

«Универсальная мультимодальная дистрибутивная система»

Инструкция по установке

Оглавление

1	Численность и квалификация персонала Системы.....	4
2	Действия системного администратора при установке и наладке системы.....	5
2.1	Требования к рабочему месту системного администратора.....	5
2.2	Требования к техническому и программному обеспечению Системы	5
2.3	Развертывание ядра Системы	9
2.3.1	Установка и настройка системы управления АСУ	10
2.3.2	Настройка рабочих узлов АСУ	14
2.3.3	Установка и настройка Consul.....	17
2.3.4	Установка и настройка Kibana	28
2.3.5	Установка и настройка Ansible.....	29
2.4	Установка и настройка СУБД.....	32
2.4.1	Установка MongoDB.....	32
2.4.2	Настройка MongoDB	33
2.4.3	Установка и запуск ClickHouse	35
2.4.4	Установка PostgreSQL	37
2.4.5	Установка и настройка Elasticsearch	38
2.4.6	Установка Redis	39
2.4.7	Настройка Redis	40
2.4.8	Установка Rabbit MQ.....	40
2.4.9	Установка Kafka.....	42
2.5	Установка и настройка прочего системного ПО.....	45
2.5.1	Установка открытой универсальной платформы разработки .NET.....	45
2.5.2	Установка системы контейнеризации Mesosphere DC/OS	46
2.5.3	Установка балансировщика нагрузки Marathon-lb	47
2.5.4	Установка системы мониторинга Prometheus.....	48
2.5.5	Установка и настройка панели визуализации данных Kibana	51
2.5.6	Установка и настройка веб-сервера Nginx.....	53
2.5.7	Установка системы мониторинга Zabbix	58
2.6	Установка и настройка приложений УМДС	61
2.6.1	Подготовка системы управления АСУ	61
2.6.2	Развертка приложений УМДС	62
3	Эксплуатация, техническое обслуживание, ремонт и хранение компонентов системы.....	64
4	Действия при возникновении ошибок и неполадок	65

Перечень сокращений

Термин	Определение
АСУ потребителя	Автоматизированная система управления потребителя, к которой подключена Система
База данных (БД)	Совокупность данных, организованных в соответствии с концептуальной схемой, описывающей характеристики этих данных и связи между соответствующими им объектами, поддерживающая одну или несколько предметных областей
Доступ к информации (Доступ)	Ознакомление с информацией, ее обработка, в частности, копирование, модификация или уничтожение информации
Пользователь	Сотрудник потребителя или его партнера, участвующее в функционировании Системы или использующее результаты ее функционирования
Потребитель	Юридическое или физическое лицо, приобретшее право пользования Системой
ПО	Программное обеспечение
Разработчик	Общество с ограниченной ответственностью «ОНЭЛИЯ»
Система	Универсальная мультимодальная дистрибутивная система
СУБД	Система управления БД
ИС	Проприетарный набор серверов для нескольких служб Интернета от компании Microsoft

1 Численность и квалификация персонала Системы

Минимальное количество специалистов, необходимое для обеспечения работы Системы.

Для работы Системы минимально необходимы:

- 1 системный администратор;
- 1 администратор БД;
- 1 сетевой администратор;
- 1 специалист технической поддержки;
- 1 специалист по ИБ.

Персонал Системы должен обладать высоким уровнем квалификации в следующих областях:

- администрирование технических средств (серверы, рабочие станции);
- администрирование программного обеспечения операционных систем и систем управления базами данных;
- разработка, управление и реализация эффективной политики информационной безопасности;
- модернизация программных и технических средств.

2 Действия системного администратора при установке и наладке системы

2.1 Требования к рабочему месту системного администратора

Системный администратор получает доступ к Системе после заключения договора между разработчиком Системы и потребителем. Все подробные детали необходимо смотреть в соответствующем договоре. Доступ пользователя к Системе осуществляется в соответствии с договором напрямую или в режиме тонкого клиента, функционирующего в различных операционных средах – Microsoft Windows, Unix (Linux), Mac OS.

Для работы с Системой в режиме тонкого клиента необходимо:

- терминальное устройство (компьютер, смартфон, планшет и т.п.);
- доступ к сети Интернет или Интранет;
- браузер с поддержкой HTML 4.0, CSS Level 2, JavaScript 1.1. и выше, режима асинхронного взаимодействия JavaScript/XML (XMLHttpRequest и т.п.). Пользовательские интерфейсы Системы протестированы на совместимость с браузерами: Яндекс.Браузер версии 21.5 и выше, Safari версии 14 и выше, Google Chrome версии 90 и выше, Mozilla Firefox версии 89 и выше, Opera версии 76 и выше;
- сертификат безопасности (необходимость установки, установка и выдача сертификата безопасности определяется и производится системным администратором).

2.2 Требования к техническому и программному обеспечению Системы

Сведения о рекомендуемых технических и программных средствах, необходимых для функционирования Системы, приведены в таблицах 1,

Название сервера	Процессор	Оперативная память, Гб	Жесткий диск SSD, Гб	Жесткий диск HDD, Гб
Ядро Системы				
Ansible	2×2 МГц	1	0	21
Consul_N	2×2 МГц	1	21	0
Elasticsearch	4×2 МГц	8	32	96
Kube-cp_N	2×2 МГц	4	26	28
Kube-wn_N	8×2 МГц	16	32	96

Название сервера	Процессор	Оперативная память, Гб	Жесткий диск SSD, Гб	Жесткий диск HDD, Гб
MongoDB_N	4×2 МГц	8	0	520
Подсистема продаж				
Partner_N	2×2 МГц	2	10	0
Web_N	2×2 МГц	2	10	0
Manage	2×2 МГц	2	10	0
Справочная подсистема				
Nginx_frontend	2×2 МГц	2	18	0
Nginx_API_frontend	2×2 МГц	2	18	0
API_N	4×2 МГц	4	18	0
Zabbix	2×2 МГц	2	18	0
Kibana	2×2 МГц	2	18	0
Elastic_N	8×2 МГц	8	20	80
Logstash	4×2 МГц	4	18	0
Mongo_op	4×2 МГц	8	20	80
Mongo_hist	2×2 МГц	2	0	1000
Redis	2×2 МГц	2	18	0
App_N	2×2 МГц	2	17	0
App_robofarm_N	4×2 МГц	4	17	0
App_indexer	2×2 МГц	2	17	0
App_postgres	2×2 МГц	2	20	30
Подсистема мультимодальных путешествий				
PostgreSQL	4×2 МГц	15	750	150
MongoDB_AR	48×2 МГц	256	1500	150
ClickHouse	4×2 МГц	15	1125	150
ElasticC11	2×2 МГц	19	250	150

Название сервера	Процессор	Оперативная память, Гб	Жесткий диск SSD, Гб	Жесткий диск HDD, Гб
ElasticCl2	2×2 МГц	19	250	150
ElasticDataN	4×2 МГц	15	300	150
Прочие сервера				
CommCarrier_N	2×2 МГц	2	10	0
ProtectionPD	2×2 МГц	2	10	0

Таблица 2. Если в названии сервера имеется символ «N», значит количество этих серверов зависит от планируемой нагрузки на Систему.

Таблица 1 – Технические средства, необходимые для функционирования Системы

Название сервера	Процессор	Оперативная память, Гб	Жесткий диск SSD, Гб	Жесткий диск HDD, Гб
Ядро Системы				
Ansible	2×2 МГц	1	0	21
Consul_N	2×2 МГц	1	21	0
Elasticsearch	4×2 МГц	8	32	96
Kube-cp_N	2×2 МГц	4	26	28
Kube-wn_N	8×2 МГц	16	32	96
MongoDB_N	4×2 МГц	8	0	520
Подсистема продаж				
Partner_N	2×2 МГц	2	10	0
Web_N	2×2 МГц	2	10	0
Manage	2×2 МГц	2	10	0
Справочная подсистема				
Nginx_frontend	2×2 МГц	2	18	0
Nginx_API_frontend	2×2 МГц	2	18	0
API_N	4×2 МГц	4	18	0

Название сервера	Процессор	Оперативная память, Гб	Жесткий диск SSD, Гб	Жесткий диск HDD, Гб
Zabbix	2×2 МГц	2	18	0
Kibana	2×2 МГц	2	18	0
Elastic_N	8×2 МГц	8	20	80
Logstash	4×2 МГц	4	18	0
Mongo_op	4×2 МГц	8	20	80
Mongo_hist	2×2 МГц	2	0	1000
Redis	2×2 МГц	2	18	0
App_N	2×2 МГц	2	17	0
App_robofarm_N	4×2 МГц	4	17	0
App_indexer	2×2 МГц	2	17	0
App_postgres	2×2 МГц	2	20	30
Подсистема мультимодальных путешествий				
PostgreSQL	4×2 МГц	15	750	150
MongoDB_AR	48×2 МГц	256	1500	150
ClickHouse	4×2 МГц	15	1125	150
ElasticC11	2×2 МГц	19	250	150
ElasticC12	2×2 МГц	19	250	150
ElasticDataN	4×2 МГц	15	300	150
Прочие сервера				
CommCarrier_N	2×2 МГц	2	10	0
ProtectionPD	2×2 МГц	2	10	0

Таблица 2 – ПО, необходимое для функционирования Системы

ПО	Версия
ОС	Debian 10
СУБД ClickHouse	23.1.3.5 или новее

ПО	Версия
СУБД Elasticsearch	7.17.1 или новее
СУБД Kafka	14.6.0 или новее
СУБД MongoDB	5.0.2 или новее
СУБД PostgreSQL	12.4 или новее
СУБД Rabbit MQ	3.9.13 или новее
СУБД Redis	4.0.14 или новее
Открытая универсальная платформа разработки .NET Core	5
Система контейнеризации Mesosphere DC/OS	1.11.5 или новее
Балансировщик нагрузки Marathon-lb	1.11.3 или новее
Система мониторинга Prometheus	2.31.1 или новее
Система мониторинга Zabbix	6.2 или новее
Панель визуализации данных Kibana	7.17.1 или новее
Веб-сервер NGINX	1.21.2 или новее
Система управления конфигурациями Ansible	2.9.23 или новее
Система обнаружения и конфигурирования сервисов Consul	1.10.2 или новее
ПО для оркестровки контейнеризированных приложений Kubernetes	1.24.10 или новее
Браузер	Яндекс.Браузер версии 21.5 и выше Safari версии 14 и выше Google Chrome версии 90 и выше Mozilla Firefox версии 89 и выше Opera версии 76 и выше

2.3 Развертывание ядра Системы

Общий вид ядра Системы представлен на рисунке 1.

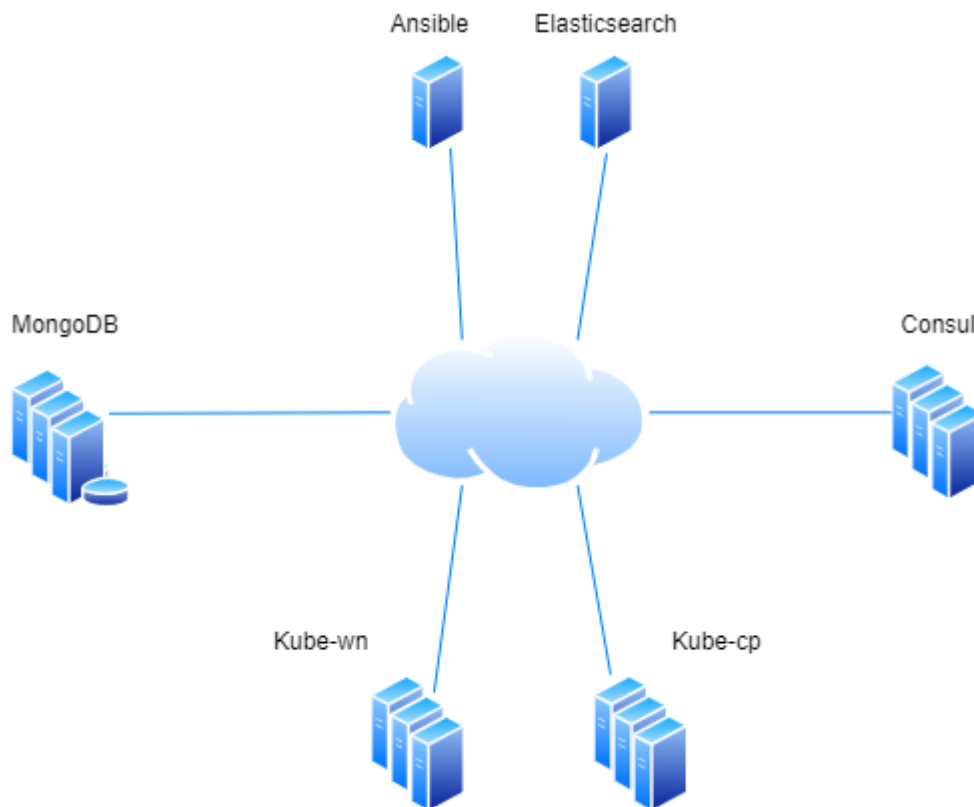


Рисунок 1 – Общий вид ядра Системы

Ядро Системы представляет собой одноранговую сеть из нескольких разнотипных серверов:

- Ansible – сервер системы управления конфигурациями (необязательно).
- Elasticsearch – сервер поисковой системы на базе платформы аналитики и визуализации Kibana (установка и настройка описаны в п. 2.4.5);
- MongoDB – комплект серверов БД (установка и настройка описаны в пп. 2.4.1 и 2.4.2);
- Consul – комплект серверов системы обнаружения и конфигурирования сервисов;
- Kube-wn – комплект серверов АСУ (рабочие узлы в кластере Kubernetes);
- Kube-cr – комплект серверов системы управления АСУ (Control Plane кластера Kubernetes).

Рекомендуемый состав ядра Системы включает в себя по одному серверу Ansible и Elasticsearch, а также комплекты по три сервера MongoDB, Consul, Kube-wn и Kube-cr.

2.3.1 Установка и настройка системы управления АСУ

Для работы АСУ потребуется механизм контейнеризации. Рекомендуется использовать docker, как самый популярный.

Настройку требуется выполнить на всех узлах системы управления АСУ (Kube-ср на рисунке 1) для формирования кластера АСУ.

Для создания системы управления АСУ требуется:

а) Предварительная настройка

На серверах системы управления АСУ требуется задать имена хоста и дополнить файлы /etc/hosts, к примеру:

```
# hostnamectl set-hostname master-node1
# cat <<EOF>> /etc/hosts
10.128.0.25 master-node1
10.128.0.26 master-node2
10.128.0.27 master-node3
10.128.0.29 node-1 worker-node-1
10.128.0.30 node-2 worker-node-2
10.128.0.31 node-3 worker-node-3
EOF
```

Рекомендуется выполнить проверку связи с рабочими узлами, чтобы убедиться в правильности работы обновленного файла хоста с помощью команды ping:

```
# ping 10.128.0.29
# ping 10.128.0.30
# ping 10.128.0.31
```

Затем требуется отключить систему принудительного контроля доступа SELinux и обновить правила брандмауэра.

```
# setenforce 0
# sed -i --follow-symlinks 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/sysconfig/selinux
# reboot
```

Заключительным этапом требуется установить правила брандмауэра для портов:

```
# firewall-cmd --permanent --add-port=6443/tcp
# firewall-cmd --permanent --add-port=2379-2380/tcp
# firewall-cmd --permanent --add-port=10250/tcp
# firewall-cmd --permanent --add-port=10251/tcp
# firewall-cmd --permanent --add-port=10252/tcp
# firewall-cmd --permanent --add-port=10255/tcp
# firewall-cmd --reload
# modprobe br_netfilter
# echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables
```

б) Настройка Kubernetes REPO

Требуется добавить хранилище Kubernetes, так как оно не установлено по умолчанию в CentOS 7.

```
cat <<EOF > /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
EOF
```

в) Установка KubeAdm и Docker

Для установки пакетов KubeAdm и Docker требуется выполнить следующую команду:

```
# yum install kubeadm docker -y
```

После успешного завершения установки требуется включить и запустить обе службы.

```
# systemctl enable kubelet
# systemctl start kubelet
```

```
# systemctl enable docker
# systemctl start docker
```

г) Установка Kubernetes Master и настройка пользователя по умолчанию

Перед инициализацией Kubernetes Master требуется отключить файл подкачки:

```
# swapoff -a
```

Инициализация Kubernetes Master на первом узле системы управления - это полностью автоматизированный процесс, управляемый командой `kubeadm init` (Kube-ср – доменное имя системы управления, 10.128.0.27 – IP адрес первого узла).

```
# kubeadm init --control-plane-endpoint=Kube-ср --apiserver-advertise-address=10.128.0.27
```

После успешного завершения установки требуется скопировать последнюю строку (далее – маркер) и сохранить ее на внешнем носителе, поскольку нужно будет запустить ее на рабочих узлах. При этом рекомендуется отредактировать ее, убрав символ «\» в конце первой строки:

```
kubeadm join Kube-ср:6443 --token nu06lu.xrsux0ss0ixtnms5 --discovery-token-ca-cert-hash
sha256:f996ea3564e6a07fdea2997a1cf8caeddafd6d4360d606dbc82314688425cd41
```

Инициализация Kubernetes Master на последующих узлах системы управления - это также полностью автоматизированный процесс, управляемый командой `kubeadm join` и дополненным маркетом (Kube-ср – доменное имя системы управления, 10.128.0.x – IP адрес узла).

```
kubeadm join Kube-ср:6443 --apiserver-advertise-address=10.128.0.25
--token nu06lu.xrsux0ss0ixtnms5 --discovery-token-ca-cert-hash
sha256:f996ea3564e6a07fdea2997a1cf8caeddafd6d4360d606dbc82314688425cd41 --control-plane
```

Для настройки системы управления АСУ требуется:

а) Инициализация Kubernetes Master

После успешной установки Kubernetes необходимо разрешить использование кластера. Рекомендуется запустить эту настройку от имени пользователя root:

```
# mkdir -p $HOME/.kube
```

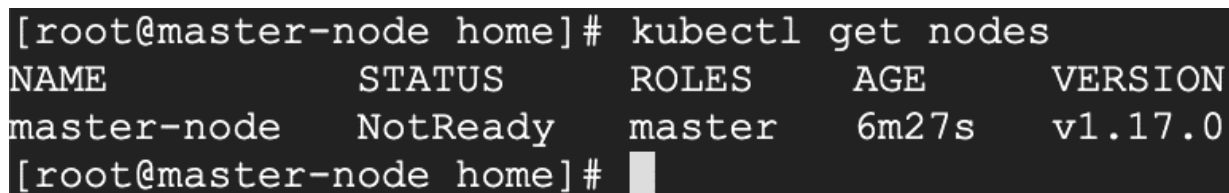
```
# cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
# chown $(id -u):$(id -g) $HOME/.kube/config
```

При необходимости запуска от имени пользователя `sudo` требуется перейти на пользователя с поддержкой `sudo` и выполнить следующие действия:

```
$ mkdir -p $HOME/.kube
$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Для завершения инициализации рекомендуется проверить статус узла системы управления.

```
# kubectl get nodes
```



```
[root@master-node home]# kubectl get nodes
NAME           STATUS    ROLES    AGE     VERSION
master-node    NotReady  master   6m27s   v1.17.0
[root@master-node home]# █
```

Рисунок 2 – Команда `kubectl` активирована

На этом этапе система управления имеет статус `NotReady`. Это связано с тем, что сеть модулей еще не развернута в кластере.

б) Настройка сети модуля

Применение сетевого кластера является очень гибким процессом в зависимости от потребностей пользователя и наличия множества доступных вариантов. Для более простой установки ниже описано использование плагина `Weavenet`, который не требует никакой конфигурации или дополнительного кода, он предоставляет один IP-адрес на модуль:

```
# export kubever=$(kubectl version | base64 | tr -d '\n')
# kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$kubever"
```

```
[root@master-node home]# export kubever=$(kubectl version | base64 | tr -d '\n')
[root@master-node home]# kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$kubever"
serviceaccount/weave-net created
clusterrole.rbac.authorization.k8s.io/weave-net created
clusterrolebinding.rbac.authorization.k8s.io/weave-net created
role.rbac.authorization.k8s.io/weave-net created
rolebinding.rbac.authorization.k8s.io/weave-net created
daemonset.apps/weave-net created
[root@master-node home]# █
```

Рисунок 3 – Настройка сети модуля

Для завершения настройки рекомендуется проверить статус узла.

```
# kubectl get nodes
```

```
[root@master-node home]# kubectl get nodes
NAME             STATUS    ROLES    AGE   VERSION
master-node     Ready    master   23m   v1.17.0
[root@master-node home]# █
```

Рисунок 4 – Узел системы управления активирован

2.3.2 Настройка рабочих узлов АСУ

Настройку требуется выполнить на всех рабочих узлах (Kube-wп на рисунке 1) для присоединения к кластеру АСУ.

а) Предварительная настройка

На рабочих узлах требуется задать имя, а также обновить основные и рабочие узлы в файле /etc/hosts.

```
# hostnamectl set-hostname 'node-1'
# cat <<EOF>> /etc/hosts
10.128.0.25 master-node1
10.128.0.26 master-node2
10.128.0.27 master-node3
10.128.0.29 node-1 worker-node-1
10.128.0.30 node-2 worker-node-2
10.128.0.31 node-3 worker-node-3
```

EOF

Рекомендуется выполнить проверку связи с узлами системы управления, чтобы убедиться в правильности работы обновленного файла хоста с помощью команды ping:

```
# ping 10.128.0.25
# ping 10.128.0.26
# ping 10.128.0.27
```

Затем требуется отключить систему принудительного контроля доступа SELinux и обновить правила брандмауэра.

```
# setenforce 0
# sed -i --follow-symlinks 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/sysconfig/selinux
```

Заключительным этапом требуется установить правила брандмауэра для портов:

```
# firewall-cmd --permanent --add-port=6783/tcp
# firewall-cmd --permanent --add-port=10250/tcp
# firewall-cmd --permanent --add-port=10255/tcp
# firewall-cmd --permanent --add-port=30000-32767/tcp
# firewall-cmd --reload
# echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables
```

б) Настройка Kubernetes REPO

Требуется добавить хранилище Kubernetes, так как оно не установлено по умолчанию в CentOS 7.

```
cat <<EOF > /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=1
```

```
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
```

```
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
```

EOF

в) Установка KubeAdm и Docker

Для установки пакетов KubeAdm и Docker требуется выполнить следующую команду:

```
# yum install kubeadm docker -y
```

После успешного завершения установки требуется включить и запустить обе службы.

```
# systemctl enable kubelet
```

```
# systemctl start kubelet
```

```
# systemctl enable docker
```

```
# systemctl start docker
```

г) Присоединение рабочего узла к кластеру АСУ

Для присоединения к кластеру требуется маркер, созданный kubeadm init на узле системы управления (см. п. «г») процедуры создания системы управления АСУ). Его требуется скопировать и вставить во все рабочие узлы.

```
# kubeadm join Kube-ср:6443 --token nu06lu.xrsux0ss0ixtnms5 --discovery-token-ca-cert-hash sha256:f996ea3564e6a07fdea2997a1cf8caeddafd6d4360d606dbc82314688425cd41
```

```
[root@node-1 home]# kubeadm join 10.128.0.27:6443 --token nu06lu.xrsux0ss0ixtnms5 --discovery-token-ca-cert-hash sha256:f996ea3564e6a07fdea2997a1cf8caeddafd6d4360d606dbc82314688425cd41
W0102 09:13:53.311788      2432 join.go:346] [preflight] WARNING: JoinControlPlane.controlPlane settings will be ignored when control-plane flag is not set.
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[kubelet-start] Downloading configuration for the kubelet from the "kubelet-config-1.17" ConfigMap in the kube-system namespace
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

[root@node-1 home]#
```

Рисунок 5 – Рабочий узел подключен к кластеру АСУ

После подключения всех рабочих узлов с любого узла системы управления АСУ требуется выполнить команду

```
# kubectl get nodes
```

При успешной настройке будут показаны все узлы АСУ в состоянии готовности (параметр «STATUS» в значении «Ready»).

2.3.3 Установка и настройка Consul

На рабочих узлах системы управления АСУ требуется поставить consul-клиент, синхронизированный с нужными consul-серверами. На соответствующих ЭВМ необходимо выполнить следующие действия под права администратора root:

а) Установка дополнительных пакетов

Выполнить команду:

```
yum install unzip wget
```

где:

unzip — необходим для распаковки архивов.

wget — утилита для загрузки файлов по сети.

б) Настройка времени

Требуется задать часовой пояс:

```
timedatectl set-timezone Europe/Moscow
```

Примечание: допустимо указать любой часовой пояс, но единый на всех серверах Consul.

Требуется установить утилиту для синхронизации времени:

```
yum install chrony
```

Затем запустить ее в качестве сервиса:

```
systemctl enable chronyd --now
```

в) Настройка имени серверов

Для сервера Consul важна доступность по имени.

```
hostnamectl set-hostname consul01.dmosk.local
```

```
hostnamectl set-hostname consul02.dmosk.local
```

```
hostnamectl set-hostname consul03.dmosk.local
```

П р и м е ч а н и е : предполагается, что трем серверам будут заданы имена consul01, consul02 и consul03. Они будут в домене dmosk.local.

Чтобы серверы могли обращаться друг к другу по именам, их надо либо зарегистрировать в локальной системе DNS, либо добавить на каждом сервере следующие записи в файл hosts:

```
10.128.0.15 consul01.dmosk.local
```

```
10.128.0.17 consul02.dmosk.local
```

```
10.128.0.19 consul03.dmosk.local
```

г) Настройка безопасности

Требуется открыть порты в брандмауэре:

```
firewall-cmd --add-port={8300,8301,8302,8500,8600}/tcp --permanent
```

```
firewall-cmd --add-port={8301,8302,8600}/udp --permanent
```

```
firewall-cmd --reload
```

д) Установка и запуск Consul

Установка будет выполнена путем копирования бинарного файла. Для начала требуется перейти на страницу загрузки программного продукта и определить его последнюю версию.

После требуется подставить нужную версию в переменную:

```
export VER="1.10.2"
```

Примечание: в описанном примере будет устанавливаться версия 1.10.2.

После требуется загрузить и распаковать требуемый архив в каталог /usr/bin:

```
wget https://releases.hashicorp.com/consul/${VER}/consul_${VER}_linux_amd64.zip  
unzip consul_${VER}_linux_amd64.zip
```

Для проверки установки приложения требуется выполнить команду:

```
consul -v
```

На экране будет отображена информация, подобная следующей:

```
Consul v1.10.2
```

```
Revision 3cb6eedb
```

```
Protocol 2 spoken by default, understands 2 to 3 (agent will automatically use protocol >2 when  
speaking to compatible agents)
```

е) Настройка

Требуется создать системную учетную запись, от которой будет работать консул:

```
useradd -r -c 'Consul DCS service' consul
```

Требуется создать каталоги для приложения и определить на них права:

```
mkdir -p /var/lib/consul/etc/consul.d
chown consul:consul /var/lib/consul/etc/consul.d
chmod 775 /var/lib/consul/etc/consul.d
```

Примечание: в примере указано, что владельцем данных каталогов будет созданная учетная запись consul. Права будут полные у владельца, остальные смогут читать данные.

Требуется сгенерировать ключ для консула на любой из нод кластера:

```
consul keygen
```

```
zpjf5a4reDbJFpT6FeaF0LGxD0zBRPSRbIoUkLBt0ps=
```

Полученный ключ требуется сохранить. Он будет использован при настройке всех узлов кластера.

Требуется создать конфигурационный файл для консула:

```
vi /etc/consul.d/config.json
```

```
{
  "bind_addr": "0.0.0.0",
  "bootstrap_expect": 3,
  "client_addr": "0.0.0.0",
  "datacenter": "dc1",
  "data_dir": "/var/lib/consul",
  "domain": "consul",
  "enable_script_checks": true,
  "dns_config": {
    "enable_truncate": true,
    "only_passing": true
  },
  "enable_syslog": true,
  "encrypt": "zpjf5a4reDbJFpT6FeaF0LGxD0zBRPSRbIoUkLBt0ps=",
  "leave_on_terminate": true,
```

```

"log_level": "INFO",
"rejoin_after_leave": true,
"retry_join": [
  "consul01.dmosk.local",
  "consul02.dmosk.local",
  "consul03.dmosk.local"
],
"server": true,
"start_join": [
  "consul01.dmosk.local",
  "consul02.dmosk.local",
  "consul03.dmosk.local"
],
"ui_config": { "enabled": true }
}

```

где:

`bind_addr` — адрес сервера Consul. Это может быть IP любого из сетевых интерфейсов.

`bootstrap_expect` — ожидаемое количество серверов в кластере.

`client_addr` — адрес, к которому будут привязаны клиентские интерфейсы.

`datacenter` — привязка сервера к конкретному датацентру. Нужен для логического разделения. Серверы с одинаковым датацентром должны находиться в одной локальной сети.

`data_dir` — каталог для хранения данных.

`domain` — домен, в котором будет зарегистрирован сервис.

`enable_script_checks` — разрешает на агенте проверку работоспособности.

`dns_config` — параметры для настройки DNS.

`enable_syslog` — разрешение на ведение лога.

`encrypt` — ключ для шифрования сетевого трафика. В качестве значения требуется использовать сгенерированный ранее.

`leave_on_terminate` — при получении сигнала на остановку процесса консула, корректно отключать ноду от кластера.

`log_level` — минимальный уровень события для отображения в логе. Возможны варианты "trace", "debug", "info", "warn", and "err".

`rejoin_after_leave` — по умолчанию, нода покидающая кластер не присоединяется к нему автоматически. Данная опция позволяет управлять данным поведением.

`retry_join` — узлы, к которым можно присоединить кластер. Процесс будет повторяться, пока не завершится успешно.

`server` — режим работы сервера.

`start_join` — список узлов кластера, к которым пробуем присоединиться при загрузке сервера.

`ui_config` — конфигурация для графического веб-интерфейса.

Проверку корректности конфигурационного файла можно осуществить выполнением команды:

```
consul validate /etc/consul.d/config.json
```

В случае успешной проверки на экране отобразится надпись:

...

Configuration is valid!

В завершение настройки рекомендуется создать командный файл в `systemd` для возможности автоматического запуска сервиса:

```
vi /etc/systemd/system/consul.service
```

```
[Unit]
```

```
Description=Consul Service Discovery Agent
```

```
Documentation=https://www.consul.io/
```

```
After=network-online.target
```

```
Wants=network-online.target
```

```
[Service]
```

```
Type=simple
```

```
User=consul
```

```
Group=consul
```

```
ExecStart=/usr/bin/consul agent \
```

```
-node=consul01.dmosk.local \
```

```
-config-dir=/etc/consul.d
ExecReload=/bin/kill -HUP $MAINPID
KillSignal=SIGINT
TimeoutStopSec=5
Restart=on-failure
SyslogIdentifier=consul

[Install]
WantedBy=multi-user.target
```

В указанном примере указан сервер `consul01.dmosk.local`. Для каждого из настраиваемых серверов это имя сервера соответственно меняется. В результате будет сформировано три файла с разными значениями для опции `node`.

Далее требуется перезагрузить конфигурацию `systemd`:

```
systemctl daemon-reload
```

Система установлена, настроена и готова к запуску.

ж) Запуск и проверка

Запуск сервиса осуществляется командой:

```
systemctl start consul
```

Также рекомендуется разрешить автоматический старт при запуске сервера:

```
systemctl enable consul
```

Текущее состояние работы сервиса можно узнать командой:

```
systemctl status consul
```

В нормальном режиме отображается состояние:

...

Active: active (running) since ...

...

Состояние нод кластера можно узнать командой:

```
consul members
```

Дополнительную информацию можно отобразить командой:

```
consul members -detailed
```

Также доступен веб-интерфейс по адресу:

<http://<IP-адрес любого сервера консул>:8500>.

На странице по умолчанию отображается статус кластера.

з) Добавление аутентификации

При входе на веб-интерфейс или работе из командной строки, система позволяет выполнять операции без запроса идентификационных данных. Для защиты системы требуется настроить ACL и обновить сервис.

В конфигурационных файлах на всех узлах кластера Consul требуется добавить в файл `/etc/consul.d/config.json` следующие строки:

```
{
  ...
  "ui_config": { "enabled": true },
  "acl": {
    "enabled": true,
    "default_policy": "deny",
    "enable_token_persistence": true
  }
}
```

Примечание: Добавленная настройка `acl` запрещает по умолчанию доступ и требует ввода токена безопасности.

Проверка корректности внесенных настроек выполняется командой:

```
consul validate /etc/consul.d/config.json
```

Если ошибок нет, то требуется перезапустить сервис:

```
systemctl restart consul
```

Для получения токена необходимо выполнить команду:

```
consul acl bootstrap
```

Примечание: если команда нам вернула ошибку `Failed ACL bootstrapping: Unexpected response code: 500 (The ACL system is currently in legacy mode.)`, то значит, что мы не на всех нодах применили новую настройку. Проверяем, на всех ли серверах кластера внесены соответствующие изменения по добавлению ACL.

На экране будет отображен ответ следующего типа:

AccessorID: af5eaac1-4f0b-d46a-58ba-64ec857dfc4c

SecretID: 59ac7fa8-dca6-e066-ff33-0bf9bb6f466a

Description: Bootstrap Token (Global Management)

Local: false

Create Time: 2021-09-01 16:28:58.710545082 +0300 MSK

Policies:

00000000-0000-0000-0000-000000000001 - global-management

В данном примере `59ac7fa8-dca6-e066-ff33-0bf9bb6f466a` — нужный для аутентификации в веб-интерфейсе SecretID.

Для работы из командной строки теперь также необходимо авторизоваться. Для упрощения авторизации рекомендуется создать переменную в системном окружении:

```
export CONSUL_HTTP_TOKEN=59ac7fa8-dca6-e066-ff33-0bf9bb6f466a
```

где 59ac7fa8-dca6-e066-ff33-0bf9bb6f466a – полученный SecretID.

Создать новый токен можно командой:

```
consul acl token create -policy-name global-management
```

Чтобы увидеть список токенов, требуется ввести:

```
consul acl token list
```

Удалить токен можно по его AccessorID:

```
consul acl token delete -id 54b5f2bb-1a57-3884-f0ea-1284f84186f5
```

Примечание: в примере будет удален токен с AccessorID 54b5f2bb-1a57-3884-f0ea-1284f84186f5.

и) Настройка политики для DNS

После включения ACL система перестанет отвечать на запросы DNS. Это связано с политикой блокировки по умолчанию. Для разрешения запросов требуется создать политику с разрешением данных запросов, получить для нее токен и применить данный токен на всех серверах кластера Consul.

На одном из серверов требуется создать файл с политикой:

```
cd /etc/consul.d/
```

```
vi dns-request-policy.txt
```

```
node_prefix "" {  
  policy = "read"  
}
```

```
service_prefix "" {  
  policy = "read"  
}
```

Далее требуется создать политику в консуле, а затем токен безопасности на основе политики:

```
consul acl policy create -name "dns-requests" -rules @dns-request-policy.txt
```

```
consul acl token create -description "Token for DNS Requests" -policy-name dns-requests
```

На экране будет отображен ответ следующего типа:

```
AccessorID:    b53741e2-7663-eag6-fd67-a64dbd32feb5  
SecretID:     42bd65e5-42a5-356b-a81b-60eff20f657  
Description:   Token for DNS Requests  
Local:        false  
Create Time:   2021-09-02 12:37:42.342511655 +0300 MSK  
Policies:  
  89e42b6b-bbec-5263-bc7b-60f3a604a0d6 - dns-requests
```

* где SecretID — это нужный нам токен.

На всех серверах Consul выполняем команду:

```
export CONSUL_HTTP_TOKEN=59ac7fa8-dca6-e066-ff33-0bf9bb6f466a
```

* где 59ac7fa8-dca6-e066-ff33-0bf9bb6f466a — ранее созданный токен для получения доступа к управлению консулом.

А затем:

```
consul acl set-agent-token default 42bd65e5-42a5-356b-a81b-60eff20f657
```

* где 42bd65e5-42a5-356b-a81b-60eff20f657 — SecretID, полученный на основе политики, разрешающей запросы DNS.

2.3.4 Установка и настройка Kibana

Пакет Kibana используется для визуализации данных, полученных из Elasticsearch. Репозитории и публичный ключ для установки Kibana будут такими же, как в установке Elasticsearch.

а) Установка Kibana

Предварительно необходимо скопировать публичный ключ репозитория:

```
# rpm --import https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

Затем требуется подключить репозиторий Kibana:

```
# mcedit /etc/yum.repos.d/kibana.repo
```

```
[kibana-7.x]
```

```
name=Kibana repository for 7.x packages
```

```
baseurl=https://artifacts.elastic.co/packages/7.x/yum
```

```
gpgcheck=1
```

```
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

```
enabled=1
```

```
autorefresh=1
```

```
type=rpm-md
```

Установка пакета Kibana осуществляется командой:

```
# yum install kibana
```

В завершении установки требуется добавить Elasticsearch в автозагрузку и запустить его с дефолтными настройками:

```
# systemctl daemon-reload
```

```
# systemctl enable kibana.service
```

```
# systemctl start kibana.service
```

Проверку работы службы можно осуществить командой:

```
# systemctl status kibana.service
```

По умолчанию, Kibana слушает порт 5601.

Внимание! Кибана стартует длительное время. Подождите примерно минуту и проверяйте.

Проверку работы Kibana можно осуществить командой:

```
# netstat -tulnp | grep 5601
```

На экране отобразится информация, подобная следующей:

```
tcp    0    0 127.0.0.1:5601    0.0.0.0:*        LISTEN  20746/node
```

б) Настройка Kibana

Файл с настройками Kibana располагается по пути - /etc/kibana/kibana.yml.

Для разрешения прослушки внешнего интерфейса и внешних подключений требуется изменить параметр `server.host`, указав IP адрес сервера:

```
server.host: "10.128.0.29"
```

Для прослушки всех интерфейсов требуется указать в качестве адреса 0.0.0.0.

После изменения настроек Kibana сервис требуется перезапустить:

```
# systemctl restart kibana.service
```

Теперь можно зайти в веб интерфейс по адресу `http://10.128.0.29:5601`.

2.3.5 Установка и настройка Ansible

Пакет Ansible используется для централизованного управления остальными машинами системы.

а) Установка Ansible

Предварительно необходимо установить репозиторий EPEL командой:

```
sudo yum install epel-release
```

Затем требуется установить Ansible командой:

```
sudo yum install ansible
```

б) Настройка хостов Ansible

Ansible взаимодействует с серверами по SSH. При этом система не должна запрашивать паролей. Потому нужно настроить аутентификацию на основе SSH-ключей.

Чтобы получить доступ к серверам системы с сервера Ansible, следует ввести команду:

```
ssh root@your_server_ip
```

Ansible отслеживает все серверы, указанные в файле hosts. Чтобы система Ansible могла взаимодействовать с остальными машинами, нужно соответственно отредактировать этот файл, к примеру в текстовом редакторе sudo:

```
sudo vi /etc/ansible/hosts
```

В файле хранится множество настроек и примеров конфигурации. Все они закомментированы. Не стоит удалять из файла примеры конфигураций, поскольку они могут пригодиться в дальнейшей работе.

Для добавления серверов системы следует использовать такой синтаксис:

```
[group_name] alias ansible_ssh_host=your_server_ip
```

Тег group_name позволяет сослаться сразу на несколько серверов; alias задаёт имя сервера.

Серверы можно разделить на несколько групп, каждая из которых будет использовать разные параметры настроек.

В файле /etc/ansible/group_vars/all можно указать параметры конфигурации для каждого сервера независимо от группы. Индивидуальные настройки хостов можно поместить в каталог /etc/ansible/host_vars.

Для проверки подключения серверов системы к серверу Ansible можно осуществить командой ping:

```
ansible -m ping all
```

При успешной проверке на экране будет отображено:

```
host1 | SUCCESS => {
  "changed": false,
  "ping": "..."}
host3 | SUCCESS => {
  "changed": false,
  "ping": "..."}
}
```

```
host2 | SUCCESS => {  
  "changed": false,  
  "ping": "..."  
}
```

2.4 Установка и настройка СУБД

2.4.1 Установка MongoDB

MongoDB — это бесплатная БД документов с открытым исходным кодом.

Выполните следующие шаги от имени пользователя root или пользователя с привилегиями sudo для установки MongoDB в системе Debian:

а) Установите пакеты, необходимые для добавления нового репозитория:

```
sudo apt install dirmngr gnupg apt-transport-https software-properties-common ca-certificates  
curl
```

б) Добавьте в систему ключ MongoDB GPG:

```
curl -fsSL https://www.mongodb.org/static/pgp/server-4.2.asc | sudo apt-key add -
```

в) Включите репозиторий MongoDB:

```
sudo add-apt-repository 'deb https://repo.mongodb.org/apt/debian buster/mongodb-org/4.2 main'
```

Примечание: Пакеты со старыми версиями MongoDB недоступны для Debian 10.

г) Обновите список пакетов и установите mongodb-org метапакет:

```
sudo apt update sudo apt install mongodb-org
```

Следующие пакеты будут установлены в системе как часть mongodb-org пакета:

mongodb-org-server - mongod (демон и соответствующие сценарии инициализации и конфигурации).

mongodb-org-mongos - mongos (демон).

mongodb-org-shell - Оболочка mongo (интерактивный интерфейс JavaScript для MongoDB).

Он используется для выполнения административных задач через командную строку).

mongodb-org-tools. Содержит несколько инструментов MongoDB для импорта и экспорта данных, статистики, а также другие утилиты.

д) Запустите службу MongoDB и включите ее запуск при загрузке:

```
sudo systemctl enable mongod --now
```

е) Чтобы проверить, успешно ли завершилась установка, подключитесь к серверу базы данных MongoDB с помощью mongo инструмента и распечатайте статус подключения:

```
mongo --eval 'db.runCommand({ connectionStatus: 1 })'
```

Результат будет выглядеть следующим образом:

```
MongoDB shell version v4.2.1
```

```
connecting to:
```

```
mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
```

```
Implicit session: session { "id" : UUID("09f11c53-605f-44ad-abec-ec5801bb6b06") }
```

```
MongoDB server version: 4.2.1
```

```
{
  "authInfo" : {
    "authenticatedUsers" : [ ],
    "authenticatedUserRoles" : [ ]
  },
  "ok" : 1
}
```

Примечание: Значение 1 для параметра «ок» указывает на успех.

2.4.2 Настройка MongoDB

Для настройки MongoDB требуется отредактировать файл конфигурации `/etc/mongod.conf`, написанный на YAML, и создать одного административного пользователя MongoDB.

В файле требуется раскомментировать раздел безопасности и включить авторизацию:

```
/etc/mongod.conf
```

```
security:
```

```
authorization: enabled
```

Параметр `authorization` включает управление доступом на основе ролей (RBAC), которое регулирует доступ пользователей к ресурсам и операциям базы данных. Если эта опция отключена, каждый пользователь будет иметь доступ к любой базе данных и сможет выполнять любые действия.

После внесения изменений в файл конфигурации MongoDB требуется перезапустить службу `mongod`:

```
sudo systemctl restart mongod
```

Дополнительная информация о параметрах файла конфигурации доступна на сайте разработчика – <https://docs.mongodb.com/manual/reference/configuration-options/>.

Для создания административного пользователя MongoDB требуется выполнить следующие действия:

а) Загрузите оболочку MongoDB с помощью команды `mongo`.

б) В оболочке MongoDB введите набор команд для подключения к базе данных `admin`:

```
use admin
```

```
switched to db admin
```

в) Создайте нового пользователя с именем `mongoAdmin` с ролью `userAdminAnyDatabase`:

```
db.createUser(  
  {  
    user: "mongoAdmin",  
    pwd: "changeMe",  
    roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]  
  }  
)
```

```
Successfully added user: {  
  "user" : "mongoAdmin",  
  "roles" : [  
    {  
      "role" : "userAdminAnyDatabase",  
      "db" : "admin"  
    }  
  ]  
}
```

г) Выйдите из оболочки MongoDB с помощью команды `quit()`

2.4.3 Установка и запуск ClickHouse

Для установки ClickHouse на сервер под управлением Debian выполните следующие действия от имени пользователя root или пользователя с привилегиями sudo:

Шаг 1 — Установка ClickHouse

а) Установите dirmngr, выполнив:

```
sudo apt install dirmngr
```

Примечание: dirmngr — это сервер для управления сертификатами и ключами. Он необходим для добавления и проверки ключей удаленного репозитория.

б) Добавьте ключ репозитория GPG для возможности безопасно загружать проверенные пакеты ClickHouse:

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv E0C56BD4
```

При успешном выполнении команды на экране отобразится следующая информация:

```
Executing: /tmp/apt-key-gpghome.JkkcKnBAFY/gpg.1.sh --keyserver keyserver.ubuntu.com --recv E0C56BD4
```

```
gpg: key C8F1E19FE0C56BD4: public key "ClickHouse Repository Key <milovidov@yandex-team.ru>" imported
```

```
gpg: Total number processed: 1
```

```
gpg:         imported: 1
```

в) Добавьте репозиторий в свой список репозитория АРТ, выполнив:

```
echo "deb http://repo.yandex.ru/clickhouse/deb/stable/ main/" | sudo tee /etc/apt/sources.list.d/clickhouse.list
```

Примечание: команда передает вывод echo на sudo tee, чтобы этот вывод мог распечатать файл, принадлежащий root.

г) Запустите команду apt update для обновления пакетов:

```
sudo apt update
```

Пакеты clickhouse-server и clickhouse-client теперь будут доступны для установки.

д) Начиная с версии ClickHouse 19.3.6, определенные конфигурации OpenSSL 1.1.1, например `MinProtocol` и `CipherVersion`, читаются некорректно. Чтобы обойти эту несовместимость, измените файл конфигурации OpenSSL и закомментируйте строку `ssl_conf = ssl_sect` в `/etc/ssl/openssl.cnf`:

```
/etc/ssl/openssl.cnf
...

tsa_name          = yes # Must the TSA name be included in the reply?
                   # (optional, default: no)
ess_cert_id_chain = no  # Must the ESS cert id chain be included?
                   # (optional, default: no)
ess_cert_id_alg   = sha1 # algorithm to compute certificate
                   # identifier (optional, default: sha1)

[default_conf]
#ssl_conf = ssl_sect

[ssl_sect]
...
```

е) Теперь после корректировки конфигурации OpenSSL требуется установить пакеты сервера и клиента ClickHouse:

```
sudo apt install clickhouse-server clickhouse-client
```

Во время установки будет предложено задать пароль пользователя ClickHouse по умолчанию.

Шаг 2 — Запуск службы

Пакет `clickhouse-server` при установке создает службу `systemd`, которая выполняет такие действия, как запуск, остановка и перезапуск сервера базы данных. `systemd` — это система инициализации, позволяющая Linux инициировать и управлять службами.

а) Запустите службу `clickhouse-server`, выполнив:

```
sudo service clickhouse-server start
```

б) Предыдущая команда не будет отображать никакие результаты. Чтобы убедиться, что служба работает успешно, выполните:

```
sudo service clickhouse-server status
```

Результат будет выглядеть примерно так:

```
clickhouse-server.service - ClickHouse Server (analytic DBMS for big data)
```

```
Loaded: loaded (/etc/systemd/system/clickhouse-server.service; enabled; vendor preset: enabled)
```

```
Active: active (running) since Sat 2018-12-22 07:23:20 UTC; 1h 9min ago
```

```
Main PID: 27101 (ClickHouse-serv)
```

```
Tasks: 34 (limit: 1152)
```

```
CGroup: /system.slice/ClickHouse-server.service
```

```
└─27101 /usr/bin/ClickHouse-server --config=/etc/ClickHouse-server/config.xml
```

Результат означает, что сервер работает.

2.4.4 Установка PostgreSQL

Для установки PostgreSQL на сервер под управлением Debian выполните следующие действия от имени пользователя root или пользователя с привилегиями sudo:

а) Обновите пакет APT, если это не было сделано при установке другого ПО Системы:

```
sudo apt update
```

б) Установите сервер PostgreSQL и пакет contrib, который предоставляет дополнительные функции для базы данных PostgreSQL:

```
sudo apt install postgresql postgresql-contrib
```

в) После завершения установки запустится служба PostgreSQL. Чтобы проверить установку, используйте инструмент psql для печати версии сервера:

```
sudo -u postgres psql -c "SELECT version();"
```

Результат должен выглядеть примерно так:

```
PostgreSQL 11.5 (Debian 11.5-1+deb10u1) on x86_64-pc-linux-gnu, compiled by gcc (Debian 8.3.0-6) 8.3.0, 64-bit
```

psql — это интерактивная терминальная программа, которая позволяет вам взаимодействовать с сервером PostgreSQL.

2.4.5 Установка и настройка Elasticsearch

Пакет Elasticsearch используется для хранения, анализа, поиска информации по логам.

Для установки Elasticsearch на сервер seed_platform выполните следующие действия:

а) Установка Elasticsearch

Предварительно необходимо скопировать публичный ключ репозитория:

```
# rpm --import https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

б) Затем требуется подключить репозиторий Elasticsearch:

```
# mcedit /etc/yum.repos.d/elasticsearch.repo
```

```
[elasticsearch]
```

```
name=Elasticsearch repository for 7.x packages
```

```
baseurl=https://artifacts.elastic.co/packages/7.x/yum
```

```
gpgcheck=1
```

```
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

```
enabled=0
```

```
autorefresh=1
```

```
type=rpm-md
```

в) Установка пакета Elasticsearch осуществляется командой:

```
# yum install --enablerepo=elasticsearch elasticsearch
```

В завершение установки требуется добавить Elasticsearch в автозагрузку и запустить его с дефолтными настройками:

```
# systemctl daemon-reload
```

```
# systemctl enable elasticsearch.service
```

```
# systemctl start elasticsearch.service
```

г) Проверку работы службы можно осуществить командой:

```
# systemctl status elasticsearch.service
```

д) Настройка Elasticsearch

Настройки Elasticsearch находятся в файле `/etc/elasticsearch/elasticsearch.yml`. Интерес представляют следующие параметры:

```
path.data: /var/lib/elasticsearch # директория для хранения данных;
network.host: 127.0.0.1 # слушаем только локальный интерфейс.
```

По умолчанию Elasticsearch слушает `localhost`. Так как необходимо, чтобы Elasticsearch слушал все сетевые интерфейсы, то требуется изменить параметр `network.host`:

```
network.host: 0.0.0.0
```

Директории с данными будут занимать значительное место, поэтому следует продумать их размещение заранее.

К списку параметров в файле `/etc/elasticsearch/elasticsearch.yml` требуется добавить:

```
discovery.seed_hosts: ["127.0.0.1", "[:,1]"]
```

Параметр указывает, что хосты кластера следует искать только локально.

Для применения измененных настроек требуется перезапустить службу:

```
# systemctl restart elasticsearch.service
```

2.4.6 Установка Redis

Redis — сетевое журналируемое хранилище данных типа «ключ - значение» с открытым исходным кодом. Нереляционная высокопроизводительная СУБД.

Redis версии 5.0.x включен в репозитории Debian 10 по умолчанию. Для его установки выполните следующие команды от имени пользователя `root` или пользователя с привилегиями `sudo`:

```
sudo apt update
sudo apt install redis-server
```console-bash
```

The Redis service will start automatically when the installation finishes. You can verify it by typing:

```
```console-bash
sudo systemctl status redis-server
```

Вывод должен выглядеть следующим образом:

```
redis-server.service - Advanced key-value store
Loaded: loaded (/lib/systemd/system/redis-server.service; enabled; vendor preset: enabled)
Active: active (running) since Thu 2019-11-28 14:15:23 PST; 27s ago
    Docs: http://redis.io/documentation,
           man:redis-server(1)
Main PID: 2024 (redis-server)
    Tasks: 4 (limit: 2359)
    Memory: 6.9M
    CGroup: /system.slice/redis-server.service
            └─2024 /usr/bin/redis-server 127.0.0.1:6379
```

Служба Redis не запустится, если на вашем сервере отключен IPv6.

2.4.7 Настройка Redis

Конфигурация сервиса: `/etc/redis.conf`

Лог сервиса: `/var/log/redis/`

Настройки ОС, выполненные для корректной работы сервиса:

`/etc/sysctl.d/redis.conf`:

`net.core.somaxconn = 1024` - настройка максимального количества прослушиваемых портов; `vm.overcommit_memory = 1` - правило работы `overcommit`.

`/etc/sysconfig/grub` - отключение `transparent hugepages (THP)` (`transparent_hugepage=never`).

Сервис добавлен в автозапуск.

2.4.8 Установка Rabbit MQ

Для установки Rabbit MQ выполните следующие действия от имени пользователя `root` или пользователя с привилегиями `sudo`:

а) Проверка обновлений ОС

Для проверки обновлений ОС, если это не было сделано при установке другого ПО Системы, выполните следующие команды:

```
sudo apt update
```

```
sudo apt upgrade
```

б) Установка Erlang

RabbitMQ требует, чтобы в системе был установлен Erlang. Выполните следующую команду, чтобы установить Erlang:

```
wget https://packages.erlang-solutions.com/erlang/debian/pool/esl-erlang_23.1.5-1~debian~stretch_amd64.deb
```

```
sudo dpkg -i esl-erlang_23.1.5-1~debian~stretch_amd64.deb
```

Затем обновите список системных пакетов и установите Erlang:

```
sudo apt update
```

```
sudo apt install erlang erlang-nox
```

в) Установка RabbitMQ на Debian 10

Добавьте в систему репозиторий RabbitMQ apt. Кроме того, необходимо импортировать ключ подписи RabbitMQ:

```
sudo add-apt-repository 'deb http://www.rabbitmq.com/debian/ testing main'
```

```
wget -O- https://www.rabbitmq.com/rabbitmq-release-signing-key.asc | sudo apt-key add -
```

После этого обновите apt-cache и установите сервер RabbitMQ в системе:

```
sudo apt update
```

```
sudo apt install rabbitmq-server
```

После успешной установки используйте следующие команды, чтобы включить службу RabbitMQ в системе. Также запустите сервис RabbitMQ:

```
sudo systemctl enable rabbitmq-server
```

```
sudo systemctl start rabbitmq-server
```

г) Создание пользователя на RabbitMQ

Создайте учетную запись администратора на сервере RabbitMQ, используя следующие команды. Замените пароль своим паролем:

```
sudo rabbitmqctl add_user admin password
```

```
sudo rabbitmqctl set_user_tags admin administrator
```

```
sudo rabbitmqctl set_permissions -p / admin ".*" ".*" ".*"
```

д) Настройка панели управления RabbitMQ

При необходимости можно включить веб-панель управления RabbitMQ для упрощения управления:

```
sudo rabbitmq-plugins enable rabbitmq_management
```

Примечание: RabbitMQ будет доступен по HTTP-порту 15672 по умолчанию.

2.4.9 Установка Kafka

а) Создание пользователя для Kafka

Поскольку Kafka может обрабатывать запросы через сеть, лучше всего создать для этого отдельного пользователя. Это позволит минимизировать ущерб для системы в случае взлома сервера Kafka.

Выполните вход с помощью пользователя без прав root с привилегиями sudo и создайте пользователя kafka с помощью команды useradd:

```
sudo useradd kafka -m
```

Флаг -m гарантирует, что для пользователя будет создана домашняя директория. Домашняя директория /home/kafka будет выступать в качестве директории рабочего пространства для последующего выполнения команд.

Установите пароль с помощью команды passwd:

```
sudo passwd kafka
```

Введите пароль для этого пользователя.

Затем добавьте пользователя kafka в группу sudo с помощью команды adduser для предоставления ему необходимых привилегий для установки зависимостей Kafka:

```
sudo adduser kafka sudo
```

Новый пользователь kafka готов к работе.

Выполните вход в учетную запись с помощью команды su:

```
su -l kafka
```

После создания пользователя для Kafka можно перейти к загрузке и извлечению двоичных файлов Kafka.

б) Загрузка и извлечение двоичных файлов Kafka

Создайте директорию в /home/kafka с названием Downloads, чтобы сохранить там загруженные данные:

```
mkdir ~/Downloads
```

Затем установите curl с помощью команды apt-get для получения возможности загрузки удаленных файлов:

```
sudo apt-get update && sudo apt-get install curl
```

После получения соответствующего предупреждения введите Y для подтверждения загрузки curl (используется для загрузки двоичных файлов Kafka):

```
curl "https://archive.apache.org/dist/kafka/2.1.1/kafka_2.11-2.1.1.tgz" -o ~/Downloads/kafka.tgz
```

Создайте директорию с названием kafka и замените ее на эту директорию. Она будет служить базовой директорией для установки Kafka:

```
mkdir ~/kafka && cd ~/kafka
```

Извлеките загруженный архив, с помощью команды tar (в директорию ~/kafka/):

```
tar -xvzf ~/Downloads/kafka.tgz --strip 1
```

После успешной загрузки и извлечения двоичных файлов мы можем перейти к настройке сервера Kafka.

в) Настройка сервера Kafka

Поведение Kafka по умолчанию не позволяет удалять название темы, категории, группы или ветки, где были опубликованы сообщения. Для изменения нужно отредактировать файл конфигурации.

Опции конфигурации Kafka указаны в файле server.properties. Откройте файл в nano или другом редакторе:

```
nano ~/kafka/config/server.properties
```

Добавьте настройку, которая позволит удалять темы Kafka. Добавьте в конец файла следующую строку:

```
~/kafka/config/server.properties  
...  
group.initial.rebalance.delay.ms  
  
delete.topic.enable = true
```

Рисунок 6 – Редактирование файла server.properties

Сохраните файл и закройте nano. После настройки Kafka можно создать файлы элементов systemd для запуска и активации Kafka при загрузке системы.

2.5 Установка и настройка прочего системного ПО

2.5.1 Установка открытой универсальной платформы разработки .NET

Для установки .NET на сервер `seed_platform` выполните следующие действия:

- а) Добавление ключа подписывания пакета Майкрософт в список доверенных ключей и репозитория пакетов

Откройте терминал и выполните следующие команды:

```
wget https://packages.microsoft.com/config/debian/10/packages-microsoft-prod.deb -O
packages-microsoft-prod.deb
sudo dpkg -i packages-microsoft-prod.deb
rm packages-microsoft-prod.deb
```

- б) Установка пакета SDK

Пакет SDK для .NET позволяет разрабатывать приложения с помощью .NET. При установке пакета SDK для .NET не нужно устанавливать соответствующую среду выполнения. Чтобы установить пакет SDK для .NET, выполните приведенные ниже команды:

```
sudo apt-get update && \
sudo apt-get install -y dotnet-sdk-7.0
```

- в) Установка среды выполнения

Среда выполнения ASP.NET Core позволяет запускать приложения, созданные с помощью версии .NET без поддержки среды выполнения. Приведенные ниже команды позволяют установить среду выполнения ASP.NET Core, которая больше всего совместима с .NET. В терминале выполните приведенные ниже команды.

```
sudo apt-get update && \
sudo apt-get install -y aspnetcore-runtime-7.0
```

В качестве альтернативы среде выполнения ASP.NET Core можно установить среду выполнения .NET без поддержки ASP.NET Core. Для этого в приведенной выше команде замените `aspnetcore-runtime-7.0` на `dotnet-runtime-7.0`.

```
sudo apt-get install -y dotnet-runtime-7.0
```

2.5.2 Установка системы контейнеризации Mesosphere DC/OS

Для установки системы контейнеризации Mesosphere DC/OS на сервер `seed_platform` выполните следующие операции:

1. С узла `bootstrap` запустите сценарий установки DC/OS для создания файла сборки DC/OS. Файлы сборки создаются в каталоге `./genconf/serve/`.

Для просмотра всех команд сценария автоматической установки используйте флаг `dcos_generate_config.sh --help`.

```
sudo bash dcos_generate_config.sh
```

После выполнения данной команды структура каталога должна выглядеть как показано ниже:

```
├── dcos-genconf.<HASH>.tar
├── dcos_generate_config.sh
├── genconf
│   ├── config.yaml
│   └── ip-detect
```

2. Запустите следующую команду из домашнего каталога, чтобы опубликовать установочные файлы DC/OS в Nginx:

```
sudo docker run -d -p <port>:80 -v $PWD/genconf/serve:/usr/share/nginx/html:ro nginx
```

3. На каждом узле `master` последовательно запустите следующие команды:

- a Создайте новый каталог и перейдите в него:

```
mkdir /tmp/dcos && cd /tmp/dcos
```

- b Скачайте файл установки DC/OS:

```
curl -fsSL https://downloads.dcos.io/dcos/stable/dcos_generate_config.sh
```

- c Установите DC/OS на узлах `master`: `sudo bash dcos_install.sh master`

DC/OS может показывать сообщения об ошибках, пока все узлы `master` не будут настроены.

4. На каждом узле-агенте последовательно запустите следующие команды:

- a) Создайте новый каталог и перейдите в него:
`mkdir /tmp/dcos && cd /tmp/dcos`
- b) Скачайте файл установки DC/OS:
`curl -fsSL https://downloads.dcos.io/dcos/stable/dcos_generate_config.sh`
- c) Установите DC/OS на узлах-агентах:
 - Команда для частных узлов-агентов: `sudo bash dcos_install.sh slave`
 - Команда для публичных узлов-агентов: `sudo bash dcos_install.sh slave_public`

5. Отслеживать установку можно через Exhibitor на странице <http://<ip-узла-master>:8181/exhibitor/v1/ui/index.html>. Весь процесс может занять до 10 минут. В интерфейсе Exhibitor можно наблюдать подключенность узлов master (они отмечены зеленой иконкой).

6. Запустите веб-интерфейс DC/OS по адресу <http://<публичный-ip-узла-master>/>.

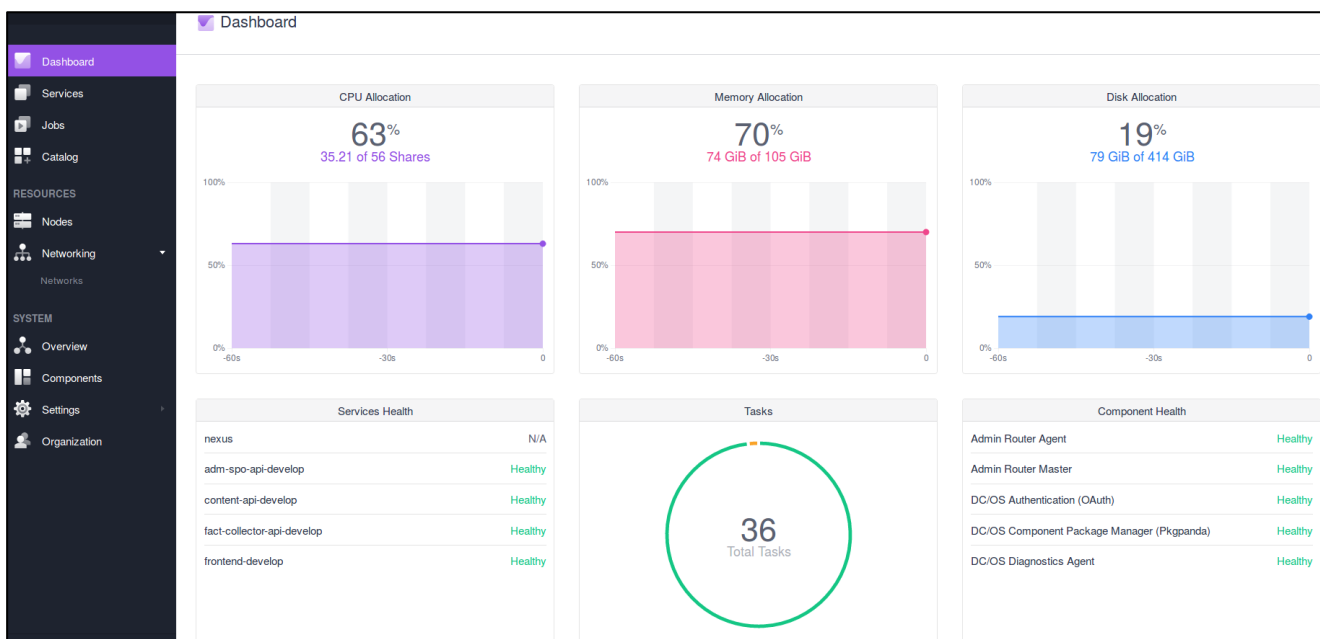


Рисунок 7 – UI DS/OS

2.5.3 Установка балансировщика нагрузки Marathon-lb

Установка балансировщика нагрузки Marathon с помощью веб-интерфейса Mesosphere DC/OS:

- а) Выберите Universe (Вселенная).
- б) Выполните поиск по запросу "Marathon-LB".
- в) Выберите Install (Установить).

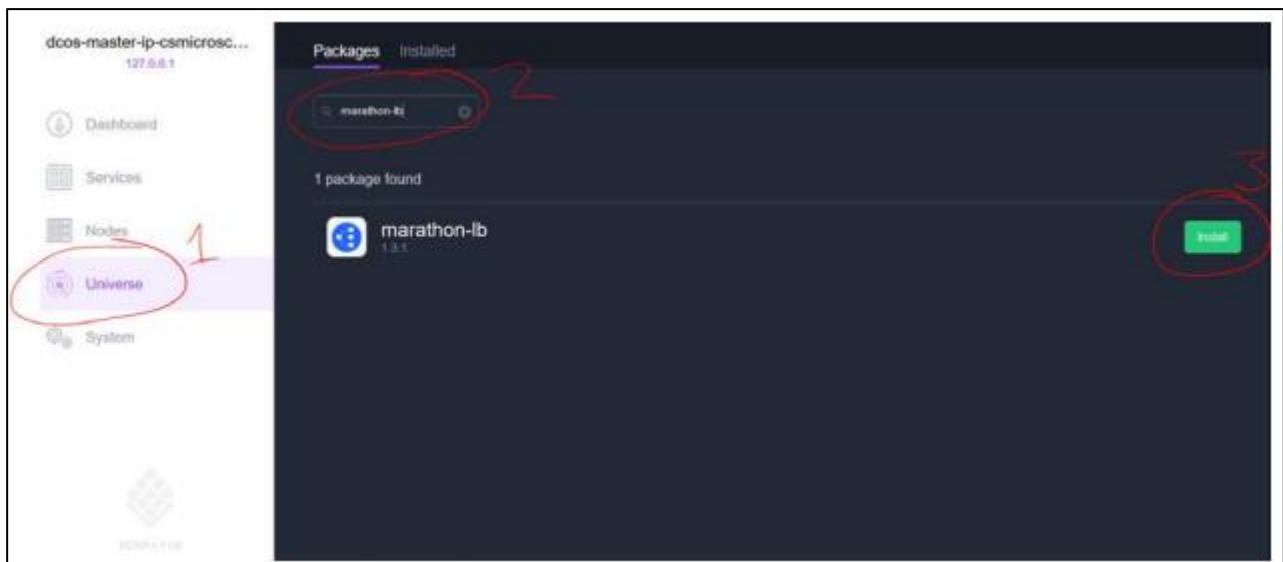


Рисунок 8 – Установка балансировщика нагрузки Marathon с помощью веб-интерфейса DC/OS

2.5.4 Установка системы мониторинга Prometheus

Prometheus - это бесплатное программное приложение, используемое для мониторинга событий и оповещения. Он записывает метрики в реальном времени в базу данных временных рядов, построенную с использованием модели HTTP-запроса, с гибкими запросами и оповещениями в режиме реального времени. В Системе используется для мониторинга внешних связей Системы.

Для установки Prometheus на сервер seed_platform выполните следующие действия:

1. Создайте пользователя Prometheus:

```
adduser --no-create-home --disabled-login --shell /bin/false --gecos "Prometheus Monitoring User" prometheus
```

2. Создайте директории для хранения данных и конфигурации:

```
mkdir -p /opt/data/prometheus
mkdir -p /opt/configs/
```

3. Добавьте права на директорию где будут храниться данные:

```
chown -R prometheus:prometheus /opt/data/Prometheus
```

4. Скачайте актуальную версию Prometheus:

```
wget https://github.com/prometheus/prometheus/releases/download/v2.33.4/prometheus-2.33.4.linux-amd64.tar.gz
```

5. Распакуйте скачанный архив в opt:

```
tar xzf prometheus-2.33.4.linux-amd64.tar.gz -C /opt/
```

За счет использования при запуске указания конкретной версии, можно обеспечить обновление до новой версии путем изменения одного файла и перезапуска сервиса. Так же при необходимости можно будет вернуться на предыдущую версию.

6. Создайте файл сервиса `/etc/systemd/system/prometheus.service` со следующим текстом:

```
[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/opt/prometheus-2.33.4.linux-amd64/prometheus \
  --config.file /opt/configs/prometheus.yml \
  --storage.tsdb.path /opt/data/prometheus/ \
  --web.enable-admin-api --web.enable-lifecycle \
  --web.console.templates=/opt/prometheus/consales \
  --web.console.libraries=/opt/prometheus/console_libraries \
  --storage.tsdb.retention.time=180d

[Install]
WantedBy=multi-user.target
```

7. Создайте пустой файл конфигурации:

```
> /opt/configs/prometheus.yml
```

8. Установите и запустите сервис:

```
systemctl daemon-reload
systemctl enable prometheus
systemctl restart prometheus
```

9. Откройте порт 9090:

```
ufw allow 9090/tcp
```

Для проверки установки откройте веб-консоль Prometheus в браузере.

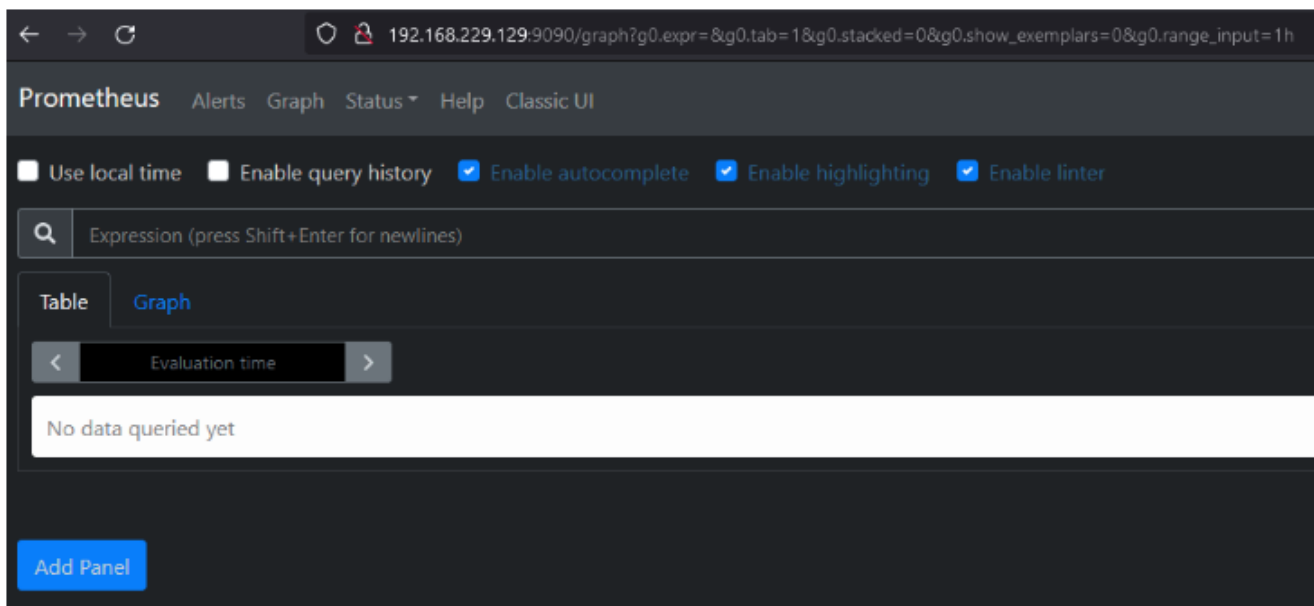


Рисунок 9 – Пользовательский интерфейс Prometheus

Базовый конфиг /opt/configs/prometheus.yml (без экспортеров):

```
global:
  scrape_interval: 30s
  scrape_timeout: 10s
  evaluation_interval: 20s
scrape_configs:
  - job_name: pushgateway
    honor_labels: true
    static_configs:
      - targets:
        - '<ip>:9091'
```

<ip> - IP адрес сетевого интерфейса.

10. Для применения настроек перезагрузите сервис:

```
systemctl restart prometheus
```

2.5.5 Установка и настройка панели визуализации данных Kibana

Пакет Kibana используется для визуализации данных, полученных из Elasticsearch. Репозитории и публичный ключ для установки Kibana будут такими же, как в установке Elasticsearch.

Для установки Kibana на сервер seed_platform выполните следующие действия:

а) Установка Kibana

Предварительно необходимо скопировать публичный ключ репозитория:

```
# rpm --import https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

Затем требуется подключить репозиторий Kibana:

```
# mcedit /etc/yum.repos.d/kibana.repo
```

```
[kibana-7.x]
```

```
name=Kibana repository for 7.x packages
```

```
baseurl=https://artifacts.elastic.co/packages/7.x/yum
```

```
gpgcheck=1
```

```
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

```
enabled=1
```

```
autorefresh=1
type=rpm-md
```

Установка пакета Kibana осуществляется командой:

```
# yum install kibana
```

В завершение установки требуется добавить Elasticsearch в автозагрузку и запустить его с дефолтными настройками:

```
# systemctl daemon-reload
# systemctl enable kibana.service
# systemctl start kibana.service
```

Проверку работы службы можно осуществить командой:

```
# systemctl status kibana.service
```

По умолчанию, Kibana слушает порт 5601.

Примечание: Кибана стартует длительное время. Подождите примерно минуту и проверяйте.

Проверку работы Kibana можно осуществить командой:

```
# netstat -tulnp | grep 5601
```

На экране отобразится информация, подобная следующей:

```
tcp    0    0 127.0.0.1:5601    0.0.0.0:*        LISTEN    20746/node
```

б) Настройка Kibana

Файл с настройками Kibana располагается по пути - /etc/kibana/kibana.yml.

Для разрешения прослушки внешнего интерфейса и внешних подключений требуется изменить параметр server.host, указав IP адрес сервера:

```
server.host: "10.128.0.29"
```

Для прослушки всех интерфейсов требуется указать в качестве адреса 0.0.0.0.

После изменения настроек Kibana сервис требуется перезапустить:

```
# systemctl restart kibana.service
```

Теперь можно зайти в веб интерфейс по адресу <http://10.128.0.29:5601>.

2.5.6 Установка и настройка веб-сервера Nginx

Nginx — веб-сервер и почтовый прокси-сервер.

Для установки веб-сервера Nginx на сервер `seed_platform` под управлением Debian выполните следующие действия от имени пользователя `root` или пользователя с привилегиями `sudo`:

а) Установка Nginx

Nginx доступен в стандартных репозиториях Debian, так что его можно установить прямо оттуда, используя пакетный менеджер `apt`.

Нужно выполнить стандартные действия - сначала обновить локальный индекс пакетов:

```
$ sudo apt update
```

А затем уже установить сам `nginx`:

```
$ sudo apt install nginx
```

б) Настройка профиля приложений

Введите команду для просмотра всех доступных профилей приложений на сервере:

```
$ sudo ufw app list
```

На экране отобразится список профилей приложений:

```
Available applications:
```

```
...
```

```
Nginx Full
```

```
Nginx HTTP
```

```
Nginx HTTPS
```

```
...
```

Для Nginx обычно доступны 3 профиля:

- **Nginx Full**: профиль открывает два порта – 80 (обычный, незашифрованный веб-трафик) и 443 (TLS/SSL зашифрованный трафик);
- **Nginx HTTP**: профиль открывает только порт 80 (обычный, незашифрованный веб-трафик);

- **Nginx HTTPS:** профиль открывает только порт 443 (TLS/SSL зашифрованный трафик).

Лучше всего использовать наиболее ограниченный профиль, но при этом разрешающий необходимый трафик. Так как на сервере еще не настроен SSL, то на данном этапе необходимо открыть порт 80, который разрешает трафик для незашифрованного трафика HTTP.

Для этого нужно выполнить команду:

```
$ sudo ufw allow 'Nginx HTTP'
```

Затем проверьте состояние UFW:

```
$ sudo ufw status
```

На экране отобразятся заданные правила (HTTP разрешен):

```
Status: active
```

To	Action	From
--	-----	----
OpenSSH	ALLOW	Anywhere
Nginx HTTP	ALLOW	Anywhere
OpenSSH (v6)	ALLOW	Anywhere (v6)
Nginx HTTP (v6)	ALLOW	Anywhere (v6)

в) Проверка работы веб-сервера

После завершения процесса установки ОС запустит nginx. То есть веб-сервер уже будет работать.

Это можно проверить следующей командой:

```
$ systemctl status nginx
```

На экране отобразится следующая информация:

```
nginx.service - A high performance web server and a reverse proxy server
```

```
Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
```

Active: active (running) since Wed 2019-07-12 12:52:54 UTC; 4min 23s ago

Docs: man:nginx(8)

Main PID: 3942 (nginx)

Tasks: 3 (limit: 4719)

Memory: 6.1M

CGroup: /system.slice/nginx.service

└─3942 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;

└─3943 nginx: worker process

└─3944 nginx: worker process

Это значит, что сервис успешно запущен.

Далее загрузите начальную страницу Nginx, перейдя по IP-адресу сервера Системы.

Если на экране отобразилась информация об успешной установке Nginx, значит сервер работает корректно.

г) Настройка веб-сервера Nginx

Сервер Nginx предназначен для проксирования запросов к сайту partner.companu.ru на 2 сервера приложений (программные интерфейсы Системы), при этом запросы балансируются по принципу Round-Robin. Также сервер может обслуживать другие сайты без балансировки нагрузки.

Конфигурация сервиса представлена далее отдельными информационными блоками.

Параметры балансировки: запросы будут распределяться по указанным серверам по принципу Round-Robin:

```
upstream backend
{ server IP_APPSERVER : [PORT];
server IP_APPSERVER : [PORT]; }
```

Выход из строя сервера определяется по невозможности установить с ним TCP-соединение, критерий выхода из строя - "хотя бы один запрос", после этого сервер на 10 секунд помечается как сбойный и запросы на него не отправляются.

Описание сайта в случае, если клиент обращается по указанным доменным именам на 80 порт, то в ответ получает 301 код с редиректом на HTTPS:

```
server { listen [PORT]; server_name
partner.company.ru api.partner.company.ru;
return 301 https://$host$request_uri; }
```

Описание сайта с указанием используемых сертификатов:

```
server { listen 443 ssl; server_name partner.company.ru api. partner.company.ru im.local api.
im.local IP_APPSERVER; ssl_certificate /etc/nginx/ssl/ partner.company.ru.crt;
ssl_certificate_key /etc/nginx/ssl/ partner.company.key; }
```

Список шифров, считающихся безопасными (нужно периодически проверять):

```
ssl_ciphers "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-
AES256SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-
ECDSAAES256-SHA:DHE-DSS-AES256-GCM-SHA384:DHE-DSS-AES256-SHA256:DHE-DSS-
AES256-
SHA:DHE-DSS-CAMELLIA256-SHA:ECDH-RSA-AES256-GCM-SHA384:ECDH-
ECDSAAES256-GCM-SHA384:ECDH-RSA-AES256-SHA384:ECDH-ECDSA-AES256-
SHA384:ECDH-
RSA-AES256-SHA:ECDH-ECDSA-AES256-SHA:AES256-GCM-SHA384:AES256-
SHA256:AES256-SHA:CAMELLIA256-SHA:PSK-AES256-CBC-SHA:ECDHE-RSA-
AES128GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-
SHA256:ECDHE-
ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:DHE-
DSSAES128-GCM-SHA256:DHE-DSS-AES128-SHA256:DHE-DSS-AES128-SHA:ECDHE-RSA-
DESCBC3-SHA:ECDHE-ECDSA-DES-CBC3-SHA:DHE-DSS-SEED-SHA:DHE-DSS-
CAMELLIA128-
```

```
SHA:EDH-DSS-DES-CBC3-SHA:ECDH-RSA-AES128-GCM-SHA256:ECDH-ECDSA-AES128-  
GCM-SHA256:ECDH-RSA-AES128-SHA256:ECDH-ECDSA-AES128-SHA256:ECDH-  
RSAES128-SHA:ECDH-ECDSA-AES128-SHA:ECDH-RSA-DES-CBC3-SHA:ECDH-ECDSA-  
DESCBC3-SHA:AES128-GCM-SHA256:AES128-SHA256:AES128-SHA:SEED-  
SHA:CAMELLIA128-  
SHA:DES-CBC3-SHA:IDEA-CBC-SHA:PSK-AES128-CBC-SHA:PSK-3DES-EDE-  
CBCSHA:KRB5-IDEA-CBC-SHA:KRB5-DES-CBC3-SHA:KRB5-IDEA-CBC-MD5:KRB5-DES-  
CBC3-  
MD5:ECDFE-ECDSA-RC4-SHA:ECDH-RSA-RC4-SHA:ECDH-ECDSA-RC4-SHA:PSK-  
RC4SHA:KRB5-RC4-SHA:KRB5-RC4-MD5";
```

Принудительное использование клиентами только тех шифров, которые описаны выше:

```
ssl_prefer_server_ciphers on;
```

Список используемых протоколов, считающихся безопасными (нужно периодически проверять):

```
ssl_protocols TLSv1.1 TLSv1.2;
```

Примечание: Приведенные настройки могут вызвать проблемы (невозможность соединения) на стороне клиентов в тех случаях, если у них будет стоять старый браузер, не поддерживающий указанные протоколы и шифры.

```
location / {  
    proxy_read_timeout 1200;  
    proxy_connect_timeout 1200;  
    proxy_pass http://backend;  
    proxy_set_header Host $host;  
    proxy_set_header X-Real-IP $remote_addr; // проброс заголовков на бэкенд  
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for; // проброс заголовков на бэкенд  
    proxy_set_header X-HTTPS yes; // проброс заголовков на бэкенд  
    access_log /var/log/nginx/upstream-access-onlya.log upstreamlog; }
```

```
error_page 404 /404.html; location
= /40x.html {
}
```

```
error_page 500 502 503 504 /50x.html; location
= /50x.html { }
```

2.5.7 Установка системы мониторинга Zabbix

Zabbix - это свободно распространяемое программное обеспечение с открытым исходным кодом, часто используемое для мониторинга ИТ-инфраструктуры. В Системе Zabbix используется для внутреннего мониторинга в связке с БД PostgreSQL и веб-сервером Nginx.

Для установки Zabbix на сервер `seed_platform` выполните следующие действия:

а) Установка и настройка Zabbix Server

Для установки требуется добавить официальный репозиторий Zabbix в список разрешенных репозиториях, сделать это можно выполнив следующие команды:

```
$ wget https://repo.zabbix.com/zabbix/6.2/debian/pool/main/z/zabbix-release/zabbix-
release_6.2-1+debian11_all.deb
$ sudo dpkg -i zabbix-release_6.2-1+debian11_all.deb
```

Далее установите сервер Zabbix, веб интерфейс, пакеты агента используя следующие команды:

```
$ sudo apt update
$ sudo apt -y install zabbix-server-pgsql zabbix-frontend-php php7.4-pgsql zabbix-nginx-conf
zabbix-sql-scripts zabbix-agent2
```

Если в процессе установки автоматически установился `apache2`, удалите его:

```
$ sudo apt -y remove apache2
$ sudo apt -y autoremove
```

б) Первичная настройка PostgreSQL для Zabbix

Отредактируйте конфигурационный файл `pg_hba.conf`, включив авторизацию по паролю для локальных соединений

```
...
#host all all 127.0.0.1/32 scram-sha-256
```

```
host all all 127.0.0.1/32 md5
# IPv6 local connections:
#host all all ::1/128 scram-sha-256
host all all ::1/128 md5
...
```

Перезапустите сервис:

```
$ sudo systemctl restart postgresql
```

в) Настройка PHP-FPM

Смена Timezone:

```
$ sudo nano /etc/zabbix/php-fpm.conf (или любой другой текстовый редактор)
```

...

```
php_value[date.timezone] = Europe/Moscow
```

Смена прав на файл zabbix.sock:

```
$ sudo chmod 777 /run/php/zabbix.sock
```

Перезапуск сервиса:

```
$ sudo systemctl restart php7.4-fpm
```

г) Дополнительная настройка Nginx

Редактирование конфигурационного файла zabbix.conf:

```
listen 80;
listen [::]:80;
server_name _;
# listen 8080;
# server_name example.com;

root /usr/share/zabbix;
index index.php;
client_max_body_size 100M;
...
```

Отключение конфигурационного файла по умолчанию:

```
$ sudo mv /etc/nginx/conf.d/default.conf /etc/nginx/conf.d/default.conf.disabled
```

Перезапуск сервиса:

```
$ sudo systemctl restart nginx
```

д) Окончательная настройка PostgreSQL для Zabbix

Создание пользователя:

```
$ sudo -u postgres createuser --pwprompt zabbix
```

```
Enter password for new role: mysuperpass
```

Создание БД:

```
$ sudo -u postgres createdb -O zabbix zabbix
```

Загрузка данных:

```
$ zcat /usr/share/doc/zabbix-sql-scripts/postgresql/server.sql.gz | sudo -u zabbix psql zabbix
```

е) Настройка Zabbix

Редактирование конфигурационного файла `zabbix_server.conf`, добавление пароля пользователя:

```
$ sudo nano /etc/zabbix/zabbix_server.conf
```

```
...
```

```
DBPassword=mysuperpass
```

```
...
```

Перезапуск сервисов, добавление их в автозагрузку:

```
$ sudo systemctl restart zabbix-server zabbix-agent2 nginx php7.4-fpm
```

```
$ sudo systemctl enable zabbix-server zabbix-agent2 nginx php7.4-fpm
```

Zabbix server установлен.

Завершение установки в web-интерфейсе:

```
Default login: Admin
```

```
Default pass: zabbix
```

2.6 Установка и настройка приложений УМДС

2.6.1 Подготовка системы управления АСУ

По выбранной архитектуре, сервисы УМДС запускаются через Nginx ingress controller, работающий как сервис NodePort (запросы приходят на внешний балансировщик и после маршрутизируются в этот Nginx ingress controller с балансировкой нагрузки по всем рабочим нодам).

Для настройки системы управления АСУ (см. п. 2.3.1) для релиза модулей УМДС необходимо выполнить следующие шаги:

а) Для дополнительной настройки параметров Nginx ingress controller (значения проху-body-size и пр.) нужно выполнить скрипт: <https://git.gateline.net/b2b/onelia/onelia-k8s/-/blob/master/mds-prod/apply-ingress-nginx-config.sh>

б) На рабочих узлах системы управления АСУ требуется скопировать конфигурационные файлы в папку /mnt/data: <https://git.gateline.net/b2b/onelia/onelia-k8s/-/tree/master/mds-prod/volume-data>

в) Создать deps неймспейс:

```
./kubectl create namespace deps
```

г) Для отправки логов с узла системы управления АСУ в ПО служебного мониторинга (например, LogViewer) нужно задеплоить сервис rc-logviewerapi в deps неймспейс:

```
https://git.gateline.net/b2b/onelia/onelia-k8s/-/blob/master/mds-prod/helm-charts/deploy-rc-logviewer.sh
```

д) Для настройки нового источника логов (LogViewer.Api) для соответствующего сайта (LogViewer.UI) требуется соответствующим образом изменить конфигурационный файл (LogViewer.json):

```
{  
  "LogServers": [  
    ...  
    {  
      "Name": "K-N1",  
      "Host": "172.30.20.51:5221"  
    }  
  ]  
}
```

После перегрузки страницы просмотра лог-файлов (LogViewer.UI) добавится новый источник логов K-N1.

Примечание: В конфигурационном файле (LogViewer.json) нужно прописать все рабочие узлы системы управления АСУ.

е) На сервере TS (Terminal Server), с которого будет идти развертка приложений УМДС, требуется сделать `git clone` для следующих репозиторий:

- <https://git.gateline.net/b2b/onelia/onelia-k8s> в папку `/root/scripts/onelia-k8s`.
- <https://git.gateline.net/sys/ansible> в папку `/root/scripts/ansible`.

9. Развертка приложений УМДС (см. п. 2.6.2).

Примечание: После добавления нового рабочего узла в системе управления АСУ необходимо скопировать на нее конфигурационные файлы (см. п. «б» текущей процедуры) и установить на ней Consul-клиент (см. п. 2.3.3). Сервис `rc-logviewerapi` установится автоматически средствами системы управления АСУ (т.к. `rc-logviewerapi` ставится как `DaemonSet`).

2.6.2 Развертка приложений УМДС

Для развертки приложений УМДС необходимо выполнить следующие шаги:

а) Сделать `Git pull` для репозиторий `/root/scripts/ansible` и `/root/scripts/onelia-k8s` (на TS), и задать значения версии и неймспейса:

```
export NEW_VER=1.118.0
```

```
export NEW_NS=mds-1-118-0
```

б) Через Ansible скопировать конфигурационные файлы и значения конфигов в Consul.

Например из папки `/root/scripts/ansible/b2b` на TS:

```
ansible-playbook ./deploy-mds-prod-new-version.yml -i ../inventory/b2b-mds-prod.ini -u root --extra-vars "new_version=$NEW_VER old_consul_version=K8S_MDS-1.117.0"
```

Параметры:

- *new_version* - новая версия продукта (X.Y.Z);
 - *old_consul_version* - версия в Consul, с которой значения копируются в новую версию
- Для текущего примера название новой версии в Consul будет ``K8S_MDS-1.118.0``.

в) Настроить конфигурационные файлы в Consul для новой версии продукта (согласно инструкции по выкладке).

г) Создать и настроить новый неймспейс в системе управления АСУ. Например:

```
./configure-new-product-namespace.sh "$NEW_NS"
```

Примечание: для значений версии в неймспейсе все "." необходимо заменить на "-".

Теперь в системе управления АСУ создан неймспейс `mds-1-118-0`, в который можно развертывать сервисы заданной версии продукта.

д) Развернуть сервисы в системе управления АСУ:

– прописать список имен сервисов для развертки в файле `/root/scripts/onelia-k8s/temp/apps_list.txt` (на TS). Сервисы, которые не нужно развертывать, можно закомментировать через `#`.

– запустить скрипт развертки из папки /root/scripts/onelia-k8s/mds-prod/helm-charts на TS:
./get-and-deploy-apps.sh ../../temp/apps_list.txt "\$NEW_VER" "\$NEW_NS"

Примечание: для скрипта `get-and-deploy-apps.sh` задаются версия продукта и неймспейс.

В результате развертки запустятся сервисы с URL: `\${APP}-mds-\${APP_VER}.gds.prod`

е) Если необходимо, то нужно отскалировать (scale) в системе управления АСУ количество реплик сервиса до нужного значения реплик.

ж) Проверить, что пишутся логи в новую папку версии продукта (`/mnt/data/mds/prod-{{new_version}}/logs/*`) и настроить сбор логов через систему служебного мониторинга (LogViewer.Api) из этой папки (если такой сбор не был настроен ранее).

з) Проверить и настроить доступ к URL `\${APP}-mds-\${APP_VER}.gds.prod`.

и) Передать новые URL для тестирования (если производилось обновление релиза УМДС в данный момент одновременно работают и старые, и новые версии сервисов).

к) После успешного тестирования переключить основные URL на новую версию УМДС.

л) Осуществить Smoke-test из локальной сети:

```
curl http://172.20.20.21:31000 -H 'Host: dksyncservice-mds-1-118-0.gds.prod'
```

```
curl --fail --resolve pdfgenerator-mds-1-118-0.gds.prod:31000:172.20.20.21  
http://pdfgenerator-
```

```
mds-1-118-0.gds.prod:31000/api/v1/generate >/dev/null
```

м) Изучить дашборд для ingress-controller в ПО для мониторинга (Grafana):
<http://172.20.20.21:32330/>

н) После успешного переключения на новую версию УМДС можно отскалировать до 0 количество реплик старой версии. Так же, через какое-то время, можно удалить старый helm-релиз.

3 Эксплуатация, техническое обслуживание, ремонт и хранение компонентов системы

Специального регламентного обслуживания Системы не требует.

Контроль над используемым дисковым пространством БД осуществляется штатными средствами администрирования Linux, таким как logrotate.

Logrotate - приложение, разработанное для облегчения управления лог-файлами. Утилита позволяет в автоматическом режиме архивировать, удалять, очищать и отправлять на email лог-файлы. Этот процесс обычно называется ротацией лог файлов. Logrotate может быть настроен на ежедневную, еженедельную или ежемесячную ротацию.

Контроль над приложением осуществляется штатными средствами администрирования Linux.

Контроль за изменением БД осуществляет администратор базы данных с помощью системы мониторинга Zabbix. Рекомендуется использовать следующие плагины:

- pgbadger (позволяет осуществлять аналитику по БД, см. <https://itc-life.ru/nastrojka-sbora-statistiki-postgresql-s-pomoshhyu-pgbadger/>);
- mamonsu (позволяет осуществлять мониторинг БД, см. <https://postgrespro.ru/docs/postgrespro/11/mamonsu>);
- powa (позволяет собирать статистику и строить по ней аналитику, см. <https://powa.readthedocs.io/en/latest/quickstart.html#install-powa-from-packages-on-rhel-centos>).

4 Действия при возникновении ошибок и неполадок

В случае возникновения любой критической ошибки при использовании системы требуется:

- ввести систему в сервисный режим;
- связаться с разработчиком.

Критической ошибкой в Системе является любая невозможность использования основного функционала.