Санкт-Петербургское государственное унитарное предприятие
«Санкт-Петербургский информационно-аналитический центр»
(СПб ГУП «СПб ИАЦ»)

ПРОГРАММА ДЛЯ ЭВМ

ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ ПЛАТФОРМА «ЮНИВЕРСАЛ»

ИНСТРУКЦИЯ ПО РАЗВЕРТЫВАНИЮ
На 14 листах

Санкт-Петербург
2021

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ И СОКРАЩЕНИЯ

В документе применены следующие сокращения:

Сокращение (обозначение)	Значение сокращения (обозначения)
1	2
БД	База данных
ОПО	Общее программное обеспечение
ОС	Операционная система
ПО	Программное обеспечение
СПО	Специальное программное обеспечение
СУБД	Система управления базами данных

1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ ОБЕСПЕЧЕНИИ

1.1 Назначение программного обеспечения

Информационно-аналитическая платформа «Юниверсал» предназначена для автоматизации информационно-аналитического обеспечения процессов принятия управленческих решений главой региона, Правительством и должностными лицами исполнительных органов государственной власти региона, повышения эффективности информационного взаимодействия органов государственного, регионального и муниципального управления, а также информационного обеспечения государственной автоматизированной информационной системы «Управление» сведениями о социально-экономическом развитии и жизнедеятельности региона. Является частью СПО Информационно-аналитической системы Ситуационного центра Главы региона (ИАС СЦ).

1.2 Описание дистрибутива программного обеспечения

Информационно-аналитическая платформа «Юниверсал» включает программные компоненты, представленные в таблице 1.

Таблица 1– Программные компоненты информационно-аналитической платформы «Юниверсал»

Техническое наименование программного компонента
1
data-hub
data-receiver
emiss-service
incident-management-integration
incident-service
calendar-service
task-service
calc-service
integration-journal
auth-server
report-service
library-service
platform
citypanel
data-gateway
gasu-integration
crypto-service
smev2-client
smev3-client

Для развертывания ПО «Информационно-аналитическая платформа «Юниверсал» необходимо воспользоваться дистрибутивом ПО. В архиве дистрибутива ПО содержатся следующие директории:

ansible_playbook - директория с набором ролей, инвентори и плейбуков ansible для подготовки серверов рабочей среды;

docker_images - директория, которая содержит архивы с дистрибутивами и сохраненные docker-образы программных компонентов информационно-аналитической платформы «Юниверсал».

1.3 Минимальный состав технических средств

Программа может функционировать на технических средствах, удовлетворяющих требованиям, перечисленным ниже.

1.3.1 Сервер базы данных PostgreSQL

Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 2 ядер;
объем оперативной памяти – не менее 8 Гб;
дисковая подсистема – не менее 700 Гб;
SSD/HDD 15000 об/мин;
сетевой адаптер – не менее 1000 Мбит.

1.3.2 Сервер управления кластером на ОС Linux

Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 4 ядер;
объем оперативной памяти – не менее 6 Гб;
дисковая подсистема – не менее 50 Гб;
сетевой адаптер – не менее 1000 Мбит.

1.3.3 Сервер приложений на ОС Linux

Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 4 ядер;
объем оперативной памяти – не менее 14 Гб;
дисковая подсистема – не менее 100 Гб;
сетевой адаптер – не менее 1000 Мбит.

1.3.4 Сервер приложений на ОС Linux

Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 4 ядер;
объем оперативной памяти – не менее 14 Гб;
дисковая подсистема – не менее 100 Гб;
сетевой адаптер – не менее 1000 Мбит.

1.3.5 Сервер приложений на ОС Linux

Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 4 ядер;
объем оперативной памяти – не менее 14 Гб;
дисковая подсистема – не менее 100 Гб;
сетевой адаптер – не менее 1000 Мбит.

1.3.6 Сервер приложений на ОС Linux

Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 4 ядер;
объем оперативной памяти – не менее 14 Гб;
дисковая подсистема – не менее 100 Гб;
сетевой адаптер – не менее 1000 Мбит.

1.3.7 Сервер доставки сообщений кластера Kafka
Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 4 ядер;
объем оперативной памяти – не менее 19 Гб;
дисковая подсистема – не менее 500 Гб;
SSD/HDD 15000 об/мин;
сетевой адаптер – не менее 1000 Мбит.
1.3.8 Сервер доставки сообщений кластера Kafka
Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 4 ядер;
объем оперативной памяти – не менее 19 Гб;
дисковая подсистема – не менее 500 Гб;
SSD/HDD 15000 об/мин;
сетевой адаптер – не менее 1000 Мбит.
1.3.9 Сервер доставки сообщений кластера Kafka
Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 4 ядер;
объем оперативной памяти – не менее 19 Гб;
дисковая подсистема – не менее 500 Гб;
SSD/HDD 15000 об/мин;
сетевой адаптер – не менее 1000 Мбит.
1.3.10 Сервер репозиториев компонентов
Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 2 ядер;
объем оперативной памяти – не менее 4 Гб;
дисковая подсистема – не менее 250 Гб;
сетевой адаптер – не менее 1000 Мбит.
1.3.11 Сервер репозиториев кода
Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 2 ядер;
объем оперативной памяти – не менее 6 Гб;
дисковая подсистема – не менее 271 Гб;
сетевой адаптер – не менее 1000 Мбит.
1.3.12 Сервер непрерывной интеграции
Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 2 ядер;
объем оперативной памяти – не менее 8 Гб;
дисковая подсистема – не менее 101 Гб;
сетевой адаптер – не менее 1000 Мбит.
1.3.13 Сервер межведомственного взаимодействия
Требования:
архитектура – x86-64;
процессор – не менее 2.60 ГГц, не менее 1 ядер;
объем оперативной памяти – не менее 3 Гб;
дисковая подсистема – не менее 70 Гб;
сетевой адаптер – не менее 1000 Мбит.

1.3.14 Сервер R

Требования:

архитектура – x86-64;

процессор – не менее 2.60 ГГц, не менее 2 ядер;

объем оперативной памяти – не менее 2 Гб;

дисковая подсистема – не менее 50 Гб;

сетевой адаптер – не менее 1000 Мбит.

1.3.15 Прокси-сервер

Требования:

архитектура – x86-64;

процессор – не менее 2.60 ГГц, не менее 2 ядер;

объем оперативной памяти – не менее 4 Гб;

дисковая подсистема – не менее 50 Гб;

сетевой адаптер – не менее 1000 Мбит.

1.3.16 Сервер данных

Требования:

архитектура – x86-64;

процессор – не менее 2.60 ГГц, не менее 2 ядер;

объем оперативной памяти – не менее 2 Гб;

дисковая подсистема – не менее 521 Гб;

сетевой адаптер – не менее 1000 Мбит.

1.3.17 Рабочее место пользователя

Требования:

процессор – не менее 1,5 ГГц, не менее 2 ядер;

объем оперативной памяти – не менее 4 Гб;

дисковая подсистема – не менее 100 Гб;

сетевой адаптер – не менее 100 Мбит;

монитор жидкокристаллический с разрешающей способностью не менее 1920*1080;

клавиатура русифицированная;

манипулятор типа «мышь».

1.4 Минимальный состав программных средств

1.4.1 Сервер базы данных PostgreSQL

Требования:

ОС CentOS 7;

СУБД PostgreSQL 11.

1.4.2 Сервер управления кластером на ОС Linux

ОС CentOS 7;

ОПО: Docker 18.09, Kubernetes 1.16.

1.4.3 Сервера приложений на ОС Linux

Требования:

ОС CentOS 7;

ОПО: JDK, GlassFish Server Open Source Edition (Glassfish Enterprise Server), Docker 18.09, Kubernetes 1.16, Open Distro for Elasticsearch 7.6.

1.4.4 Сервера доставки сообщений кластера Kafka

Требования:

ОС CentOS 7;

ОПО: Confluent Platform 5.5 (Apache Kafka).

1.4.5 Сервер репозиториев компонентов

Требования:

ОС CentOS 7;

ОПО: Sonatype Nexus 3.

1.4.6 Сервер репозиториев кода
Требования:
ОС CentOS 7;
ОПО: GitLab.
1.4.7 Сервер непрерывной интеграции
Требования:
ОС CentOS 7;
ОПО: Docker 18.09, Jenkins.
1.4.8 Сервер межведомственного взаимодействия
Требования:
ОС CentOS 7.
ОПО: CryptoPro JCP.
1.4.9 Сервер R
Требования:
ОС CentOS 7;
ОПО: R 3.4, Docker 18.09.
1.4.10 Прокси-сервер
Требования:
ОС CentOS 7;
ОПО: Nginx 1.18.
1.4.11 Сервер данных
Требования:
ОС CentOS 7.
1.4.12 Рабочее место пользователя
Требования:
Веб-браузер: Google Chrome и Mozilla Firefox (последние актуальные версии);
ПО для просмотра документов формата:
Microsoft Office;
PDF;
ОПО для работы с архивными файлами (архиватор).

2 НАСТРОЙКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Для функционирования СПО системы необходимо установить на серверы общее программное обеспечение, перечисленное в разделе 1.4 настоящего документа.

Далее требуется произвести установку и настройку ОПО и СПО системы в следующей последовательности:

ОПО на прокси-сервере;
ОПО на сервере репозиториев компонентов;
ОПО на сервере репозиториев кода;
ОПО на сервере непрерывной интеграции;
ОПО на сервере данных;
ОПО на серверах приложений и управления кластера Kubernetes;
ОПО на сервере межведомственного взаимодействия;
ОПО на серверах СУБД;
ОПО на сервере доставки сообщений кластера Kafka;
СПО на серверах приложений и управления кластера Kubernetes;
СПО на сервере межведомственного взаимодействия;
ОПО на сервере R;
СПО на сервере R.

2.1 Установка программного обеспечения

Для функционирования программы требуется установить на серверы программные средства, приведенные в разделе 1.4.

Для упрощения процедуры установки, перед ее началом рекомендуется заполнить следующую таблицу данными, которые пригодятся в процессе установки.

В таблице 2 приводится шаблон для заполнения.

Таблица 23422- Шаблон с данными, необходимыми в процессе установки

Роль	DNS-имя сервера	IP-адрес основного интерфейса
1	2	3
Сервер репозиториев компонентов	bin-repo.dns.suffix	w.x.y.z
Сервер репозиториев кода	repo.dns.suffix	w.x.y.z
Сервер непрерывной интеграции	ci.dns.suffix	w.x.y.z
Сервер R	r-calc.dns.suffix	w.x.y.z
Сервер СУБД	postgres-db.dns.suffix	w.x.y.z
Сервер межведомственного взаимодействия	smev.dns.suffix	w.x.y.z
Сервер управления кластером приложений	k8s-master.dns.suffix	w.x.y.z
Сервер приложений	k8s-node01.dns.suffix	w.x.y.z
Сервер приложений	k8s-node02.dns.suffix	w.x.y.z
Сервер приложений	k8s-node03.dns.suffix	w.x.y.z
Сервер приложений	k8s-node04.dns.suffix	w.x.y.z
Сервер доставки сообщений кластера Kafka	kafka01.dns.suffix	w.x.y.z
Сервер доставки сообщений кластера Kafka	kafka03.dns.suffix	w.x.y.z
Сервер доставки сообщений кластера Kafka	kafka03.dns.suffix	w.x.y.z
Сервер данных	nfs-data.dns.suffix	w.x.y.z
Прокси-сервер	proxy.dns.suffix	w.x.y.z

Установка программы включает следующие шаги:

1. На всех серверах для выполнения и унификации настройки создать локального пользователя sysop с домашним каталогом и возможностью повышения привилегий в системе с помощью утилиты sudo до уровня пользователя root.
2. Подготовить инфраструктуру DNS: создать имена в соответствии с шаблоном в таблице 2, создать и проверить разрешение внешнего dns-имени и доступы к системе.
3. На сервере непрерывной интеграции переключиться на работу под пользователем sysop.
4. Сгенерировать ключи для асимметричного шифрования подключений с помощью протокола SSH:

```
ssh-keygen
```

5. Скопировать публичный ключ пользователя sysop на остальные сервера, заменив в примере ниже dns-имя сервера БД из таблицы:

```
ssh-copy-id sysop@<ИМЯ СЕРВЕРА>.dns.suffix
```

6. Установить на сервер непрерывной интеграции программу автоконфигурации ansible версии не ниже 2.4 (сайт разработчика: http://ansible.com), и дополнительные зависимости с использованием пакетного менеджера ОС: jinja2 версии не ниже 2.9.6 (сайт разработчика http://jinja.pocoo.org/); netaddr (сайт разработчика https://netaddr.readthedocs.io/en/latest/); pbr версии не ниже 1.6 (сайт разработчика https://docs.openstack.org/pbr/latest. Этот шаг требует повышения привилегий до уровня root.
7. Создать файл настроек с именем .ansible.cfg (точка в начале имени имеет значение) в домашнем каталоге пользователя sysop со следующим содержимым:

```
    [defaults]
remote_user=sysop
become_ask_pass = true
host_key_checking = false
log_path=/tmp/ansible_runner.log
gathering = smart
fact_caching = jsonfile
fact_caching_connection = /tmp

inventory_ignore_extensions = ~, .orig, .bak, .ini, .cfg, .retry, .pyc, .pyo,
.creds

[inventory]
ignore_patterns = artifacts, credentials

[ssh_connection]
ssh_args = -o ControlMaster=auto -o ControlPersist=1800s
pipelining = true
```

8. Скопировать с дистрибутивного диска директорию ansible со всем ее содержимым в домашнюю директорию пользователя sysop.
9. В директории ansible создать текстовый файл с именем production без расширения на основе файла production.example и прописать в нем пароль пользователя sysop и dns-имена из таблицы как приведено в примере ниже:

```
    ---
# file: production
# Промышленный контур

# Сервер СУБД PostgreSQL
[dbservers_pg]
postgres-db.dns.suffix ansible_become_pass='sysop_user_pass'
```

10. В директории ansible/host_vars создать текстовые файлы с именами, совпадающими с dns-именами из таблицы 2, на основе файлов с примерами <ФАЙЛ ПРИМЕРА>.example содержащиеся в директории ansible/host_vars.
11. В файлах директории ansible/group_vars создать текстовые файлы с названиями,

совпадающими с наименованиями групп из файла production в директории ansible на основе файлов с примерами <ФАЙЛ ПРИМЕРА>.example содержащиеся в ansible/group_vars.

12. В качестве локального репозитория бинарных пакетов, специфичных для данной версии программного комплекса, предлагается использовать ПО Nexus Repository OSS (сайт разработчика https://www.sonatype.com/). Для корректной работы скриптов развертывания надо определить значения следующих переменных в файлах prod_engine_servers директории ansible/group_vars: bin_repo_url, bin_repo_user, bin_repo_pass. Данное ОПО располагается на сервере репозиториев компонентов.

Необходимо загрузить docker-образа в приватный регистри bin-repo.dns.suffix. Для загрузки образов необходимо восстановить их из tar-архивов, которые предоставлены в дистрибутиве. Для восстановления образов необходимо локально иметь предустановленное ПО Docker для загрузки образов СПО из архивов. Пример команды загрузки образа:

```
docker load < auth-server_2.5.7.tar
```

Далее необходимо сменить тег, чтобы можно было отправить образ в приватный регистри:

```
docker tag bin-repo.dns.suffix/auth-server:2.5.7 bin-repo.dns.suffix/auth-server:2.5.7
```

После этого необходимо отправить образ в регистри:

```
docker push bin-repo.dns.suffix/auth-server:2.5.7
```

Команды, перечисленные выше, необходимо выполнить для каждого приложения в директории apps.

13. Проверить и при необходимости внести изменения в работу ролей в директории ansible/roles для функционирования в рамках локальной инфраструктуры.

14. Перейти в директорию ansible.

15. Перед запуском системы автонастройки проверить синтаксис и корректность измененных параметров:

```
ansible_playbook production.yml –i production –syntax-check
```

16. После успешного выполнения проверки поочерёдно запустить систему автонастройки на серверах:

```
ansible_playbook production.yml –i production –l <ГРУППА СЕРВЕРОВ>:&<ИМЯ СЕРВЕРА>.dns.suffix
```

17. Произвести автонастройку прокси-сервера.

18. Произвести автонастройку сервера репозиториев компонентов.

19. После успешного выполнения автонастройки необходимо на сервере репозиториев компонентов создать репозитории, необходимые для работы сценариев Ansible при автонастройке серверов и заполнить их необходимыми компонентами в соответствии с документацией разработчиков на продукт: https://help.sonatype.com/repomanager3.

20. Произвести автонастройку сервера репозиториев кода.

21. После успешного выполнения автонастройки необходимо на сервере репозиториев кода произвести его настройку в соответствии с документацией разработчиков на продукт: https://docs.gitlab.com/ce/README.html. Добавить на сервер репозитории кода из директории CI/jenkins_pipeline для организации автоматического развёртывания приложений.

22. Произвести автонастройку сервера непрерывной интеграции.

23. После успешного выполнения автонастройки необходимо на сервере непрерывной интеграции произвести его настройку в соответствии с документацией разработчиков на продукт: https://jenkins.io/doc/. Организовать взаимодействие с сервером репозиториев компонентов и сервером репозиториев кода для организации автоматического развёртывания приложений.

24. Произвести автонастройку сервера данных.

25. После успешного выполнения автонастройки необходимо на сервере данных произвести установку ОПО с помощью сервера непрерывной интеграции.

26. Произвести автонастройку серверов приложений и сервера управления кластером

Kubernetes.

27. Произвести автонастройку сервера межведомственного взаимодействия.

28. Произвести автонастройку сервера СУБД PostgreSQL.

29. После успешного выполнения автонастройки необходимо выполнить подготовку сервера СУБД, для чего авторизоваться на сервере СУБД под пользователем root и создать учётные записи СУБД и базы данных, которые будут использоваться СПО для своей работы, в соответствии с официальной документацией по СУБД: https://www.postgresql.org/docs/.

30. Произвести автонастройку серверов доставки сообщений кластера Kafka.

31. После успешного выполнения автонастройки необходимо выполнить подготовку серверов кластера Kubernetes с помощью сервера непрерывной интеграции.

32. На сервере межведомственного взаимодействия необходимо произвести установку СПО с помощью сервера непрерывной интеграции.

33. Произвести автонастройку сервера R.

34. После успешного выполнения автонастройки необходимо на сервере R произвести установку СПО с помощью сервера непрерывной интеграции.

2.2 Развертывание баз данных системы

Для функционирования ПО необходимо создать БД.

При развертывании ПО из связки необходимо в соответствии с официальной документацией СУБД PostgreSQL:

создать табличное пространство «tablespace_ias»;

создать базы данных и пользователей (название базы данных совпадает с пользователем).

Поправить конфигурационные файлы СПО на сервере репозиториев кода, указав в них настройки для подключения к БД.

3 ПРОВЕРКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

После выполнения установки и настройки ПО проверка работоспособности осуществляется в два этапа:

проверка технического состояния ПО;

проверка установленного ПО.

3.1 Проверка технического состояния ПО

Проверка технического состояния ПО выполняется непосредственно на серверах. Требуется повышение привилегий до уровня пользователя root.

3.1.1 Сервер репозиториев компонентов

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# systemctl status nexus
```

3. Проверить открытые сетевые порты:

```
# ss -tln | egrep '80|8081'
```

4. Проверить доступность ОПО через аутентификацию в приложении:

```
http://<hostname>:8081
http://<hostname>:80
```

3.1.2 Сервер репозиториев кода

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# gitlab-ctl status
```

3. Проверить открытые сетевые порты:

```
# ss -tln | egrep '80'
```

4. Проверить доступность ОПО через аутентификацию в приложении:

```
http://<hostname>:80
```

3.1.3 Сервер непрерывной интеграции

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# systemctl status jenkins
```

3. Проверить открытые сетевые порты:

```
# ss -tln | egrep '80|8080'
```

4. Проверить журнал на предмет наличия ошибок:

```
# less /var/log/jenkins/jenkins.log
```

5. Проверить доступность веб-страницы http://<hostname>.

3.1.4 Сервер R

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# docker ps
```

3. Проверить открытые сетевые порты:

```
# ss -tln | egrep '6311'
```

3.1.5 Сервер баз данных PostgreSQL

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# systemctl status postgresql-11
```

3. Проверить открытые сетевые порты:

```
# ss -tln | egrep 5432
```

4. Проверить доступность БД через аутентификацию в приложении.

3.1.6 Сервер межведомственного взаимодействия

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# systemctl status smev3client
# systemctl status crypto-service
```

3. Проверить открытые сетевые порты:

```
# ss –tln | egrep '8090|8091'
```

4. Проверить журнал на предмет наличия ошибок:

```
# less /opt/smev3client/logs/spring.log
# less /opt/crypto-service/log/crypto-service.log
```

5. Проверить доступность http://<hostname>:8090/health
http://<hostname>:8091/actuator/health

3.1.7 Сервера приложений и управления кластером

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# docker ps
```

3. Проверить открытые сетевые порты:

```
# ss –tln | egrep '80'
```

3.1.8 Прокси-сервер

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# systemctl status nginx
```

3. Проверить открытые сетевые порты:

```
# ss –tln | egrep '80|443'
```

4. Проверить журнал на предмет наличия ошибок:

```
# less /var/log/nginx/error.log
```

5. Проверить доступность веб-страницы http://<hostname>.

3.1.9 Сервера доставки сообщений кластера Kafka

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# systemctl status confluent-kafka
# systemctl status confluent-zookeeper
# systemctl status confluent-kafka-connect
# systemctl status confluent-schema-registry
```

3. Проверить открытые сетевые порты:

```
# ss –tln | egrep '2181|7771|8080|8081|8083|9091|9092'
```

4. Проверить журнал на предмет наличия ошибок:

```
# less /var/log/kafka/server.log
# less /var/log/confluent/schema-registry/schema-registry.log
```

5. Проверить логи приложений на наличие успешного подключении к серверу доставки сообщений Kafka;

3.1.10 Сервер данных

1. Открыть интерактивный сеанс работы с сервером (консоль или ssh-сессия).
2. Проверить процессы требуемых сервисов:

```
# systemctl status nfs-server
```

3. Проверить открытые сетевые порты:

```
# ss –tln | egrep '111|2049|32765|32767'
```

4. Проверить журнал на предмет наличия ошибок:

```
# less /var/log/messages
```

5. Проверить на серверах приложений доступность NFS хранилища

3.2 Проверка установленного ПО

Чтобы выполнить проверку веб-интерфейса необходимо с рабочего места открыть в веб-браузере URL-адрес веб-приложения.