете использовать внешний сервер СУБД, создайте БД по инструкции (<https://t.me/JumpServerPAM/57>).

```
# nano /opt/jumpserver-installer-v3.10.3/config-example.txt
```

Запускаем установку:

```
# cd ./jumpserver-installer-v3.10.3
# ./jmsctl.sh install
```

Во время установки вам нужно будет подтвердить введенные в файле конфигурации данные или указать другие данные, если вы не заполняли файл конфигурации заранее.

4. Запуск приложения

После завершения установки нужно перейти в папку с продуктом и запустить приложение

```
# cd /opt/jumpserver-installer-v3.10.3
# ./jmsctl.sh start
```

После этого можно будет зайти в веб-интерфейс по адресу:

http://IP/

логин: **admin**

пароль: **ChangeMe**

И приступить к настройке системы.

Эксплуатация и обслуживание с помощью командной строки - jmsctl

Эксплуатация и обслуживание - jmsctl

JumpServer по умолчанию включает встроенный инструмент командной строки для эксплуатации и обслуживания - **jmsctl**. Для просмотра справочных документов выполните команду:

```
jmsctl help
```

Управление приложением JumpServer:

```
./jmsctl.sh [COMMAND] [ARGS...]  
./jmsctl.sh --help
```

Команды установки:

- install - Установка сервиса JumpServer

Команды управления:

- config - Конфигурация инструмента, выполните `jmsctl config --help` для просмотра справки
- start - Запуск сервиса JumpServer
- stop - Остановка сервиса JumpServer
- restart - Перезапуск сервиса JumpServer
- status - Проверка состояния сервиса JumpServer
- down - Остановка сервиса JumpServer
- uninstall - Деинсталляция сервиса JumpServer

Дополнительные команды:

- load_image - Загрузка Docker-образа
- backup_db - Резервное копирование базы данных JumpServer
- restore_db [file] - Восстановление данных из резервного файла базы данных
- raw - Выполнение команды docker compose
- tail [service] - Просмотр логов сервиса

Описание сетевых портов

Список сетевых портов

JumpServer требует открытия следующих сетевых портов для нормальной работы. Администраторы могут открыть соответствующие порты в сети и на хосте в зависимости от схемы развертывания компонентов JumpServer.

Порт	Назначение	Описание
22	SSH	Установка, обновление и управление
80	Web HTTP сервис	Доступ к веб-интерфейсу JumpServer по HTTP
443	Web HTTPS сервис	Доступ к веб-интерфейсу JumpServer по HTTPS
3306	Сервис базы данных	Используется MySQL
6379	Сервис базы данных	Используется Redis
3389	Razor сервисный порт	Подключение к Windows-активам через RDP Client
2222	SSH Client	Подключение к JumpServer через терминальные инструменты (Xshell, PuTTY и т.д.)
33061	Magnus MySQL сервисный порт	Подключение к MySQL через DB Client
33062	Magnus MariaDB сервисный порт	Подключение к MariaDB через DB Client
54320	Magnus PostgreSQL порт	Подключение к PostgreSQL через DB Client
63790	Magnus Redis порт	Подключение к Redis через DB Client
30000-30100	Magnus Oracle порты	Подключение к Oracle через DB Client, диапазон портов может быть настроен

Установка

Настройка HTTPS и обратного прокси для веб-интерфейса JumpServer

Для чего нужен обратный прокси JumpServer?

Nginx отвечает на поддержку защищенных websockets (wss://), организовывая управление подключениями и защитой канала с помощью SSL-сертификата. Для того чтобы функция копирования и вставки в протоколе RDP работала, необходимо развернуть доверенный SSL-сертификат. Использование копирования и вставки в RDP-активах возможно при доступе через протокол HTTPS.

Установка SSL сертификата и настройка HTTPS для веб-интерфейса

Подготовьте SSL-сертификат (обратите внимание, что сертификат **должен быть в формате PEM**). Сертификаты необходимо разместить в директории **/opt/jumpserver/config/nginx/cert**

Остановите сервис JumpServer:

```
./jmsctl.sh stop
```

Откройте файл конфигурации JumpServer

```
vi /opt/jumpserver/config/config.txt
```

Найдите и измените параметры для конфигурации Nginx:

```
## Конфигурация Nginx
HTTP_PORT=80
SSH_PORT=2222
RDP_PORT=3389

## HTTPS Конфигурация
HTTPS_PORT=443          # Внешний порт для HTTPS, по умолчанию 443
SERVER_NAME=www.domain.com # Ваш домен для HTTPS
SSL_CERTIFICATE=xxx.pem   # Имя вашего сертификата в /opt/jumpserver/config/nginx/cert
SSL_CERTIFICATE_KEY=xxx.key # Имя файла ключа в /opt/jumpserver/config/nginx/cert
```

Сохраните изменения файла конфигурации и запустите JumpServer

```
./jmsctl.sh start
```

Если вам нужно дополнительно настроить файл конфигурации Nginx:

```
vi /opt/jumpserver/config/nginx/lb_http_server.conf
```

Многоуровневый обратный прокси на Nginx

Подсказка:

Эта конфигурация подходит для случаев, когда на верхнем уровне есть общий внешний прокси-сервер. Это пример многоуровневого обратного проксирования на Nginx. Каждая прокси-секция должна быть настроена для поддержки длительных соединений WebSocket.

Редактирование конфигурационного файла:

```
vi /etc/nginx/conf.d/jumpserver.conf
```

Пример конфигурации без SSL:

```
server {

    listen 80;
    server_name demo.jumpserver.org; # Замените на ваш домен

    client_max_body_size 4096m; # Ограничение на максимальный размер загружаемых файлов

    location / {
        # Здесь указывается IP-адрес Nginx сервера JumpServer
        proxy_pass http://192.168.244.144;
        proxy_http_version 1.1;
        proxy_buffering off;
        proxy_request_buffering off;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
```

```
        proxy_set_header X-Forwarded-For $remote_addr;
    }
}
```

Рекомендация:

Для более безопасного доступа рекомендуется настроить SSL и использовать протокол HTTPS, следуя рекомендациям [Mozilla SSL Configuration Generator](#).

Пример конфигурации с SSL:

Перенаправление HTTP на HTTPS:

```
server {
    listen 80;
    server_name demo.jumpserver.org; # Замените на ваш домен
    return 301 https://$server_name$request_uri; # Перенаправление всех HTTP-запросов на HTTPS
}
```

Настройка HTTPS:

```
server {
    listen 443 ssl http2;
    server_name demo.jumpserver.org; # Замените на ваш домен
    ssl_certificate sslkey/1_jumpserver.org_bundle.crt; # Укажите путь к вашему SSL-сертификату
    ssl_certificate_key sslkey/2_jumpserver.org_bundle.key; # Укажите путь к ключевому файлу
    сертификата
    ssl_session_timeout 1d;
    ssl_session_cache shared:MozSSL:10m;
    ssl_ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-
    AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-
    RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384;
    ssl_prefer_server_ciphers off;
    ssl_protocols TLSv1.1 TLSv1.2;
    add_header Strict-Transport-Security "max-age=63072000" always;

    client_max_body_size 4096m; # Ограничение на размер загружаемых файлов и записей

    location / {
        # Здесь указывается IP-адрес Nginx сервера JumpServer
        proxy_pass http://192.168.244.144;
        proxy_http_version 1.1;
        proxy_buffering off;
        proxy_request_buffering off;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $remote_addr;
    }
}
```

3. Другие балансировщики нагрузки (SLB)

Подсказка:

1. Необходимо правильно настроить поддержку длительных соединений WebSocket.
2. Важно учитывать вопросы, связанные с управлением сессиями.

Установка

Настройка HAProxy для HA-кластера JumpServer

HAProxy (High Availability Proxy) — это **программный инструмент с открытым исходным кодом** для балансировки нагрузки и проксирования трафика на уровне сетевого протокола, обычно используемый для распределения нагрузки между несколькими серверами. Он является одним из самых популярных решений для повышения доступности и производительности веб-приложений и служб.

Установка HAProxy, на примере Ubuntu:

```
sudo apt install haproxy -y
```

После установки нужно отредактировать файл конфигурации, это вызывает основные вопросы по настройке.

Файл конфигурации обычно находится в **/etc/haproxy/haproxy.cfg**

Пример файла конфигурации с сайта вендора:

```
global
# to have these messages end up in /var/log/haproxy.log you will
# need to:
#
# 1) configure syslog to accept network log events. This is done
# by adding the '-r' option to the SYSLOGD_OPTIONS in
# /etc/sysconfig/syslog
#
# 2) configure local2 events to go to the /var/log/haproxy.log
# file. A line like the following can be added to
# /etc/sysconfig/syslog
#
# local2.*                /var/log/haproxy.log
#
log      127.0.0.1 local2

chroot   /var/lib/haproxy
pidfile  /var/run/haproxy.pid
maxconn  4000
user     haproxy
group    haproxy
daemon

# turn on stats unix socket
stats socket /var/lib/haproxy/stats
```

```

#-----
# common defaults that all the 'listen' and 'backend' sections will
# use if not designated in their block
#-----

defaults
    log                global
    option              dontlognull
    option              redispatch
    retries              3
    timeout http-request 10s
    timeout queue        1m
    timeout connect      10s
    timeout client       1m
    timeout server       1m
    timeout http-keep-alive 10s
    timeout check        10s
    maxconn              3000

listen stats
    bind *:8080
    mode http
    stats enable
    stats uri /haproxy          # Страница мониторинга, измените по необходимости: http://
192.168.100.100:8080/haproxy
    stats refresh 5s
    stats realm haproxy-status
    stats auth admin:KXOeyNgDeTdpeu9q # # Учетная запись и пароль для доступа к http://
192.168.100.100:8080/haproxy

#-----
# check Параметры проверки активности
# inter Интервал времени, единица: миллисекунды
# rise Количество последовательных успешных попыток, единица: раз
# fall Количество последовательных неудачных попыток, единица: раз
# Пример: inter 2s rise 2 fall 3
# Означает проверку состояния каждые 2 секунды, 2 последовательных успеха указывают на
нормальную работу сервиса, 3 последовательных неудачи указывают на ошибку в сервисе
#
# server Параметры сервера
# server 192.168.100.21 192.168.100.21:80 weight 1 cookie web01
# Первый 192.168.100.21 используется как идентификатор для отображения, его можно
заменить на любой другой строковый идентификатор
# Второй 192.168.100.21:80 — это фактический адрес и порт задействованного сервиса
# weight указывает на вес сервера, при наличии нескольких узлов балансировка нагрузки
будет зависеть от веса
# cookie определяет, что на стороне клиента в cookie будет содержаться этот идентификатор,
чтобы различать текущий используемый задний узел

```

```

# Пример: server db01 192.168.100.21:3306 weight 1 cookie db_01
#-----

listen jms-web
    bind *:80                # Прослушивание порта 80
    mode http

    # redirect scheme https if !{ ssl_fc } # Перенаправление на https
    # bind *:443 ssl crt /opt/ssl.pem      # Настройка https

    option httpchk GET /api/health/      # Интерфейс проверки активности Core

    stick-table type ip size 200k expire 30m
    stick on src

    balance leastconn
    server 192.168.100.21 192.168.100.21:80 weight 1 cookie web01 check inter 2s rise 2 fall 3 #
JumpServer 002708
    server 192.168.100.22 192.168.100.22:80 weight 1 cookie web02 check inter 2s rise 2 fall 3
    server 192.168.100.23 192.168.100.23:80 weight 1 cookie web03 check inter 2s rise 2 fall 3
    server 192.168.100.24 192.168.100.24:80 weight 1 cookie web03 check inter 2s rise 2 fall 3

listen jms-ssh
    bind *:2222
    mode tcp

    option tcp-check

    fullconn 500
    balance source
    server 192.168.100.21 192.168.100.21:2222 weight 1 check inter 2s rise 2 fall 3 send-proxy
    server 192.168.100.22 192.168.100.22:2222 weight 1 check inter 2s rise 2 fall 3 send-proxy
    server 192.168.100.23 192.168.100.23:2222 weight 1 check inter 2s rise 2 fall 3 send-proxy
    server 192.168.100.24 192.168.100.24:2222 weight 1 check inter 2s rise 2 fall 3 send-proxy

listen jms-koko
    mode http

    option httpclose
    option forwardfor
    option httpchk GET /koko/health/ HTTP/1.1\r\nHost:\ 192.168.100.100 # KoKo 000000, host 0000
HAProxy 70 ip 5757

    cookie SERVERID insert indirect
    hash-type consistent
    fullconn 500
    balance leastconn
    server 192.168.100.21 192.168.100.21:80 weight 1 cookie web01 check inter 2s rise 2 fall 3
    server 192.168.100.22 192.168.100.22:80 weight 1 cookie web02 check inter 2s rise 2 fall 3
    server 192.168.100.23 192.168.100.23:80 weight 1 cookie web03 check inter 2s rise 2 fall 3

```

```

server 192.168.100.24 192.168.100.24:80 weight 1 cookie web03 check inter 2s rise 2 fall 3

listen jms-lion
    mode http

    option httpclose
    option forwardfor
    option httpchk GET /lion/health/ HTTP/1.1\r\nHost:\ 192.168.100.100 # Lion 000000, host 0000
HAProxy 0 ip 0000

    cookie SERVERID insert indirect
    hash-type consistent
    fullconn 500
    balance leastconn
server 192.168.100.21 192.168.100.21:80 weight 1 cookie web01 check inter 2s rise 2 fall 3
server 192.168.100.22 192.168.100.22:80 weight 1 cookie web02 check inter 2s rise 2 fall 3
server 192.168.100.23 192.168.100.23:80 weight 1 cookie web03 check inter 2s rise 2 fall 3
server 192.168.100.24 192.168.100.24:80 weight 1 cookie web03 check inter 2s rise 2 fall 3

listen jms-magnus
    bind *:30000
    mode tcp

    option tcp-check

    fullconn 500
    balance source
server 192.168.100.21 192.168.100.21:30000 weight 1 check inter 2s rise 2 fall 3 send-proxy
server 192.168.100.22 192.168.100.22:30000 weight 1 check inter 2s rise 2 fall 3 send-proxy
server 192.168.100.23 192.168.100.23:30000 weight 1 check inter 2s rise 2 fall 3 send-proxy
server 192.168.100.24 192.168.100.24:30000 weight 1 check inter 2s rise 2 fall 3 send-proxy

```

После изменения файла конфигурации запустите службу haproxy:

```

systemctl enable haproxy
systemctl start haproxy

```


Описание и пример настройки HA-кластера JumpServer

JumpServer (в том числе Community Edition) полностью поддерживает кластеризацию HA без каких-либо ограничений. В этой статье я покажу, как это работает.

Зачем нужен HA-кластер для JumpServer

HA (High Availability) кластер для JumpServer необходим для обеспечения высокой доступности системы и минимизации простоев. Он позволяет:

1. **Избежать простоев:** Если один из узлов кластера выходит из строя, другой продолжает обслуживать запросы пользователей.
2. **Обеспечить отказоустойчивость:** Кластеризация позволяет системе автоматически переключаться на доступные узлы при возникновении ошибок.
3. **Увеличить производительность:** Нагрузка распределяется между несколькими узлами, что улучшает отклик системы при одновременном использовании большого числа пользователей.
4. **Повысить надежность хранения данных:** Использование общих ресурсов, таких как MySQL и Redis, с поддержкой кластеризации минимизирует риск потери данных.
5. **Масштабируемость:** Кластер можно легко расширить, добавив дополнительные узлы для обслуживания большего числа пользователей и задач.

Эта архитектура особенно важна для организаций, где JumpServer используется как критически важная система доступа и контроля.

Компоненты кластера JumpServer.

Ноды\Узлы JumpServer - основные узлы кластера, сервера с установленным JumpServer, каждый сервер не содержит "полезных" данных, можно клонировать\копировать, удалять\добавлять узлы JumpServer.

База данных MySQL\PostgreSQL - основная СУБД для хранения всех данных JumpServer: хранит настройки системы, параметры устройств, учетных записей, пароли к целевым системам. Также по умолчанию хранит текстовые логи сессий: команды SSH, SQL запросы, введенные команды с клавиатуры в RDP сессии.

По умолчанию JumpServer создает\использует PostgreSQL внутри контейнера на том же сервере, где устанавливаете JumpServer.

База данных Redis - вспомогательная база данных для кэширования, может быть как единой для всего кластера так и отдельными базами для каждой ноды кластера. По умолчанию *JumpServer* создает\использует *Redis* внутри контейнера на том же сервере, где устанавливаете *JumpServer*.

Хранилище видеозаписей - по умолчанию хранит записи сессий в папке с продуктом **\$folder/core/data/media**, где **\$folder** - папка указанная в основном конфиг-файле, по умолчанию **VOLUME_DIR=/data/jumpserver**. Через веб-интерфейс продукта можно настроить внешнее хранилище видеозаписей: **SFTP, S3, Ceph, Minlo и другие**

Хранилище логов команд - по умолчанию хранятся в основной БД, через веб-интерфейс можно настроить хранение команд и запросов в **Elasticsearch**.

Балансировщик - обычно на базе HAProxy, но можно использовать любой.

Архитектура кластера.

Обычно кластер *JumpServer* состоит из 2 и более узлов кластера, которые:

- подключены к общей БД(или кластеру) MySQL\PostgreSQL
- подключены к общему Redis(или каждый к своему)
- имеют общую СХД для хранения видеозаписей:
 - общая папка данными **\$folder/core/data/ (реализуется обычно с помощью NFS сервера)**
 - ИЛИ
 - общая внешняя СХД для хранения сессий: SFTP, S3, Ceph, Minlo и другие
 - ИЛИ
 - отдельная СХД для каждой ноды
- имеют общую СХД для записей логов команд:
 - на базе общей БД(по умолчанию)
 - ИЛИ
 - на базе общей внешней Elasticsearch
 - ИЛИ
 - отдельный Elasticsearch для каждой ноды кластера
- балансировщик

Пример создания кластера HA JumpServer из 2 узлов

Пример создания кластера *JumpServer*

- с единой общей БД MySQL
- с единой общей БД Redis
- с общей папкой для хранения видеозаписей **\$folder/core/data/** с помощью NFS сервера Linux

Для этого нам потребуется:

1. **Сервер с NFS, MySQL, Redis:**
 - 4 CPU, 8 ГБ оперативной памяти.
2. **Узел JumpServer Node1:**
 - 4 CPU, 8 ГБ оперативной памяти, 100Гб свободного места на диске
3. **Узел JumpServer Node2:**
 - 4 CPU, 8 ГБ оперативной памяти, 100Гб свободного места на диске
4. **Сервер HAProxy** (или другой балансировщик нагрузки).

1. Подготовка сервера с NFS, MySQL и Redis

- **Сервер:** Ubuntu 22.04, IP: 10.10.50.10

Установка и настройка NFS

Команды могут отличаться для других версий Linux, но в целом нужно создать общую папку:

```
sudo apt install nfs-kernel-server
sudo mkdir -p /data
sudo chown -R nobody:nogroup /data/
sudo chmod 777 /data/
sudo nano /etc/exports
```

Добавьте строку в файл `/etc/exports`:

```
/data 10.10.50.10/24(rw,sync,no_subtree_check)
```

Примените настройки и перезапустите NFS-сервис:

```
sudo exportfs -a
sudo systemctl restart nfs-kernel-server
```

Установка и настройка MySQL

Инструкции зависят от версии ОС. Для создания базы данных и пользователя выполните:

```
mysql -uroot
mysql> create database jumpserver default charset 'utf8';
mysql> set global validate_password_policy=LOW;
mysql> create user 'jumpserver'@'%' identified by 'KXOeyNgDeTdpeu9q';
mysql> grant all on jumpserver.* to 'jumpserver'@'%';
mysql> flush privileges;
mysql> exit;
```

Не забудьте настроить фаерволл для открытия порта MySQL (3306).

Установка и настройка Redis

Инструкции зависят от версии ОС. После установки Redis выполните:

```
sed -i "s/bind 127.0.0.1/bind 0.0.0.0/g" /etc/redis.conf
sed -i "s/561i maxmemory-policy allkeys-lru/ /etc/redis.conf
sed -i "s/481i requirepass KXOeyNgDeTdpeu9q/ /etc/redis.conf
```

Это позволит доступ к Redis с паролем `KXOeyNgDeTdpeu9q`. Обязательно используйте уникальный пароль для вашего сервера. Откройте порт `6379` в файрволле.

2. Установка JumpServer

Установка первой ноды JumpServer

Монтирование каталога NFS

Установите клиент NFS, смонтируйте папку и настройте автоматическое монтирование при загрузке:

```
sudo apt install nfs-common
mkdir -p /opt/jumpserver/core/data
mount -t nfs 10.10.50.10:/data /opt/jumpserver/core/data
echo "10.10.50.10:/data /opt/jumpserver/core/data nfs defaults 0 0" >> /etc/fstab
```

Настройка конфигурации JumpServer

Редактируйте файл `config-example.txt` в папке установщика:

```
# Измените следующие параметры, остальные оставьте по умолчанию.
# ВАЖНО: SECRET_KEY должен совпадать на всех узлах JumpServer, иначе данные не будут
расшифровываться.

VOLUME_DIR=/opt/jumpserver

SECRET_KEY=
BOOTSTRAP_TOKEN=
LOG_LEVEL=ERROR
SESSION_EXPIRE_AT_BROWSER_CLOSE=True

# MySQL

DB_HOST=10.10.50.10
DB_PORT=3306
DB_USER=jumpserver
DB_PASSWORD=KXOeyNgDeTdpeu9q
DB_NAME=jumpserver

# Redis

REDIS_HOST=10.10.50.10
REDIS_PORT=6379
REDIS_PASSWORD=KXOeyNgDeTdpeu9q

# KoKo Lion
SHARE_ROOM_TYPE=redis
REUSE_CONNECTION=False
```

Запустите установку:

```
./jmsctl.sh install
```

После завершения установки вы получите значения для:

```
SECRET_KEY=kWQdmdCQKjaWIHYpPhkNQDkfaRuIM6YnHctsHLISPs8287o2kW  
BOOTSTRAP_TOKEN=KXOeyNgDeTdpeu9q
```

Установка второй ноды JumpServer

Установите клиент NFS, смонтируйте папку точно также как на первой ноды.

При редактировании файла конфигурации JumpServer **заполните значения** **SECRET_KEY** и **BOOTSTRAP_TOKEN** , полученные после установки первой ноды:

```
VOLUME_DIR=/opt/jumpserver  
  
SECRET_KEY=kWQdmdCQKjaWIHYpPhkNQDkfaRuIM6YnHctsHLISPs8287o2kW  
BOOTSTRAP_TOKEN=KXOeyNgDeTdpeu9q  
LOG_LEVEL=ERROR  
SESSION_EXPIRE_AT_BROWSER_CLOSE=True  
  
# MySQL  
  
DB_HOST=10.10.50.10  
DB_PORT=3306  
DB_USER=jumpserver  
DB_PASSWORD=KXOeyNgDeTdpeu9q  
DB_NAME=jumpserver  
  
# Redis  
  
REDIS_HOST=10.10.50.10  
REDIS_PORT=6379  
REDIS_PASSWORD=KXOeyNgDeTdpeu9q  
  
# KoKo Lion  
SHARE_ROOM_TYPE=redis  
REUSE_CONNECTION=False
```

И запустите установку

```
./jmsctl.sh install
```

Результат

После завершения настройки вы получите два узла JumpServer, которые используют один MySQL/Redis сервер и хранилище NFS. Вы можете использовать любой из узлов

для доступа к целевым устройствам или настроить HAProxy для автоматического перенаправления пользователей на активный узел.

Системные настройки

Интеграция с Active Directory(LDAP) и синхронизация с группами AD

Настройка интеграции с Active Directory

1. Зайдите в раздел "**System settings**" - "**Auth**", выбираем вкладку **LDAP**.
2. Введите адрес LDAP сервера, учётную запись для подключения к нему и пароль от этой учётной записи.
3. Укажите OU и фильтр поиска пользователей. Пример фильтра пользователей конкретной группы смотрите на скриншоте.

The screenshot shows the 'Auth' settings page for the Axiom Auth System, specifically the 'LDAP' tab. The left sidebar contains a menu with options: Settings, Basic, Organizations, Messages, Features, Auth (selected), Storage, Terminal, Applets, Security, Interface, Tools, Tasks, and License. The main content area is titled 'Auth' and has sub-tabs: Basic, LDAP (selected), CAS, Passkey, OIDC, SAML2, OAuth2, WeCom, DingTalk, FeiShu, Slack, and Radius. The 'Basic' section includes a toggle for 'Enable LDAP auth' (checked), an 'LDAP server' field with the value 'ldap://afidc.afilocal:389' and a hint 'eg: ldap://localhost:389', a 'Bind DN' field with the value 'testadmin@afilocal', and a 'Password' field. The 'LDAP User' section includes a 'User OU' field with the value 'DC=afilocal' and a hint 'Use | split multi OUs', a 'User search filter' field with the value '(&(objectClass=user)(memberOf=CN=AFI_IT,CN=Users,DC=afilocal))' and a hint 'Choice may be (cn|uid|sAMAccountName)=%(user)s', and a 'User attr map' field with a JSON object:

```
{ 1: { 2: "username": "sAMAccountName", 3: "name": "cn", 4: "email": "mail" 5: }
```

4. Нажмите кнопку "**Submit**" для сохранения настроек. Внимание: после изменения параметров и настроек нужно всегда нажимать кнопку "**Submit**" для применения настроек, иначе тест будет запускаться со старыми параметрами.

5. Нажмите кнопку **"Test connection"** для проверки настроек или **"Test login"** для проверки авторизации конкретного пользователя.

6. Нажмите кнопку **"Bulk Import"**. Вы должны увидеть пользователей группы, которые будут добавлены для авторизации в РАМ. Там же можно выделить нужных пользователей и нажать **"Import"** или импортировать всех, нажав **"Import all"**.

7. Также вы можете настроить автоматическую синхронизацию пользователей, нажав кнопку **"Sync setting"**.

The screenshot shows a 'Sync setting' dialog box with the following fields and controls:

- Organization:** A dropdown menu showing 'Default' with a close icon.
- Periodic perform:** A toggle switch that is currently turned on.
- Regularly perform:** A text input field containing the cron expression `*/15 * * * *`. Below it, there is explanatory text: 'For example: every Sunday at 03:05 execute <5 3 * * 0>' and 'Using the 5-bit Linux crontab expression <minute hour day month week> ([Online tool](#))'. A note states: 'If both regularly perform and cycle perform execution are set, use regularly perform first'.
- Cycle perform:** A text input field containing the value '1'. Below it, the text 'Unit hour' is displayed.
- Recipient:** A dropdown menu with the text 'Select'.
- Buttons:** 'Reset' and 'Submit' buttons at the bottom.

Синхронизация с группами Active Directory

Для чего используется синхронизация с AD группами?

Управлять правами доступа к целевым системам можно привычными группами Active Directory - добавление или удаление пользователя из таких групп будет автоматически синхронизироваться с матрицей прав в JumpServer, и пользователь будет получать или терять права доступа.

Настройка синхронизации с группами AD.

1. Зайдите в **System settings - Authentication - LDAP**

2. В поле **User attribute** добавьте параметр **groups**, так чтобы получилось:

```
{
  "username": "sAMAccountName",
  "name": "cn",
  "email": "mail",
```

```
"groups": "memberOf"
}
```

См скриншот:

Basic

LDAP



* Server ?

ldap://afidc.afi.local:389

* Bind DN ?

testadmin@afi.local

Password ?

Password

Search

* Search OU ?

DC=afi,DC=local

* Search filter ?

(&(objectClass=user)(memberOf=CN=AFI_IT,CN=Users,DC=afi,DC=local))

* User attribute ?

1

2

3

4

5

6

"username": "sAMAccountName",

"name": "cn",

"email": "mail",

"groups": "memberOf"

}

3. Нажмите кнопку **Submit** для сохранения настроек

4. Нажмите кнопку **User Import** и в открывшемся окне нажмите **Sync Users**

Если все верно, вы увидите список пользователей и столбец с атрибутами групп **AD:**

Ldap user

Please submit ldap configuration before import

Search

<input type="checkbox"/>	Username	Name	Email	Groups	Already exists
<input type="checkbox"/>	denis	Морозов Денис	-	CN=TestJS,OU=subOU,OU=TestOU,DC=afi,DC=local CN=AFL_IT...	Yes
<input type="checkbox"/>	sergey	Попцов Сергей	-	CN=AFL_IT,CN=Users,DC=afi,DC=local	Yes
<input type="checkbox"/>	nlo	Наталья Орлова	no@afi-d.ru	CN=TestJS,OU=subOU,OU=TestOU,DC=afi,DC=local CN=thyc...	Yes
<input type="checkbox"/>	Вася	Вася	-	CN=TestJS,OU=subOU,OU=TestOU,DC=afi,DC=local CN=AFL_IT...	Yes
<input type="checkbox"/>	testnlo	testnlo	-	CN=AFL_IT,CN=Users,DC=afi,DC=local CN=Domain Admins,CN...	Yes
<input type="checkbox"/>	TST_User	TST_User	-	CN=AFL_IT,CN=Users,DC=afi,DC=local	Yes

Total 6

15/page

1

Sync users

Import

Import all

Cancel

5. Нажмите **Import all**, чтобы добавить пользователей в систему.

Если зайдете в **Console - User - Groups**, увидите группы пользователей JS, с именами групп AD с теми же пользователями в них:

JumpServer

Console

Groups

+ Create

Actions

<input type="checkbox"/>	Name	Users
<input type="checkbox"/>	AD AFL_IT	6
<input type="checkbox"/>	AD Domain Admins	2
<input type="checkbox"/>	AD Remote Desktop Users	1
<input type="checkbox"/>	AD TestJS	3
<input type="checkbox"/>	AD thycotic	1
<input type="checkbox"/>	Default	16

Включение встроенной 2-факторной авторизации(TOTP)

В версии **Community Edition** доступна двухфакторная авторизация через **TOTP** (Google Authenticator)

Чтобы ее включить, перейдите в раздел **System setting - Security - Auth Security**. Параметр **Global MFA auth** позволяет выключить двухфакторную авторизацию или включить ее для всех пользователей или только для администраторов.

Внимание: для корректной работы **TOTP**, на сервере JumpServer необходимо настроить NTP сервис для получения точного времени.

MFA

Global MFA auth ☐ Not enabled ☒ All users ☐ Only admin users

MFA in login page ☐

Eu security regulations(GDPR) require MFA to be on the login page

Third-party login users perform MFA ☐

authentication

The third-party login modes include OIDC, CAS, and SAML2

* MFA verify TTL

3600

Unit: second, The verification MFA takes effect only when you view the account password

OTP issuer name

JumpServer

* OTP valid window

2

Reset

Submit

В **JumpServer Enterprise** также доступны другие варианты двухфакторной авторизации, например двухфакторная авторизация с помощью **RADIUS**.

Настройка отправки событий по Syslog

1. Изменение конфигурационного файла JumpServer

Конфигурационные файлы JumpServer по умолчанию находятся в: `/opt/jumpserver/config/config.txt`

В конфигурацию JumpServer необходимо добавить следующие элементы:

```
# Настройка syslog
SYSLOG_ENABLE=true
SYSLOG_ADDR=10.1.12.116:514 # IP и порт сервера Syslog
SYSLOG_FACILITY=local2 # В соответствии с конфигурацией файла Syslog
```

2. Перезапуск JumpServer

После изменения конфигурационного файла JumpServer необходимо перезапустить для загрузки новых конфигураций.

Команда:

```
jmsctl restart
```

3. Проверка конфигурации

Войдите в службу JumpServer, чтобы создать журнал входа, и проверьте, есть ли вывод на сервере Syslog. Пример выходного журнала входа:

```
[root@jumpserver ~]# cat /tmp/messages
Apr 18 16:27:23 10.1.14.125 root: message:rsyslog logging From JumpServer
Apr 18 16:27:42 10.1.14.125 root: message:rsyslog logging From JumpServer(UDP)
Apr 18 16:40:42 10.1.14.125 jumpserver: login_log - {"backend": "Password", "backend_display": "密码", "city": "局域网", "datetime": "2023/04/18 16:34:08 +0800", "id": "adf0e434-e306-4693-9a51-23f256cb025d", "ip": "10.1.10.35", "mfa": {"label": "禁用", "value": 0}, "reason": "", "reason_display": "", "status": {"label": "成功", "value": true, "type": {"label": "Web", "value": "W"}}, "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0", "username": "admin"}
```

4. Анализ информации журнала Syslog

Тип события	Пример записи Syslog
-------------	----------------------

Вход в систему	Apr 19 15:25:11 10.1.14.125 jumpserver: login_log - {"backend": "Password", "backend_display": "пароль", "city": "local", "datetime": "2023/04/19 15:18:36 +0800", "id": "cfc378e5-6337-4bf9-a8ac-15f33c2b0314", "ip": "10.1.10.35", "mfa": {"label": "отключено", "value": 0}, "reason": "", "reason_display": "", "status": {"label": "успешно", "value": true}, "type": {"label": "Web", "value": "W"}, "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, как Gecko) Chrome/112.0.0.0 Safari/537.36 Edg/112.0.1722.48", "user_name": "admin"}
Загрузка файла	Apr 19 15:27:26 10.1.14.125 jumpserver: ftp_log - {"account": "root(root)", "asset": "10.1.12.182-root(10.1.12.182)", "date_start": "2023/04/19 15:20:51 +0800", "filename": "/tmp/vmware-root/файл.pdf", "id": "6e7721c0-2091-49fb-8853-fc18e0a2e432", "is_success": true, "operate": {"label": "uploading", "value": "upload"}, "org_id": "00000000-0000-0000-0000-000000000002", "remote_addr": "10.1.10.35", "user": "Administrator(admin)"}
Скачивание файла	Apr 19 15:28:08 10.1.14.125 jumpserver: ftp_log - {"account": "root(root)", "asset": "10.1.12.182-root(10.1.12.182)", "date_start": "2023/04/19 15:21:33 +0800", "filename": "/tmp/vmware-root/файл.pdf", "id": "113c0601-80c1-47d1-a053-5038fd89698c", "is_success": true, "operate": {"label": "скачивание файла", "value": "download"}, "org_id": "00000000-0000-0000-0000-000000000002", "remote_addr": "10.1.10.35", "user": "Administrator(admin)"}
Выполнение операции	Apr 19 15:28:44 10.1.14.125 jumpserver: operation_log - {"action": {"label": "update", "value": "update"}, "datetime": "2023/04/19 15:22:09 +0800", "id": "f844f014-2ac5-459d-abd0-ec8f853fa09c", "org_id": "00000000-0000-0000-0000-000000000004", "org_name": "SYSTEM", "remote_addr": "10.1.10.35", "resource": "GLOBAL", "resource_type": "System settings", "user": "Administrator(admin)"}
Смена пароля	Apr 19 15:29:58 10.1.14.125 jumpserver: password_change_log - {"change_by": "Administrator(admin)", "datetime": "2023/04/19 15:23:23 +0800", "id": "0cd278ed-8335-49d5-a0c3-0211e9858441", "remote_addr": "10.1.10.35", "user": "Евгений МЕН(МЕН)"}
Запуск сессии доступа	Apr 19 15:31:29 10.1.14.125 jumpserver: host_session_log - {"account": "root(root)", "account_id": "49536b5e-bf06-4d16-bacd-7d628de3a3f2", "asset": "10.1.12.182-root(10.1.12.182)", "asset_id": "dfba9962-7988-4d29-9b04-6f82dd8e02c3", "can_join": true, "can_replay": false, "can_terminate": true, "comment": null, "date_end": null, "date_start": "2023/04/19 15:24:54 +0800", "has_command": false, "has_replay": false, "id": "4896b882-299a-4759-804e-32250f5b05b7", "is_finished": false, "is_success": true, "login_from": {"label": "веб-терминал", "value": "WT"}, "org_id": "00000000-0000-0000-0000-000000000002", "org_name": "default", "protocol": "ssh", "remote_addr": "10.1.10.35", "terminal": {"id": "7076d4aa-4050-4a2f-855b-2af7a7bd6674", "name": "[KoKo]-jumpserver-v3-86c4b2fc7167", "type": {"label": "normal", "value": "normal"}, "user": "Administrator(admin)", "user_id": "cdab8252-0f45-46d0-9872-b2c7c52022fd"}}
Выполнение команды	Apr 19 15:34:00 10.1.14.125 jumpserver: session_command_log - {"account": "root(root)", "asset": "10.1.12.182-root(10.1.12.182)", "id": "28400256-e9e2-4454-8127-4880fe5b9684", "input": "free -h", "org_id": "00000000-0000-0000-0000-000000000002", "output": "free -h\r\n\n total used free shared buff/cache available\r\nMem: 7.6G 4.3G 136M 28M 3.2G 3.0G", "remote_addr": "10.1.10.35", "risk_level": {"label": "обычный", "value": 0}, "session": "4896b882-299a-4759-804e-32250f5b05b7", "timestamp": 1681889159, "timestamp_display": "2023/04/19 15:25:59 +0800", "user": "Administrator(admin)"}

Настройка RDS (RemoteApp) для публикации приложений

Примечание: Community Edition поддерживает только режим публикации HTTP приложений.

RemoteApp - это публикация приложений на Microsoft RDS. Для ее использования вам необходим Windows Server с настроенным RDS(RemoteApp). **JumpServer** сможет подключаться к приложениям, опубликованным на RDS сервере и авторизовываться в них. В основном это актуально для приложений для работы с СУБД и веб-интерфейсами.

Для поддержки RemoteApp необходимо настроить JumpServer и сервер RDS.

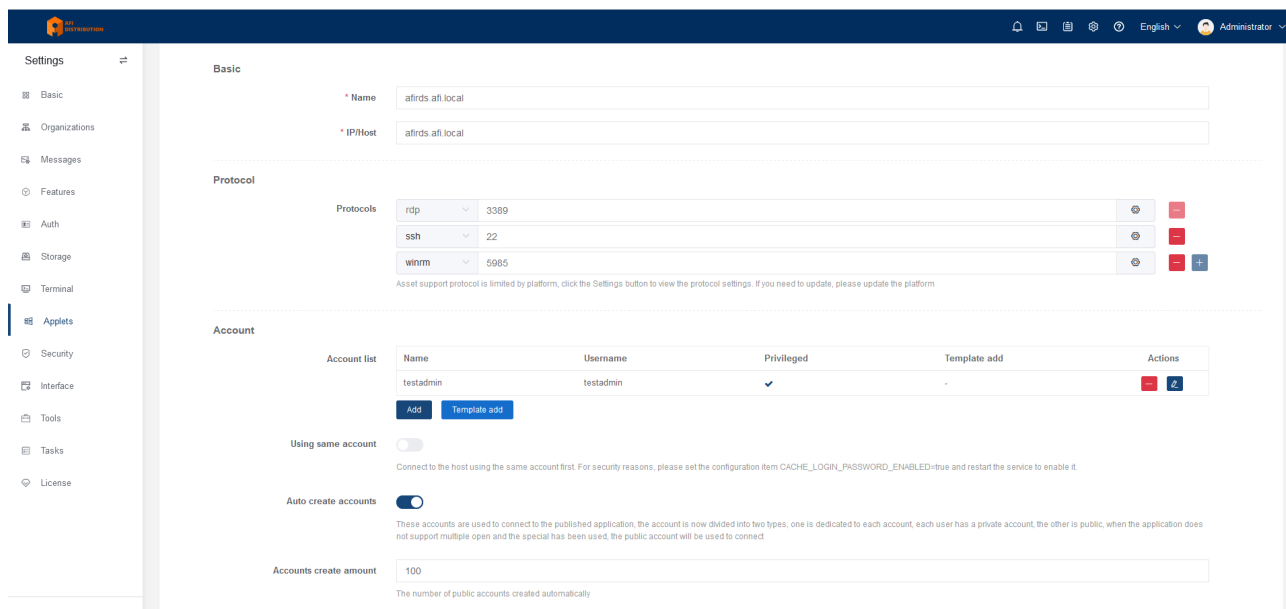
Требования:

- MS Windows Server 2016 или MS Windows Server 2019
- Установленная роль RDS (Remote Desktop Services)

Настроенный WinRM или установленный OpenSSH

Добавление сервера публикаций в JumpServer

Зайдите в "**System settings - Applets**", выберите вкладку "**Remote Hosts**" и нажмите "**Create**".



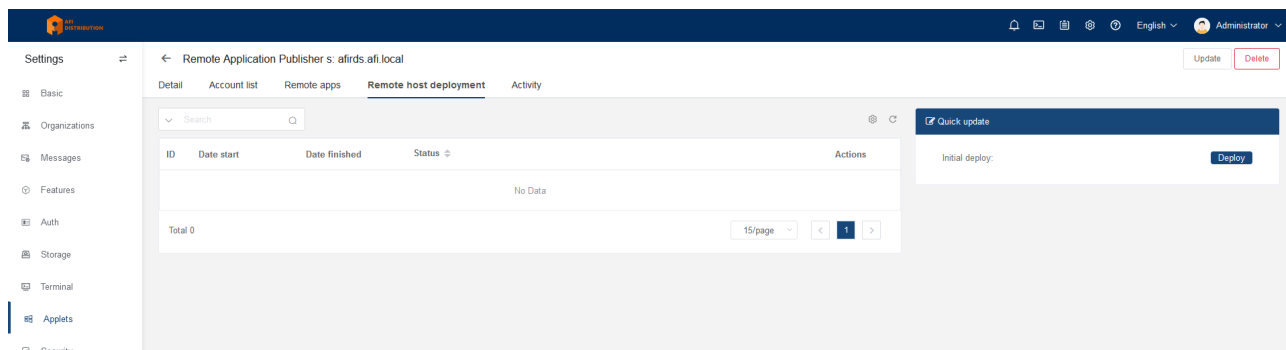
Описание параметров:

Параметр	Описание
Name	Имя устройства
IP/Host	IP или сетевое имя RDS сервера
Protocol group	Протоколы и номера портов. Укажите здесь WinRM или SSH, если будет использоваться
Account List	Учетная запись с правами администратора для доступа к RDS серверу
Automatically create an account	Включить автоматическое создание учетных записей для подключения к опубликованному серверу
Number of accounts created	Количество создаваемых учетных записей.
Core service address	Адрес связи между агентом машины для публикации удалённых приложений и базой данных. Замените адрес http://127.0.0.1 на IP вашего сервера
RDS license	Настройка сервера лицензирования RDS
RDS License Server	Параметры сервера лицензирования RDS.
RDS authorization mode	Выберите "Device" или "User" для настройки режима авторизации. A. Device: Позволяет одному устройству (любому пользователю, использующему его для публикации удалённых приложений). B. User: Предоставляет одному пользователю доступ к серверу публикаций удалённых приложений неограниченного числа клиентских компьютеров или устройств.
RDS single user single session	Выберите "Disable" или "Enable" для настройки режима одиночной сессии для одного пользователя. A. Disable: Разрешить каждому пользователю подключаться к удалённому рабочему столу одновременно. B. Enable: Запретить каждому пользователю подключаться к удалённому рабочему столу одновременно.
RDS maximum disconnect time	Если сессия достигает этого максимального времени, соединение разрывается.
RDS remote application logout time limit	Время выхода после разрыва сессии удалённого приложения.

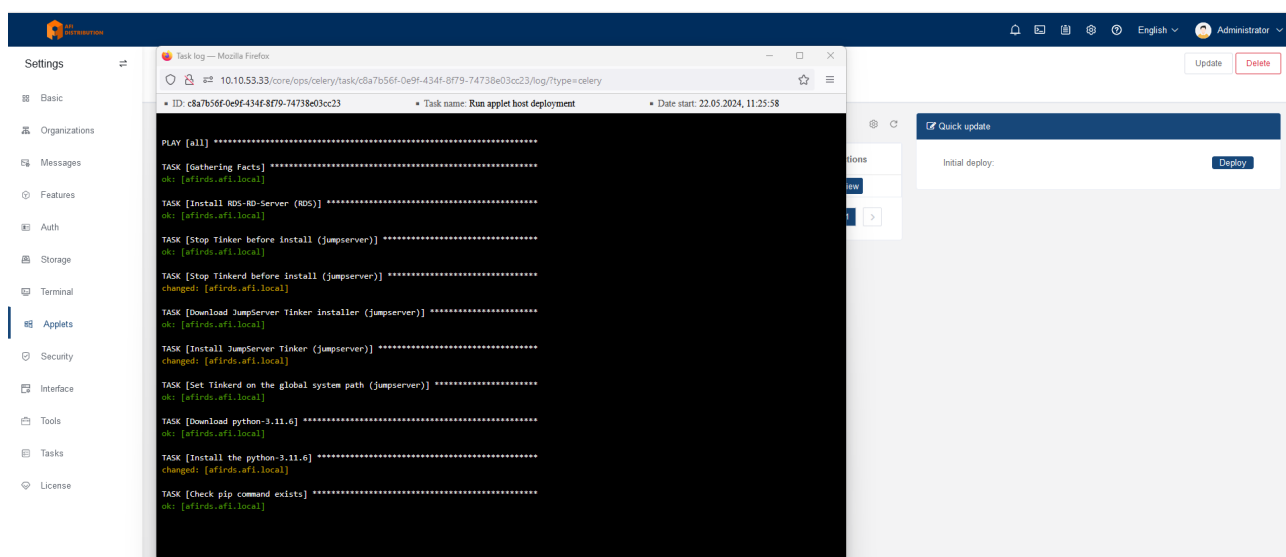
Нажмите **"Submit"** для сохранения настроек.

Установка механизма публикации приложений

Нажмите на название добавленного сервера публикаций. Откроется информация о сервере, перейдите во вкладку **"Remote host deployment"** и нажмите кнопку **"Deploy"** в правой части экрана.



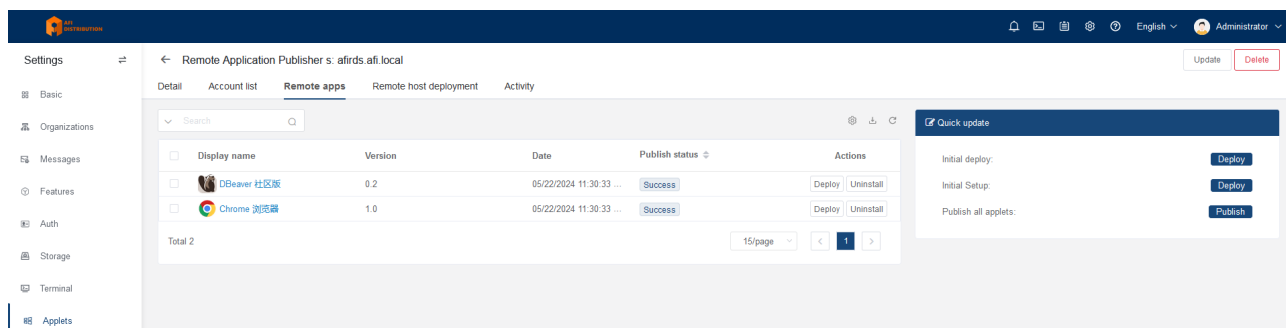
Откроется окно, где будет отображаться процесс установки:



Публикация приложений

Вам не нужно вручную устанавливать приложения на RDS сервер, у JumpServer есть готовые апплеты, которые автоматически установят и опубликуют нужные вам приложения. Существующие апплеты доступны в онлайн портале, где вы их сможете скачать.

Для публикации приложения зайдите во вкладку **"Remote Apps"**, тут вы видите список добавленных апплетов, их статус и кнопки **"Deploy"** и **"Uninstall"** для установки и удаления апплетов с сервера публикаций.



Если у апплетов статус **"Success"** то вы можете добавлять устройства и подключаться к ним с помощью соответствующих приложений. Для подключения к веб-интерфейсам (**HTTP**) можно использовать апплеты **Chrome** или **Firefox**.

Установка OpenSSH на Windows для управления УЗ Windows

Для чего устанавливать OpenSSH на Windows устройства?

OpenSSH на Windows используется для сбора информации о системе, для ротации паролей локальных УЗ Windows и для автоматического создания локальных УЗ. Если нужно подключаться по RDP, без управления учетными записями, **OpenSSH устанавливать не нужно.**

Установка OpenSSH

Вам достаточно просто запустить установочный дистрибутив OpenSSH-Win64.msi с правами администратора. Никаких настроек выполнять не нужно.

Для более безопасного подключения можно настроить авторизацию с **помощью приватного ключа.**

Настройка авторизации с помощью приватного ключа

- Настройка аутентификации на основе открытого ключа для Windows

```
ssh-keygen.exe -t rsa
cp %env:USERPROFILE\.ssh\id_rsa.pub %env:USERPROFILE\.ssh\authorized_keys
```

```
notepad C:\ProgramData\ssh\sshd_config
```

```
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

#Port 22
#AddressFamily any
```

```
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey __PROGRAMDATA__/ssh/ssh_host_rsa_key
#HostKey __PROGRAMDATA__/ssh/ssh_host_dsa_key
#HostKey __PROGRAMDATA__/ssh/ssh_host_ecdsa_key
#HostKey __PROGRAMDATA__/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
StrictModes no
#MaxAuthTries 6
#MaxSessions 10

PubkeyAuthentication yes

# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
AuthorizedKeysFile .ssh/authorized_keys

#AuthorizedPrincipalsFile none

# For this to work you will also need host keys in %programData%/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
#PasswordAuthentication yes
#PermitEmptyPasswords no

# GSSAPI options
#GSSAPIAuthentication no

#AllowAgentForwarding yes
#AllowTcpForwarding yes
#GatewayPorts no
#PermitTTY yes
```

```
#PrintMotd yes
#PrintLastLog yes
#TCPKeepAlive yes
#UseLogin no
#PermitUserEnvironment no
#ClientAliveInterval 0
#ClientAliveCountMax 3
#UseDNS no
#PidFile /var/run/sshd.pid
#MaxStartups 10:30:100
#PermitTunnel no
#ChrootDirectory none
#VersionAddendum none

# no default banner path
#Banner none

# override default of no subsystems
Subsystem sftp sftp-server.exe

# Example of overriding settings on a per-user basis
#Match User anoncvs
# AllowTcpForwarding no
# PermitTTY no
# ForceCommand cvs server

# FCP145974588
#Match Group administrators
# AuthorizedKeysFile __PROGRAMDATA__/ssh/administrators_authorized_keys
```

```
net stop sshd
net start sshd
```

Использование приватного ключа

```
ssh user@ip -i <private_key_absolute_path> (local users)
ssh user@domain@ip -i <private_key_absolute_path> (Domain users)
```

Как установить верные дату и время в JumpServer?

По умолчанию JumpServer устанавливается с часовым поясом Asia/Shanghai, из-за чего в различных журналах системы отображается неверное время.

Решение:

1. Установите актуальное время и часовой пояс на ОС, где установлен JumpServer

Важно, чтобы верно было не только дата, время но и часовой пояс.

Например, для Москвы в 18:11 Ubuntu Linux отвечает так:

```
root@js4ee:/home/serg# date
Wed Dec  4 06:11:33 PM MSK 2024
```

Если будет, например, Wed Dec 4 06:11:33 PM UTC 2024 - часовой пояс указан неверно.

Воспользуйтесь навыками администрирования Linux или обратитесь к администратору, чтобы настроить правильное время, часовой пояс и автоматическую синхронизацию сервера со службой точного времени (это важно в том числе для корректной работы безопасных подключений и сертификатов)

2. Откройте файл конфигурации:

```
# nano /opt/jumpserver/config/config.txt
```

3. В конце файла укажите ваш часовой пояс, например, для Москвы:

```
# The current running version number of JumpServer, automatically generated after installation and upgrade
#
TZ=Europe/Moscow
TIME_ZONE=Europe/Moscow
```

Внимание: если в файле только один параметр TZ, и нет параметра TIME_ZONE, то его следует добавить, чтобы было именно так как указано выше.

4. После сохранения файла конфигурации, перезапустите JumpServer:

```
# jmsctl restart
```

Настройка Panda для публикации приложений

JumpServer поддерживает использование как Windows Server, так и Linux в качестве машины для публикации приложений, например для публикации браузеров Chrome и Firefox для HTTP сессий и различных клиентов для работы с СУБД.

Типы публикации приложений:

Microsoft RemoteApp: способ публикации приложений на базе Windows Server, обеспечивающий максимальную плавность работы. Требуется дополнительная настройка Windows Server и приобретения Microsoft RDS CALs.

Panda (Виртуальное приложение): способ публикации приложений на базе Linux, характеризующийся средней плавностью работы, хорошей совместимостью и поддержкой таких операционных систем, как CentOS, RedHat, Kylin и openEuler.

Настройка Panda для публикации приложений.

Принцип работы:

Машина для публикации приложений на базе операционной системы Linux использует контейнерную технологию, которая изолирует приложение в независимой среде выполнения. С помощью компонента Panda, предоставляемого JumpServer, осуществляется управление виртуальными приложениями.

Процесс выглядит следующим образом:

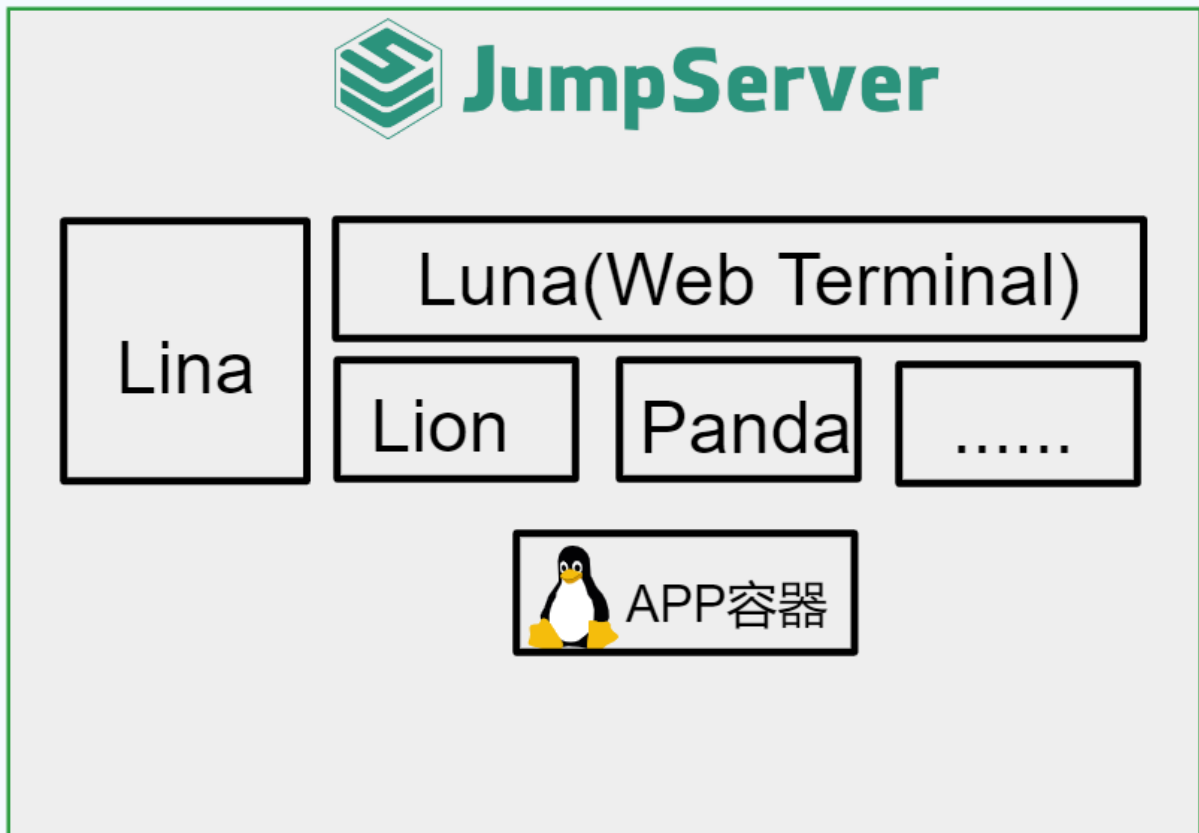
1. Пользователь получает доступ к Web Terminal JumpServer и подключается выбранному виртуальному приложению.
2. Компонент Panda создает GUI-контейнер на базе VNC и передает информацию о подключении VNC компоненту Lion.
3. Компонент Lion подключается к данному контейнеру.

Схемы развертывания

Схема 1: All in One

Использование сервера, на котором развернут JumpServer, в качестве машины для публикации виртуальных приложений.

192.168.127.162



1. Настройка основного конфигурационного файла

Откройте основной конфигурационный файл JumpServer.

```
nano /opt/jumpserver/config/config.txt
```

И добавьте в него следующие параметры

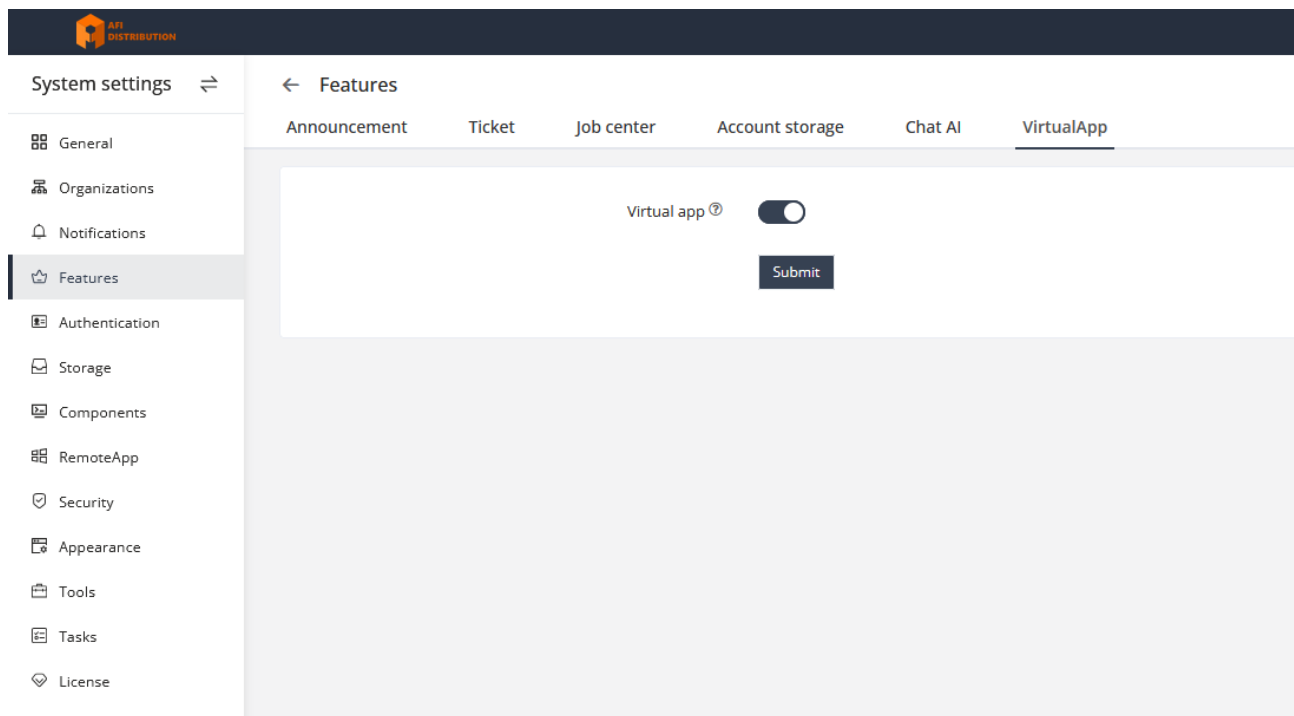
```
# Включение компонента Panda
PANDA_ENABLED=1
# Включение виртуальных приложений в ядре
VIRTUAL_APP_ENABLED=1
# IP-адрес хоста Panda (IP JumpServer)
PANDA_HOST_IP=192.168.127.162
# URL-адрес для компонента Lion к Panda
PANDA_HOST=http://panda:9001
```

Перезапустите сервис JumpServer, чтобы применить изменения.

```
[root@localhost ~]# jmsctl restart
```

2. Включение функции виртуальных приложений

В консоли управления JumpServer перейдите в **System Settings** → **Features** → **VirtualApp** и активируйте функцию виртуальных приложений.

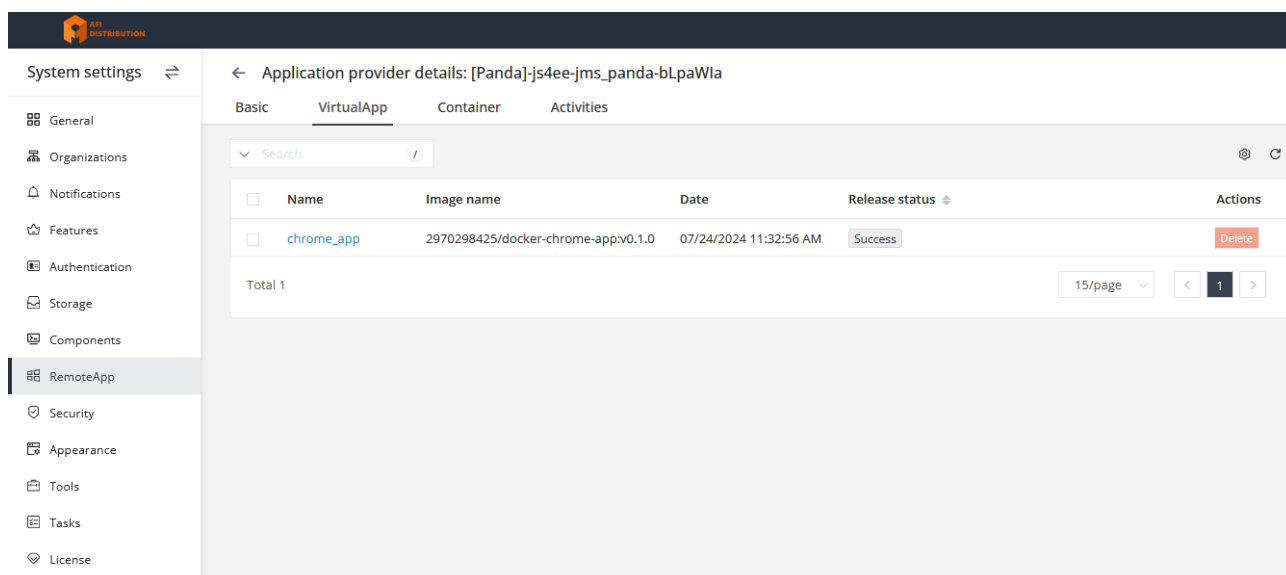


3. Загрузка виртуальных приложений

Загрузите виртуальные приложения локально. В настоящее время поддерживаются браузер **Chrome**, клиент базы данных **Dbeaver**. **Дистрибутивы этих приложений доступны на портале вендора, приложения для Panda находятся в разделе Virtual App, остальные апплеты только для RemoteApp(RDS).**

В консоли управления **JumpServer** перейдите в **System Settings** → **RemoteApps** и загрузите виртуальные приложения в раздел **VirtualApp**.

После короткого ожидания приложение будет автоматически развернуто на машине для публикации приложений. В консоли управления **JumpServer** в разделе **System Settings** → **RemoteApps** → **Application Providers** → **VirtualApp** можно увидеть успешное развертывание приложения.

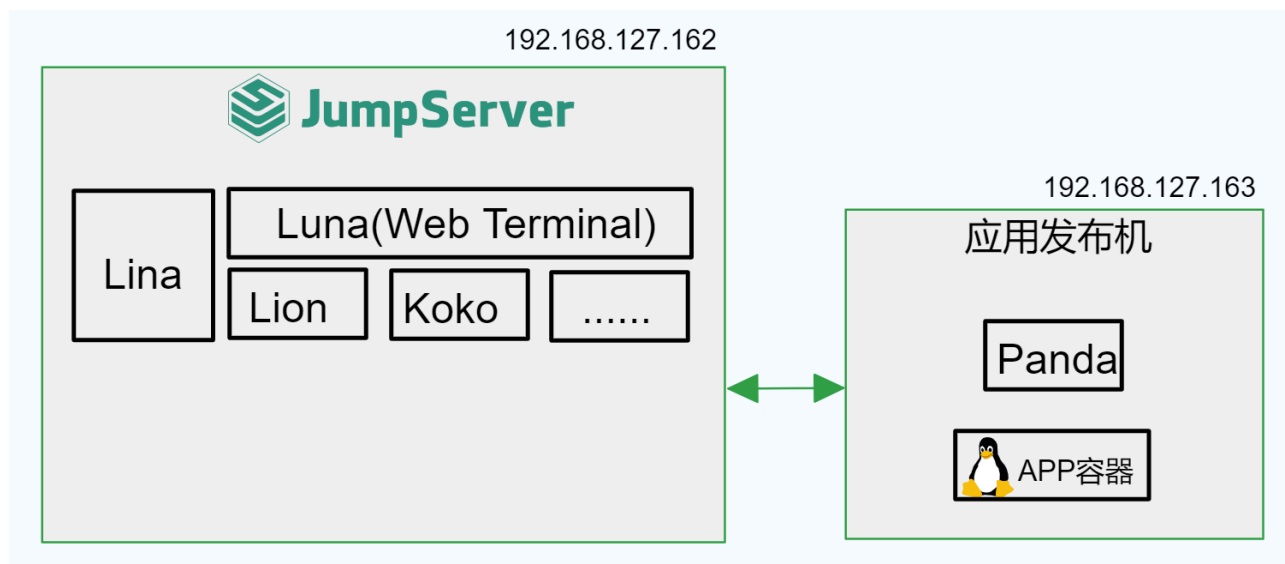


4. Использование виртуальных приложений

Подключитесь к активам, используя виртуальные приложения.

Примечание: В этот момент в сервисе JumpServer будет запущен контейнер виртуального приложения: **2970298425/docker-chrome-app:v0.1.0** (Примечание: этот контейнер весит около 1.3GB, требуется загрузка через интернет. В локальной сети можно загрузить его вручную).

Схема 2: Panda на другом сервере



1. Настройка основного конфигурационного файла

Откройте основной конфигурационный файл JumpServer.

```
nano /opt/jumpserver/config/config.txt
```

И добавьте в него следующие параметры

```
# Включение компонента Panda
PANDA_ENABLED=0
# IP-адрес Panda для компонента Lion
PANDA_HOST=http://192.168.127.163:9001
```

Перезапустите сервис JumpServer, чтобы применить изменения.

```
[root@localhost ~]# jmsctl restart
```

2. Установка Panda на сторонней машине

Распакуйте установочный пакет JumpServer на машине для публикации, установите Docker и Docker Compose, загрузите образ.

```
[root@panda ~]# tar xzvf jumpserver-offline-release-v3.10.6-amd64.tar.gz -C /opt
```

Установите Docker и Docker Compose:

```
[root@panda ~]# cd /opt/jumpserver-offline-release-v3.10.6-amd64/scripts
[root@panda scripts]# ./2_install_docker.sh
```

Загрузите образ Panda:

```
[root@panda scripts]# cd images
[root@panda images]# docker load -i panda:v3.10.6.tar
```

Создайте docker-compose для Panda:

```
[root@panda ~]# mkdir -p /data/jumpserver/panda/data
[root@panda ~]# mkdir -p panda
[root@panda ~]# cd panda
[root@panda panda]# cat docker-compose.yaml
version: '2.4'

services:
  panda:
    image: registry.fit2cloud.com/jumpserver/panda:v3.10.6
    container_name: jms_panda
    hostname: jms_panda
    ulimits:
      core: 0
    restart: always
    ports:
      - 9001:9001
    tty: true
    environment:
      - BOOTSTRAP_TOKEN=YmEyNTRkNTYtNDIyMi02OTJm
      - CORE_HOST=http://192.168.127.162
      - NAME=panda
```

```
- PANDA_HOST_IP=192.168.127.163
volumes:
- /data/jumpserver/panda/data:/opt/panda/data
- /var/run/docker.sock:/var/run/docker.sock:z
healthcheck:
test: "curl -fsL http://localhost:9001/panda/health/ > /dev/null"
interval: 10s
timeout: 5s
retries: 3
start_period: 10s
```

BOOTSTRAP_TOKEN берется из файла конфигурации JumpServer: /opt/jumpserver/config/config.txt

CORE_HOST - адрес вашего JumpServer

PANDA_HOST_IP - IP адрес Panda

Запустите контейнер Panda:

```
docker-compose up -d
```

3. Включение функции виртуальных приложений

Повторите шаги из раздела All in One

4. Загрузка виртуальных приложений

Повторите шаги из раздела All in One

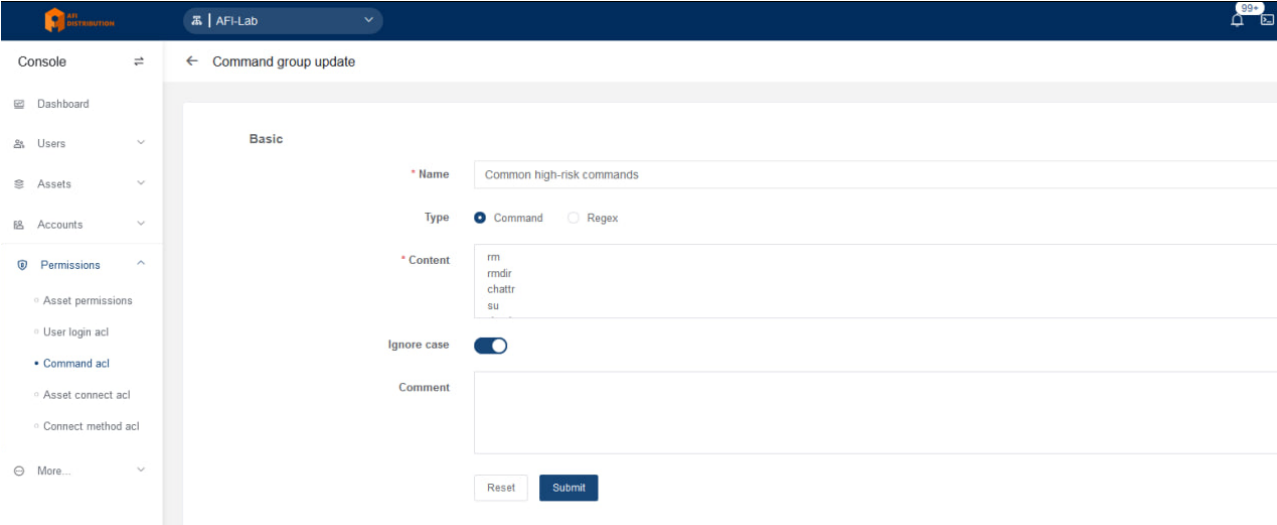
5. Использование виртуальных приложений

Повторите шаги из раздела All in One

Администрирование системы

Настройка блокировки команд SSH и запросов СУБД

1. Перейдите в раздел "**Console - Permissions - Command Acl**", откройте вкладку "**Command Group**".
2. Нажмите кнопку "**Create**", введите название списка, например "**Common high-risk commands**" и заполните список нужных команд или регулярных выражений (см. скриншот) и сохраните, нажав кнопку "**Submit**".



The screenshot shows the 'Command group update' interface. On the left is a sidebar with navigation links: Console, Dashboard, Users, Assets, Accounts, Permissions (selected), and More... The Permissions section is expanded, showing sub-items like Asset permissions, User login acl, Command acl (selected), Asset connect acl, and Connect method acl. The main area is titled 'Command group update' and contains a 'Basic' tab. The form fields are: Name (Common high-risk commands), Type (Command selected, Regex unselected), Content (rm, rmdir, chattr, su), Ignore case (toggle on), and Comment (empty). At the bottom are 'Reset' and 'Submit' buttons.

3. Перейдите во вкладку "**Command filter**", нажмите "**Create**" для создания фильтра.
4. Настройка фильтра включает следующие параметры:
 - **Priority**: приоритет фильтра, всегда выполняется действие того фильтра, чей приоритет будет выше.
 - **User**: пользователи JumpServer, для которых будет работать фильтр
 - **Asset**: целевые системы, подключение к которым будет контролироваться фильтром
 - **Account**: учетные записи на целевых системах, которые будут контролироваться фильтром
 - **Command Group**: группы команд, которые будут блокироваться
 - **Action**: действие фильтра: **Reject** - заблокировать команду, **Accept** - выполнить команду, **Review** - отправить команду на согласование указанному сотруднику, **Warning** - предупредить о выполнении команды указанного сотрудника.

AFI-Platform

AFI-Lab

99+

Console

Dashboard

Users

Assets

Accounts

Permissions

Asset permissions

User login acl

Command acl

Asset connect acl

Connect method acl

More...

Basic

Name

Common high-risk commands

Priority

50

1-100, the lower the value will be match first

User

User

AllUsers

SpecificUsers

Select By Attribute

Полков Сергей(sergey)

Наталья Орлова(nlo)

Asset

Asset

AllAsset

SpecificAsset

Select By Attribute

JumpServer22(10.10.53.22)

Account

Account

All accounts

Specify account

Command group

Command group

Common high-risk commands

Action

Action

Reject

Accept

Review

Warning

5. Нажмите **"Submit"** для сохранения настроек.

Настройка подключения к целевым системам по HTTP(веб-интерфейсы приложений)

Для подключения к целевым системам по HTTP вам необходимо:

Настроить публикацию браузера через Panda (сервер публикации приложений на базе Linux)

или

Настроить публикацию браузера через RDS(RemoteApp).

Создание устройства типа "Вебсайт"

1. Зайдите в раздел **"Console - Assets"** , нажмите кнопку **"Create"** и выберите тип целевой системы - **Website**

The screenshot shows the JuMPServer console interface. On the left is a sidebar with navigation links: Console, Dashboard, Users, Assets (selected), Domains, Platforms, Accounts, Permissions, and More. The main area is titled 'Basic' and contains the following fields:

- Name:** Script Simple Form Authentication
- URL:** https://authenticationtest.com/simpleFormAuth/
- Platform:** Website
- Node:** JFPLab

Below these is the **Selector** section with three radio buttons: **Autofill** (selected), Disabled, and Script. Under Autofill, there are three input fields:

- Username selector:** name=email
- Password selector:** name=password
- Submit selector:** Xpath=/html/body/div/div/div[2]/form/input

Below the Selector section is the **Protocol** section with a dropdown menu set to **http(s)** and a port field set to **443**. A small note below says: "Asset support protocol is limited by platform, click the Settings button to view the protocol settings. If you need to update, please update the platform".

At the bottom is the **Account** section with a link: [Account list](#) and a link: [Update account information in asset details](#).

2. В разделе **"Selector"** нужно указать параметры полей формы, которые **JuMPServer** заполнит автоматически при открытии сессии.

На пример:

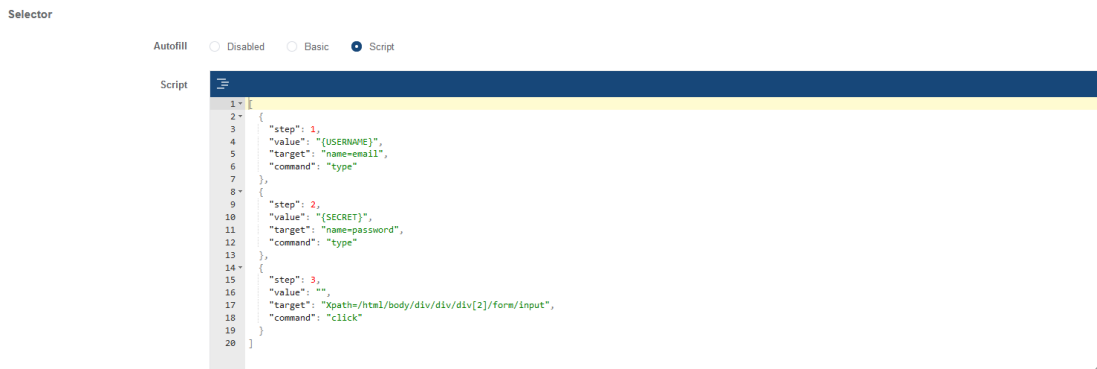
This is a close-up of the Selector configuration fields from the previous screenshot. It shows the **Autofill** radio button selected, and the following fields:

- Username selector:** name=email
- Password selector:** name=password
- Submit selector:** Xpath=/html/body/div/div/div[2]/form/input

При таких настройках, имя пользователя будет введено в HTML элемент с **name="email"**, пароль будет введен в HTML элемент с **name="password"** и затем будет нажата кнопка с **Xpath=/html/body/div/div/div[2]/form/input**

Вы можете посмотреть элементы веб-формы входа в браузере, нажав правую кнопку мышки на поле ввода и выбрав пункт "Исследовать" (для Firefox) или "Просмотреть код" (для Chrome).

Также можно использовать дополнительные настройки и параметры элементов формы входа, для этого переключитесь в режим **Script**:

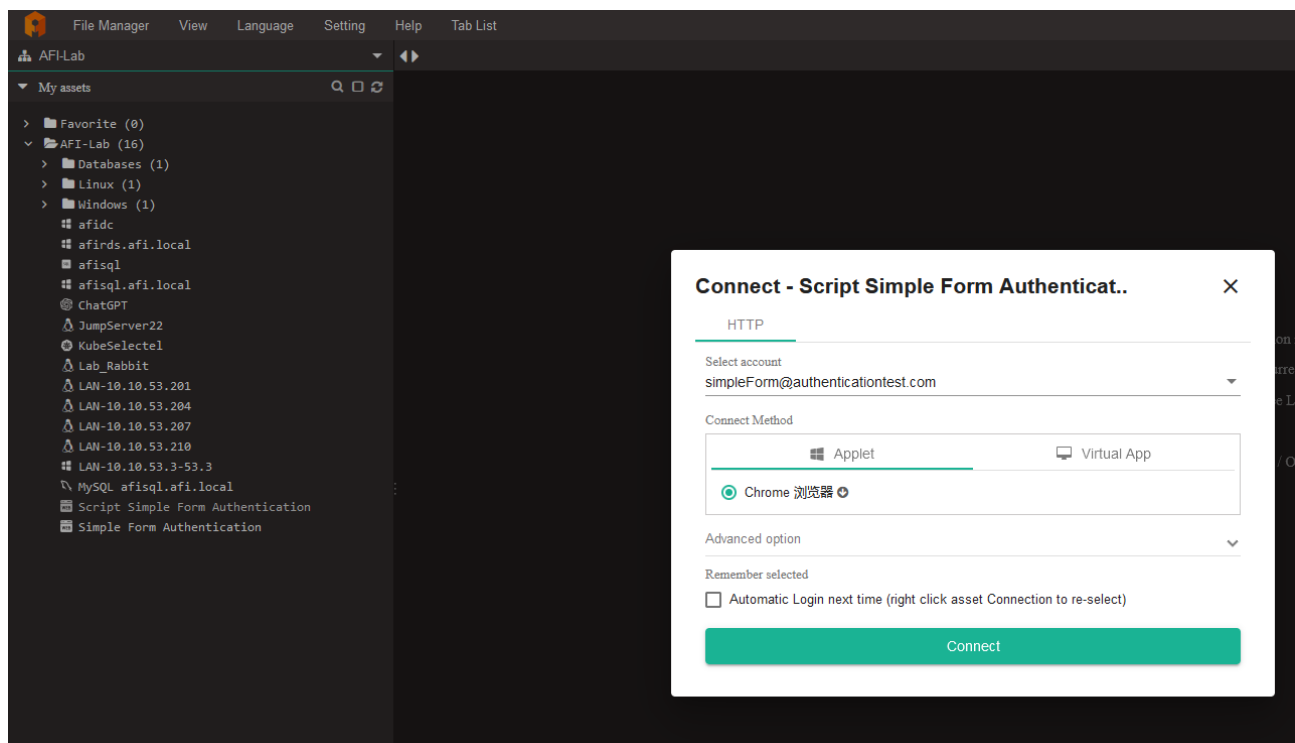


3. в разделе **Account list** нужно добавить учетную запись и пароль, которая будет использоваться при авторизации, аналогично как это делается при других типах подключения.

4. Сохранить настройки, нажав кнопку **"Submit"**.

Подключение к веб-интерфейсам через веб-терминал

Если все настроено верно, при выборе нужного устройства в веб-терминале, вы увидите вариант запуска сессии:

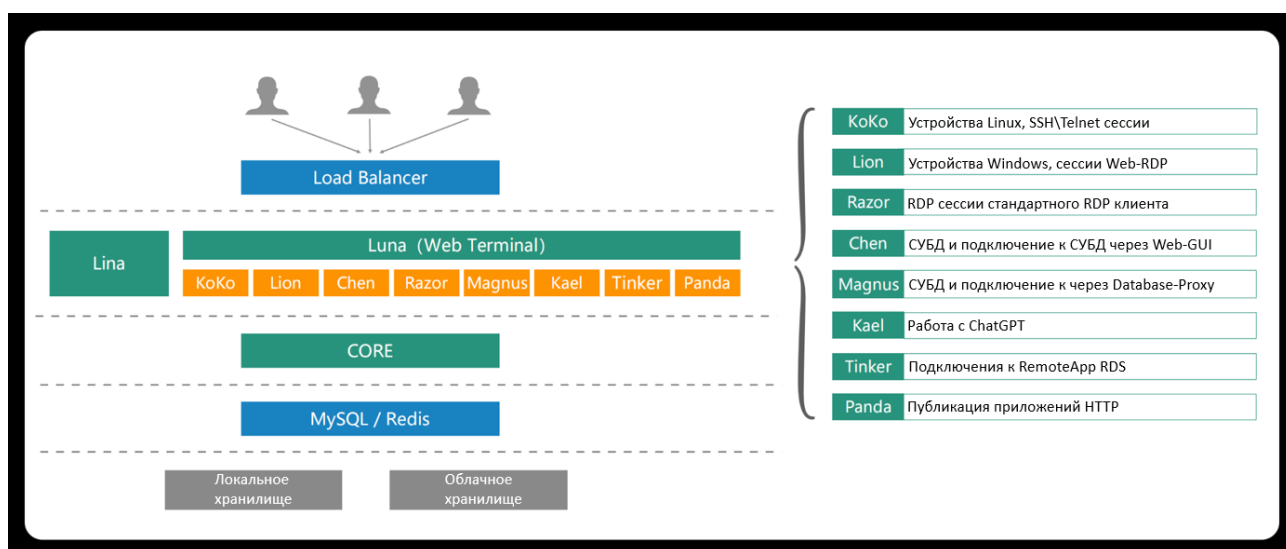


Решение проблем

Решение проблем

Проверка работы контейнеров и журналы ошибок

JumpServer устанавливается в виде набора контейнеров Docker, выполняющих различные функции.



Для проверки состояния контейнеров достаточно авторизоваться на сервер с установленным JumpServer и ввести команду

```
# docker ps -a
```

Все контейнеры в списке должны быть со статусом **Healthy**

Список контейнеров

```
jms_panda  
jms_magnus  
jms_celery  
jms_chen  
jms_koko  
jms_lion  
jms_razor  
jms_video  
jms_web  
jms_redis  
jms_kael  
jms_xrdp  
jms_mysql
```

jms_core

Для просмотра журнала ошибок того или иного модуля, введите команду

```
docker logs -f #ИМЯ_КОНТЕЙНЕРА --tail 200
```

Например:

Журнал ошибок веб-интерфейса:

```
docker logs -f jms_web --tail 200
```

Журнал ошибок Panda:

```
docker logs -f jms_panda --tail 200
```

Решение проблем с публикацией приложений RemoteApp

В этой статье я подробно опишу, как взаимодействуют JumpServer, RDS Server и Tinker, чтобы можно было выяснить, в чем проблема.

Описание процесса интеграции RDS и JumpServer.

1. JumpServer подключается по SSH к серверу RDS:

- устанавливает там **сервис Tinker**, которому сообщает адрес JS (параметр Core API)
- создает локальные сервисные УЗ вида js_* и jms_* на RDS
- добавляет созданные УЗ в группу "Remote Desktop Users"

Важно: группа "Remote Desktop Users" может называться по-другому в Windows, название может зависеть от языка ОС, если эти УЗ не присутствуют в группе, нужно будет добавить их вручную.

2. Tinker связывается с JumpServer по заданному адресу (параметр Core API)

- Tinker сообщает, что он работает, и статус RDS становится "online".
- Загружает дистрибутивы выбранных апплетов, например, Chrome и DBeaver загружаются с сервера JS по тому же адресу, указанному в Core API. Другие апплеты он может попытаться загрузить из интернета.
- Tinker устанавливает загруженные апплеты.

Описание Tinker:

Tinker - вспомогательное приложение для интеграции с JumpServer, участвует в поддержке связи между RDS и JS, устанавливает нужные для публикации приложения(апплеты), а также управляет публикацией приложений RemoteApp при запуске сессий.

По умолчанию устанавливается в папку **C:\Users\[ACCOUNT]\AppData\Local\Programs\Tinker** ([ACCOUNT] - это УЗ, которая использовалась для интеграции JS и RDS)

Устанавливает сервис JumpServer Tinker Service - он всегда должен быть запущен

Подробный лог работы и ошибок Tinker доступен в папке

C:\Users\[ACCOUNT]\AppData\Local\Programs\Tinker\data\logs

Апплеты Tinker

Апплеты - это набор из приложения и скриптов автоматизации для управления приложением, по умолчанию апплеты устанавливаются в папку

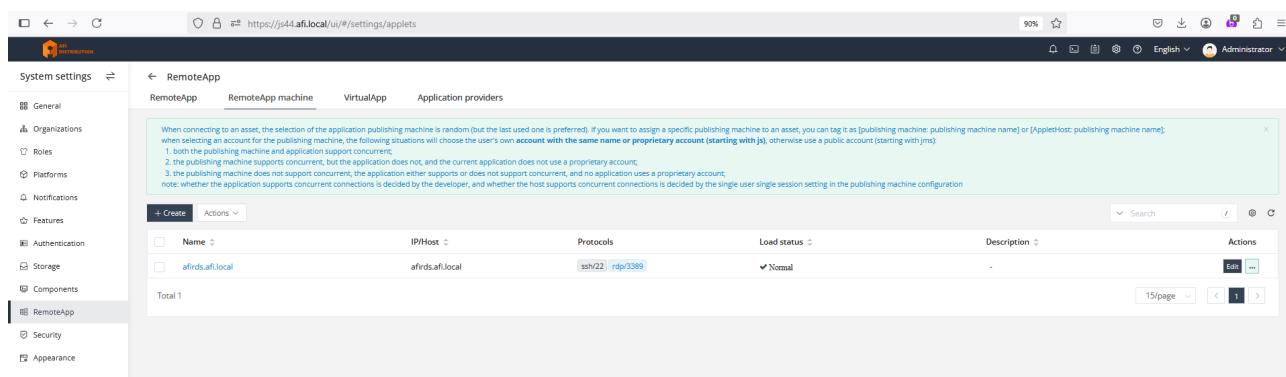
C:\Program Files\JumpServer

Именно в этой папке находятся запускаемые приложения и python скрипты с параметрами запуска, вы их можете изменять, если нужно.

Итоговый чек-лист:

- На Windows RDS должен быть установлен OpenSSH, 22 порт должен быть открыт на Windows файрволле.
- УЗ для интеграции с RDS должно иметь права администратора RDS.
- В IP/Host должен быть указан IP адрес RDS сервера или его DNS имя, которое резолвится с JumpServer.
- В Core API должен быть указан URL JumpServer, который доступен с RDS Server.
- На RDS должны быть созданы УЗ вида JS_XX и JMS_XX.
- УЗ вида JS_XX и JMS_XX на RDS должны находиться в группе пользователей удаленного рабочего стола на RDS сервере.
- Сервис JumpServer Tinker на RDS сервере должен быть запущен.
- Статус RDS сервера в интерфейсе JumpServer: Normal.
- Статус апплетов в свойствах RDS сервера в интерфейсе JumpServer: Success.

Скриншоты стенда JS:





Basic


* Name

* IP/Host

Protocol

Protocols  

 The protocols supported by the assets are restricted by the platform. Click the settings button to view the protocol settings. If updates are required, please update the platform

Account


Accounts [Update account info in asset details](#)

Using same account ☐


Auto create accounts ☒

Accounts create amount

Automation

Core API 

Ignore certificate verification ☒

Existing RDS license  ☐

Other

Zone

This domain belongs to the system organization

Active ☒

System settings

RemoteApp

RemoteApp machine

VirtualApp

Application providers

When connecting to an asset, the selection of the application publishing machine is random (but the last used one is preferred). If you want to test when selecting an account for the publishing machine, the following situations will choose the user's own account with the same name or prefix:
1. both the publishing machine and application support concurrent;
2. the publishing machine supports concurrent, but the application does not, and the current application does not use a proprietary account;
3. the publishing machine does not support concurrent, the application either supports or does not support concurrent, and no application user note: whether the application supports concurrent connections is decided by the developer, and whether the host supports concurrent connections is decided by the platform.

+ Create

Actions

Name	IP/Host	Protocols
afirds.afi.local	afirds.afi.local	ssh/22 rdp/3389

Total 1

RemoteApp machine: afirds.afi.local

Basic

Accounts

RemoteApp

Deploy publishing machine

Activities

RECENT (7 DAYS)

Recently discovered | Recently logged in | Recently modified | Recently changed password | Recent password change failed

RISKY ACCOUNT

No login for long time | Add account | Weak password | Empty password | Long time password

ACCOUNT TYPE

All | Host | Database | Cloud | Device | Web | Other

+ Create

Account templates

Actions

Name	Username	Asset	Connect	Actions
jms_5V0Ddjar	jms_5V0Ddjar	afirds.afi.local		
jms_62tQmV1	jms_62tQmV1	afirds.afi.local		
jms_D66t25p5	jms_D66t25p5	afirds.afi.local		
jms_EKj8v3N	jms_EKj8v3N	afirds.afi.local		
jms_Eat7Cxq	jms_Eat7Cxq	afirds.afi.local		
jms_JRXD9cy	jms_JRXD9cy	afirds.afi.local		
jms_OS8h7ro4	jms_OS8h7ro4	afirds.afi.local		
jms_YWj71Nn2	jms_YWj71Nn2	afirds.afi.local		
jms_ppOfstE3m	jms_ppOfstE3m	afirds.afi.local		
jms_wQD00Rf	jms_wQD00Rf	afirds.afi.local		
js_ACUNETIX	js_ACUNETIX	afirds.afi.local		
js_ACUNETIXS	js_ACUNETIXS	afirds.afi.local		
js_AFIDC	js_AFIDC	afirds.afi.local		
js_AFIDCS	js_AFIDCS	afirds.afi.local		



Дополнительные ВОЗМОЖНОСТИ

Дополнительные возможности

Структура апплетов RemoteApp и создание своего апплета

Что такой апплет?

Апплет представляет из себя набор файлов, которые описывают процесс установки и запуска того или иного приложения на Microsoft RDS через RemoteApp. Это нужно для того, чтобы JumpServer мог запускать сессию доступа к этому приложению, автоматически авторизуясь в нем и скрывая параметры авторизации от пользователя.

Структура апплета

Каждый апплет обязательно состоит из этих файлов:

```
AppletName
├─ i18n.yml
├─ icon.png
├─ main.py
├─ manifest.yml
└─ setup.yml
```

main.py - скрипт запуска и авторизации в приложении

icon.png - значок апплета

manifest.yml - метаданные, то есть описание апплета

setup.yml - файл с описанием установки

i18n.yml - файл перевода на различные языки

Файл manifest.yml

Пример на базе апплета MySQL Workbench

В файле manifest.yml находится общая информация об апплете и указание его типа и протокола.

```
# (обязательно)
name: mysql_workbench8
display_name: "{{ 'MySQL Workbench' | trans }}"
```

```
comment: "{ { 'A tool for working with MySQL, to execute SQL and design tables' | trans } }"
# (обязательно))
version: 0.1.1
# (обязательно)
exec_type: python
# (обязательно)
author: Eric
# general или web (обязательно)
type: general
update_policy: always
edition: community
# (обязательно)
tags:
  - database
# (обязательно)
protocols:
  - mysqlworkbench

# перевод на другие языки
i18n:
  MySQL Workbench:
    en: MySQL Workbench
    zh: MySQL Workbench
    ja: MySQL Workbench

A tool for working with MySQL, to execute SQL and design tables:
en: A tool for working with MySQL, to execute SQL and design tables
zh: MySQL Workbench
ja: MySQL Workbench
```

Файл setup.yml

Файл `setup.yml` описывает параметры установки апплета на RDS сервер.

```
# тип установки ПО - msi,exe, zip, manual
type: msi
# адрес для загрузки дистрибутива ПО или имя файла, если дистрибутив вместе с апплетом в архиве
source: mysql-workbench-community-8.0.31-winx64.msi
# аргументы запуска установки
arguments:
  - /qn
  - /norestart
# директория для установки
destination: C:\Program Files\MySQL\MySQL Workbench 8.0 CE
# путь и имя исполняемого файла
program: C:\Program Files\MySQL\MySQL Workbench 8.0 CE\MySQLWorkbench.exe
md5: d628190252133c06dad399657666974a
```

Скрипт main.py

main.py - основной скрипт апплета

Запуск приложения осуществляется вызовом команды:

```
python main.py base64_json_data
```

То есть запускается скрипт main.py и в него передаются параметры запуска, структура **base64_json_data** выглядит примерно так:

```
{
  "app_name": "mysql_workbench8",
  "protocol": "mysql",
  "user": {
    "id": "2647CA35-5CAD-4DDF-8A88-6BD88F39BB30",
    "name": "Administrator",
    "username": "admin"
  },
  "asset": {
    "asset_id": "46EE5F50-F1C1-468C-97EE-560E3436754C",
    "asset_name": "test_mysql",
    "address": "192.168.1.1",
    "protocols": [
      {
        "id": 2,
        "name": "mysql",
        "port": 3306
      }
    ]
  },
  "account": {
    "account_id": "9D5585DE-5132-458C-AABE-89A83C112A83",
    "username": "root",
    "secret": "test"
  },
  "platform": {
    "charset": "UTF-8"
  }
}
```

Содержимое main.py

```
import sys

from common import (block_input, unblock_input) # Импорт функций для блокировки/
разблокировки ввода
from common import convert_base64_to_dict # Импорт функции для конвертации Base64 строки
в словарь(массив)
from app import AppletApplication # Импорт основного приложения
```

```

def main():
    base64_str = sys.argv[1] # Получаем строку Base64 из аргументов командной строки
    data = convert_base64_to_dict(base64_str) # Конвертируем Base64 строку в словарь
    # словарь data содержит все параметры запуска приложения: учетную запись, имя сервера,
    имя базы данных и тд в зависимости от типа приложения
    applet_app = AppletApplication(**data) # Передаем данные словаря в функцию запуска
    приложения
    block_input() # Блокируем ввод пользователя
    applet_app.run() # Запускаем приложение
    unblock_input() # Разблокируем ввод пользователя
    applet_app.wait() # Ожидаем завершения работы приложения

if __name__ == '__main__':
    try:
        main() # Запускаем основную функцию
    except Exception as e:
        print(e) # Выводим ошибку, если она возникла

```

Содержимое app.py

App.py обычно содержит весь основной код запуска приложения с нужными параметрами, поэтому это самая важная и самая сложная часть при разработке нового апплета. Проще всего не создавать его с нуля, а взять за основу один из скриптов апплетов, которые похожи по структуре\типу приложения на новый создаваемый апплет.

```

import sys # Импортирует модуль sys для работы с системными функциями

if sys.platform == 'win32': # Проверяет, если операционная система — Windows
    from pywinauto import Application # Импортирует библиотеку для автоматизации оконных
    приложений на Windows
    from pywinauto.controls.uia_controls import (ButtonWrapper, EditWrapper, MenuItemWrapper,
        MenuWrapper, ComboBoxWrapper, ToolbarWrapper)
    # Импортирует различные элементы управления для взаимодействия с интерфейсом
    приложения

from common import (BaseApplication, wait_pid, ) # Импортирует базовое приложение и
функцию ожидания процесса

_default_path = r"C:\Program Files\MySQL\MySQL Workbench 8.0 CE\MySQLWorkbench.exe"
# Определяет путь к приложению MySQL Workbench по умолчанию

class AppletApplication(BaseApplication): # Определяет класс приложения, наследующий от
BaseApplication

    def __init__(self, *args, **kwargs): # Инициализирует приложение
        super().__init__(*args, **kwargs) # Вызов конструктора родительского класса

```

```

self.path = _default_path # Устанавливает путь к приложению
self.username = self.account.username # Получает имя пользователя из учетной записи
self.password = self.account.secret # Получает пароль из учетной записи
self.host = self.asset.address # Получает адрес хоста из параметров актива
self.port = self.asset.get_protocol_port(self.protocol) # Получает порт по протоколу
self.db = self.asset.spec_info.db_name # Получает название базы данных
self.pid = None # Переменная для хранения ID процесса приложения
self.app = None # Переменная для хранения объекта приложения

def run(self): # Метод для запуска приложения
    app = Application(backend='uia') # Создает объект приложения для взаимодействия через
    UI Automation
    app.start(self.path) # Запускает приложение по указанному пути
    self.pid = app.process # Сохраняет ID процесса приложения
    if not all([self.username, self.password, self.host]): # Проверяет, заполнены ли необходимые
    параметры
        print(f'参数不完整') # Выводит сообщение об ошибке на китайском ("Не хватает
    обязательных параметров")
        return

    # Доступ к главному окну MySQL Workbench и меню "Database"
    menubar = app.window(title="MySQL Workbench", auto_id="MainForm",
    control_type="Window") \
        .child_window(title="Database", control_type="MenuItem")
    menubar.wait('ready', timeout=10, retry_interval=5) # Ожидает готовности меню
    MenuItemWrapper(menubar.element_info).select() # Открывает меню "Database"

    # Выбирает пункт меню "Connect to Database"
    cdb = menubar.child_window(title="Connect to Database", control_type="MenuItem")
    cdb.wait('ready', timeout=10, retry_interval=5) # Ожидает готовности элемента
    MenuItemWrapper(cdb.element_info).click_input() # Нажимает на элемент "Connect to
    Database"

    # Вводит хост
    host_ele = app.top_window().child_window(title="Host Name", auto_id="Host Name",
    control_type="Edit")
    EditWrapper(host_ele.element_info).set_edit_text(self.host) # Вводит значение хоста

    # Вводит порт
    port_ele = app.top_window().child_window(title="Port", auto_id="Port", control_type="Edit")
    EditWrapper(port_ele.element_info).set_edit_text(self.port) # Вводит значение порта

    # Вводит имя пользователя
    user_ele = app.top_window().child_window(title="User Name", auto_id="User Name",
    control_type="Edit")
    EditWrapper(user_ele.element_info).set_edit_text(self.username) # Вводит имя пользователя

    # Вводит имя базы данных
    db_ele = app.top_window().child_window(title="Default Schema", auto_id="Default Schema",

```

```
control_type="Edit")
    EditWrapper(db_ele.element_info).set_edit_text(self.db) # Вводит имя базы данных

    # Нажимает на кнопку "OK" для подтверждения соединения
    ok_ele = app.top_window().child_window(title="Connection", auto_id="Connection",
control_type="Window") \
        .child_window(title="OK", control_type="Button")
    ButtonWrapper(ok_ele.element_info).click() # Нажимает на кнопку "OK"

    # Вводит пароль
    password_ele = app.top_window().child_window(title="Password", auto_id="Password",
control_type="Edit")
    password_ele.wait('ready', timeout=10, retry_interval=5) # Ожидает готовности поля для
ввода пароля
    EditWrapper(password_ele.element_info).set_edit_text(self.password) # Вводит пароль

    # Нажимает на кнопку "OK" для завершения подключения
    ok_ele = app.top_window().child_window(title="Button Bar", auto_id="Button Bar",
control_type="Pane") \
        .child_window(title="OK", control_type="Button")
    ButtonWrapper(ok_ele.element_info).click() # Нажимает на кнопку "OK"

    self.app = app # Сохраняет объект приложения для дальнейшего использования

def wait(self): # Метод для ожидания завершения работы приложения
    wait_pid(self.pid) # Ожидает завершения процесса с определенным ID (pid)
```


Дополнительные возможности

Разработка собственных приложений для Panda

Panda - встроенный механизм запуска приложений в изолированных контейнерах докера, замена RemoteApp.

Чаще всего Panda используется для доступа к веб-приложениям, то есть Panda запускает контейнер с Chrome, автоматически заходит на нужный сайт и авторизуется в нем.

По умолчанию доступна публикация **Chrome** и **DBeaver**, но есть возможность создавать и собственные приложения для публикации в Panda.

Структура апплета Panda на примере SQL Developer:

Виртуальное приложение — это Linux-контейнер с VNC. Panda передаёт переменные окружения с данными авторизации (JMS_TOKEN).

Пример структуры:

```
Sql Developer/  
├─ docker-compose.yml  
├─ Dockerfile  
├─ entrypoint.sh  
├─ sqldeveloper-23.1.1.345.2114-no-jre.zip  
├─ .config/dconf/user  
├─ app/bin/start-vnc.sh  
├─ app/bin/start-xvfb.sh  
└─ etc/supervisor/app.conf
```

docker-compose:

```
services:  
  sql_developer:  
    build:  
      context: .  
      dockerfile: Dockerfile  
    ports:  
      - "6800:5900"
```

manifest.yml:

```
name: Sql_Developer_23  
display_name: Sql Developer
```

comment: SQL Developer — мощное IDE для Oracle.
version: 0.1
author: JumpServer Team
type: panda
image_name: nickyang00/app_sqldeveloper_vapp:v0.1.0
image_protocol: vnc
image_port: 5900
tags: [database]
protocols: [oracle]

Исходный код Dbeaver-app для Panda доступен по [этой ссылке](#).

Исходный код Chrome-App для Panda доступен по [этой ссылке](#).

Руководство по проведению пилотного проекта

Здесь собраны ссылки на основные настройки, которые помогут начать пилотный проект или внедрение системы.

Установка JumpServer:

[Установка JumpServer Enterprise Edition](#)

[Описание и инструкция по установке HA кластера](#)

Системные настройки:

[Интеграция с Active Directory](#)

[Включение двухфакторной авторизации \(TOTP\)](#)

[Настройка Panda для публикации приложений](#)

[Настройка RDS \(RemoteApp\) для публикации приложений](#)

Работа с системой:

[Добавление устройств и учетных записей, политик доступа](#)

[Настройка подключения к целевым системам по HTTP\(веб-интерфейсы приложений\)](#)

[Настройка блокировки команд SSH и запросов СУБД](#)