

Государственное бюджетное профессиональное образовательное
«Южно-Уральский государственный колледж»

РАССМОТРЕНО

Председатель ПЦК

«Информационных технологий»

_____/ Назарова Н.А.

«10» мая 2023 г.

**КОМПЛЕКТ КОНТРОЛЬНО-ОЦЕНОЧНЫХ СРЕДСТВ
УЧЕБНОЙ ПРАКТИКИ**

**ПМ.12. АВТОМАТИЗАЦИЯ ПЛАНИРОВАНИЯ, РАЗВЕРТЫВАНИЯ,
МАСШТАБИРУЕМОСТИ, БАЛАНСИРОВКИ НАГРУЗКИ И
ДОСТУПНОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ**

образовательной программы по специальности СПО

09.02.07 Информационные системы и программирование

Квалификация: Разработчик веб и мультимедийных технологий

Челябинск, 2022

Разработчики:

ГБПОУ «ЮУГК»

(место работы)

Преподаватель

(занимаемая должность)

Н.А. Назарова

(инициалы, фамилия)

Эксперты:

ЗАО ЮУИК «Трейд-Альянс»

(место работы)

Руководитель отдела А.Ю. Скворцов

информационных

(инициалы, фамилия)

технологий (занимаемая

должность)

СОДЕРЖАНИЕ

1. Общие положения.....	3
2. Комплект КИМ для промежуточной аттестации.....	28

1. Общие положения

Комплект КИМ предназначен для оценки готовности обучающегося к выполнению основного вида деятельности (ВД) Использование современные системы оркестрации, основной профессиональной образовательной программы (далее ОПОП) по специальности 09.02.07 Информационные системы и программирование.

КОСы позволяют оценивать: сформированность общих и профессиональных компетенций в соответствии с показателями.

Комплект контрольно-оценочных средств позволяет оценивать:

1. Формирование элементов профессиональных компетенций (ПК) и элементов общих компетенций (ОК):

Таблица 1

Профессиональные и общие компетенции	Показатели оценки результата	Средства проверки (№№ заданий)
1	2	3
ОК 1. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам	правильно распознает задачу в профессиональном контексте, точно перечисляет методы работы в сфере ИТ	Экспертная оценка, дифференцированный зачёт
ОК 2. Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности	правильно определяет задачи для поиска информации; определять необходимые источники информации; планировать процесс поиска; правильно структурирует получаемую информацию; правильно выделяет наиболее значимое в перечне информации; точно оценивает практическую значимость результатов поиска; правильно и быстро оформляет результаты поиска	Экспертная оценка, дифференцированный зачёт
ОК 03. Планировать и реализовывать собственное профессиональное и личностное развитие	правильно определяет актуальность нормативно-правовой документации в профессиональной деятельности; правильно применяет современную научную профессиональную терминологию; определяет и выстраивает траектории профессионального развития и самообразования	Экспертная оценка, дифференцированный зачёт

ОК 4. Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами.	правильно организует работу коллектива и команды;	Экспертная оценка, дифференцированный зачёт
ОК 5. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста.	грамотно излагает свои мысли и оформляет документы по профессиональной тематике на государственном языке, проявляет толерантность в коллективе	Экспертная оценка, дифференцированный зачёт
ОК 6. Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей	правильно описывать значимость своей специальности	Экспертная оценка, дифференцированный зачёт
ОК 7. Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях.	соблюдает нормы экологической безопасности; определять направления ресурсосбережения в рамках профессиональной деятельности по специальности	Экспертная оценка, дифференцированный зачёт
ОК 8. Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности	использует физкультурно-оздоровительную деятельность для укрепления здоровья, достижения жизненных и профессиональных целей; применять рациональные приемы двигательных функций в профессиональной деятельности; пользоваться средствами профилактики перенапряжения характерными для данной специальности	Экспертная оценка, дифференцированный зачёт
ОК 9. Использовать информационные технологии в профессиональной деятельности	правильно применяет средства информационных технологий для решения профессиональных задач; правильно использует современное программное обеспечение	Экспертная оценка, дифференцированный зачёт
ОК.10. Пользоваться профессиональной документацией на государственном и иностранном языках.	точно понимает общий смысл четко произнесенных высказываний на известные темы (профессиональные и бытовые), понимает тексты на базовые профессиональные темы; участвует в диалогах на знакомые общие и	Экспертная оценка, дифференцированный зачёт

	<p>профессиональные темы; строит простые высказывания о себе и о своей профессиональной деятельности; кратко обосновывает и объясняет свои действия (текущие и планируемые); пишет простые связные сообщения на знакомые или интересующие профессиональные темы</p>	
<p>ОК 11. Использовать знания по финансовой грамотности, планировать предпринимательскую деятельность в профессиональной сфере.</p>	<p>выявляет достоинства и недостатки коммерческой идеи; презентовать идеи открытия собственного дела в профессиональной деятельности; оформляет бизнес-план; рассчитывает размеры выплат по процентным ставкам кредитования; определять инвестиционную привлекательность коммерческих идей в рамках профессиональной деятельности; презентует бизнес-идею; определяет источники финансирования</p>	<p>Экспертная оценка, дифференцированный зачёт</p>
<p>ПК 12.1 Осуществлять конфигурацию и планирование инфраструктуры с помощью современных систем оркестрации</p>	<p>правильно подготавливает и разворачивает инфраструктуры; точно и быстро планирует и конфигурирует сервера; правильно выделяет серверные ресурсы</p>	<p>Экспертная оценка, дифференцированный зачёт</p>
<p>ПК 13.1 Выявлять технические проблемы, возникающие в процессе эксплуатации баз данных и серверов.</p>	<p>точно масштабирует контейнеры согласно рабочим нагрузкам правильно осуществляет балансировку нагрузки сервера точно отслеживает состояние контейнеров правильно и точно устраняет неполадки в работе системы</p>	<p>Экспертная оценка, дифференцированный зачёт</p>
<p>ПК 13.2 Осуществлять администрирование отдельных компонент серверов.</p>	<p>обеспечивает безопасность сервера проводит анализ эффективности работы и оптимизации работы отдельных компонентов вносит изменения в конфигурацию программного обеспечения системы</p>	<p>Экспертная оценка, дифференцированный зачёт</p>

1.1.3. Освоение умений и усвоение знаний

Таблица 2.

Освоенные умения, усвоенные знания	Показатели оценки результата	№.№ заданий для проверки
1	2	3
Подготавливать и развертывать инфраструктуры	Умеет подготовить и правильно развернуть инфраструктуру	1-2
Планировать и конфигурировать сервера	Умеет правильно спланировать и сконфигурировать сервер	1-2
Выделять серверные ресурсы	Правильно выделяет серверные ресурсы	1-2
Масштабировать контейнеры сообразно рабочим нагрузкам	Умеет масштабировать контейнеры сообразно рабочим нагрузкам	1-2
Балансировать нагрузку сервера	Умеет балансировать нагрузку сервера	1-2
Отслеживать состояние контейнеров	Правильно отслеживает состояние контейнеров	3
Обеспечивать безопасность сервера	Умеет обеспечивать безопасность сервера	3, 6
Устранять неполадки в работе системы	Умеет устранить неполадки в работе системы	3, 6
Проводить анализ эффективности работы и оптимизации работы отдельных компонентов	Умеет провести анализ эффективности работы и оптимизации работы отдельных компонентов	3, 5, 6
Вносить изменения в конфигурацию программного обеспечения системы	Умеет правильно внести изменения в конфигурацию программного обеспечения системы	5, 6, 7
Понятия контейнер, микросервисный подход, DevOps-практика	Знает и точно называет понятия контейнер, микросервисный подход, DevOps-практика	4
Функционал программ-оркестраторов	Правильно перечисляет функционал программ-оркестраторов	4
Понятие декларативной модели управления объектами	Правильно называет понятие декларативной модели управления объектами	4
Команды Docker	Правильно перечисляет команды Docker	4
Набор вспомогательных утилит	Правильно и точно называет набор вспомогательных утилит	4, 5

1.2. Система контроля и оценки освоения программы междисциплинарного модуля

Таблица 4.

Элементы модуля, профессиональный модуль	Формы промежуточной аттестации
1	2
УП 12	Дифференцированный зачёт

2. Задания для оценки умений и усвоения знаний по УП 12.

Раздел 1. Современные системы оркестрации

Задание №1

Построить прототип программного продукта с использованием инструментов Twitter Bootstrap и Font Awesome. Прототип должен имитировать наиболее ключевую, с Вашей точки зрения, функциональность проектируемого приложения и быть понятным для предполагаемого заказчика.

Прототип должен быть выполнен в виде набора HTML страниц, связанных гиперссылками и оформленных при помощи вышеперечисленных инструментов.

Использование аналогов вместо Twitter Bootstrap и Font Awesome обоснованно допускается.

Критериями оценки прототипа являются:

1. Предполагаемое сходство с реальными приложениями
2. Использование реальных данных и реальных ситуаций
3. Удобство работы с прототипом для конечного пользователя
4. Понятность и близость к предметной области

Примерный ход выполнения лабораторной работы:

1. Выбор темы приложения и анализ предметной области. Допустим, целью прототипирования является приложение для организации научно-учебной деятельности. Производим краткий анализ предметной области, а также анализ пользовательских требований и пожеланий именно того самого предполагаемого пользователя, для которого планируется разработка прототипа. В нашем случае это будет сетевой инженер, в обязанности которого входит обслуживание, контроль и учет оборудования нескольких лабораторий.

2. Определяем основные обязанности выбранного пользователя. В случае, если пользователь в процессе работы формирует некоторые бумажные документы – обязательно снимаем с них копии. Лучшим решением при прототипировании является, по возможности, наиболее полное копирование структуры бумажных документов. В нашем случае это: - подсчет количества различного оборудования - его инвентаризация - учет срока службы каждой единицы оборудования, регистрация инцидентов - плановая и экстренная замена оборудования в случае поломки - ведение финансовой отчетности

3. Проектируем общую структуру приложения, разбиваем его на модули. В нашем случае один из вариантов декомпозиции приложения на модули будет выглядеть следующим образом:

- Планирование
- Операционный учет
- Отчетность

Эти пункты будут входить в главное меню прототипа наряду с набором следующих сервисных пунктов:

- Пользователь
- Справочники

Варианты заданий для реализации

1. Приложение «Бухгалтерия-онлайн»
2. Личный кабинет клиента банка
3. Приложение кредитного инспектора

4. Приложение для налоговой службы
5. Рабочее место преподавателя ВУЗа
6. Приложение для студента
7. Приложение для туриста
8. Интернет-представительство гостиницы
9. Приложение для управления производством мебели
10. Приложение для автоматизации сельского хозяйства
11. Приложение для управления слесарным цехом
12. Приложение вагоноремонтного депо
13. Библиотека финансовых инструментов для маклера
14. Приложение ЦУП космодрома
15. Система управления аэропортом

Задание №2

1. Что такое веб-сервер?
2. Примеры веб-серверов.
3. Что такое СУБД?
4. СУБД MySQL, OracleDatabase, PostgreSQL.
5. СУБД MongoDB, Redis.

Задание №3

Установка и настройка кластера Kubernetes

Есть различные готовые реализации кластера Kubernetes, например:

- Minikube — готовый кластер, который разворачивается на один компьютер. Хороший способ познакомиться с Kubernetes.
- Kubespray — набор Ansible ролей.
- Готовые кластеры в облаке, например, AWS, Google Cloud, Yandex Cloud и так далее.
- Использовать одну из готовых реализаций — быстрый и надежный способ развертывания системы оркестрации контейнеров Docker. Однако, мы рассмотрим ручное создание кластера Kubernetes из 3-х нод — один мастер (управление) и две рабочие ноды (запуск контейнеров).

Для этого выполним следующие шаги:

1. Подготовка системы

Данные действия выполняем на всех узлах будущего кластера. Это необходимо, чтобы удовлетворить программные системные требования для нашего кластера.

2. Настройка системы

1. Задаем имена узлам. Для этого выполняем команды на соответствующих серверах:

```
hostnamectl set-hostname k8s-master1.it-systems.local
```

```
hostnamectl set-hostname k8s-worker1.it-systems.local
```

```
hostnamectl set-hostname k8s-worker2.it-systems.local
```

** в данном примере мы зададим имя **k8s-master1** для мастера и, соответственно, **k8s-worker1** и **k8s-worker2** — для первого и второго рабочих нод. Каждая команда выполняется на своем сервере.*

Необходимо, чтобы наши серверы были доступны по заданным именам. Для этого необходимо на сервере DNS добавить соответствующие А-записи. Или на каждом сервере открываем hosts:

```
vi /etc/hosts
```

И добавляем записи на подобие:

```
192.168.0.15 k8s-master1.it-systems.local k8s-master1
192.168.0.20 k8s-worker1.it-systems.local k8s-worker1
192.168.0.25 k8s-worker2.it-systems.local k8s-worker2
```

** где, **192.168.0.15**, **192.168.0.20**, **192.168.0.25** — IP-адреса наших серверов, **k8s-master1**, **k8s-worker1**, **k8s-worker2** — имена серверов, **it-systems.local** — наш внутренний домен.*

2. Устанавливаем необходимые компоненты — дополнительные пакеты и утилиты. Для начала, обновим список пакетов и саму систему:

```
apt-get update && apt-get upgrade
```

Выполняем установку пакетов:

```
apt-get install curl apt-transport-https git iptables-persistent
```

** где:*

- **git** — утилита для работы с GIT. Понадобиться для загрузки файлов из репозитория git.
- **curl** — утилита для отправки GET, POST и других запросов на http-сервер. Понадобиться для загрузки ключа репозитория Kubernetes.
- **apt-transport-https** — позволяет получить доступ к АРТ-репозиториям по протоколу https.
- **iptables-persistent** — утилита для сохранения правил, созданных в iptables (не обязательна, но повышает удобство).

В процессе установки iptables-persistent может запросить подтверждение сохранить правила брандмауэра — отказываемся.

3. Отключаем файл подкачки. С ним Kubernetes не запустится.

Выполняем команду для разового отключения:

```
swapon -a
```

Чтобы swapon не появился после перезагрузки сервера, открываем на редактирование файл:

```
vi /etc/fstab
```

И комментируем строку:

```
#/swap.img none swap sw 0 0
```

Загружаем дополнительные модули ядра.

```
vi /etc/modules-load.d/k8s.conf
```

```
br_netfilter
```

```
overlay
```

** модуль **br_netfilter** расширяет возможности netfilter (подробнее); **overlay** необходим для Docker.*

Загрузим модули в ядро:

```
modprobe br_netfilter
```

```
modprobe overlay
```

Проверяем, что данные модули работают:

```
lsmod | egrep "br_netfilter|overlay"
```

Мы должны увидеть что-то на подобие:

```
overlay          114688  10
br_netfilter     28672   0
bridge          176128   1 br_netfilter
```

5. Изменим параметры ядра.

Создаем конфигурационный файл:

```
vi /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
```

* ***net.bridge.bridge-nf-call-iptables*** контролирует

возможность обработки трафика через bridge в netfilter. В нашем примере мы разрешаем данную обработку для IPv4 и IPv6.

Применяем параметры командой:

```
sysctl --system
```

3. Брандмауэр

Для мастер-ноды и рабочей создаем разные наборы правил.

По умолчанию, в Ubuntu брандмауэр настроен на разрешение любого трафика.

Если мы настраиваем наш кластер в тестовой среде, настройка брандмауэра не обязательна.

1. На мастер-ноде (Control-plane)

Выполняем команду:

```
iptables -I INPUT 1 -p tcp --match multiport --dports 6443,2379:2380,10250:10252
```

-j ACCEPT

* в данном примере мы открываем следующие порты:

- **6443** — подключение для управления (Kubernetes API).
- **2379:2380** — порты для взаимодействия мастера с воркерами (etcd server client API).
- **10250:10252** — работа с kubelet (соответственно

API, scheduler, controller-manager).

Для сохранения правил выполняем команду:

```
netfilter-persistent save
```

2. На рабочей ноде (Worker):

На нодах для контейнеров открываем такие порты:

```
iptables -I INPUT 1 -p tcp --match multiport --dports 10250,30000:32767 -j
```

ACCEPT

* где:

- **10250** — подключение к kubelet API.

- **30000:32767** — рабочие порты по умолчанию для подключения к подам (NodePort Services).

Сохраняем правила командой:

```
netfilter-persistent save
```

4. Установка и настройка Docker

На все узлы кластера выполняем установку Docker следующей командой:

```
apt-get install docker docker.io
```

После установки разрешаем автозапуск сервиса docker:

```
systemctl enable docker
```

Создаем файл:

```
vi /etc/docker/daemon.json
```

```
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2",
  "storage-opts": [
    "overlay2.override_kernel_check=true"
  ]
}
```

** для нас является важной настройкой **cgroupdriver** — она должна быть выставлена в значение **systemd**. В противном случае, при создании кластера Kubernetes выдаст предупреждение. Хоть на возможность работы последнего это не влияет, но мы постараемся выполнить развертывание без ошибок и предупреждений со стороны системы.*

И перезапускаем docker:

```
systemctl restart docker
```

5. Установка Kubernetes

Установку необходимых компонентов выполним из репозитория. Добавим его ключ для цифровой подписи:

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
```

Создадим файл с настройкой репозитория:

```
vi /etc/apt/sources.list.d/kubernetes.list
```

```
deb https://apt.kubernetes.io/ kubernetes-xenial main
```

Обновим список пакетов:

```
apt-get update
```

Устанавливаем пакеты:

```
apt-get install kubelet kubeadm kubectl
```

* где:

- **kubelet** — сервис, который запускается и работает на каждом узле кластера. Следит за работоспособностью подов.
- **kubeadm** — утилита для управления кластером Kubernetes.
- **kubectrl** — утилита для отправки команд кластеру Kubernetes.

Нормальная работа кластера сильно зависит от версии установленных пакетов. Поэтому бесконтрольное их обновление может привести к потере работоспособности всей системы. Чтобы этого не произошло, запрещаем обновление установленных компонентов:

```
apt-mark hold kubelet kubeadm kubectrl
```

Установка завершена — можно запустить команду:

```
kubectrl version --client
```

... и увидеть установленную версию программы:

```
Client Version: version.Info{Major:"1", Minor:"20", GitVersion:"v1.20.2",
GitCommit:"faecb196815e248d3ecfb03c680a4507229c2a56", GitTreeState:"clean",
BuildDate:"2021-01-13T13:28:09Z", GoVersion:"go1.15.5", Compiler:"gc",
Platform:"linux/amd64"}
```

Наши серверы готовы к созданию кластера.

6. Создание кластера

По-отдельности, рассмотрим процесс настройки мастер ноды (control-plane) и присоединения к ней двух рабочих нод (worker).

7. Настройка control-plane (мастер ноды)

Выполняем команду на мастер ноде:

```
kubeadm init --pod-network-cidr=10.244.0.0/16
```

** данная команда выполнит начальную настройку и подготовку основного узла кластера. Ключ — **pod-network-cidr** задает адрес внутренней подсети для нашего кластера.*

Выполнение займет несколько минут, после чего мы увидим что-то на подобие:

...

Then you can join any number of worker nodes by running the following on each as root:

```
kubeadm join 192.168.0.15:6443 --token f7sihu.wmgzwxkvbr8500al \
--discovery-token-ca-cert-hash
sha256:6746f66b2197ef496192c9e240b31275747734cf74057e04409c33b1ad280321
```

** данную команду нужно вводить на worker нодах, чтобы присоединить их к нашему кластеру. Можно ее скопировать, но позже мы будем генерировать данную команду по новой.*

В окружении пользователя создаем переменную KUBECONFIG, с помощью которой будет указан путь до файла конфигурации kubernetes:

```
export KUBECONFIG=/etc/kubernetes/admin.conf
```

Чтобы каждый раз при входе в систему не приходилось повторять данную команду, открываем файл:

```
vi /etc/environment
```

И добавляем в него строку:

```
export KUBECONFIG=/etc/kubernetes/admin.conf
```

Посмотреть список узлов кластера можно командой:

```
kubectl get nodes
```

На данном этапе мы должны увидеть только мастер ноду:

NAME	STATUS	ROLES	AGE	VERSION
k8s-master.it-systems.local	NotReady	<none>	10m	v1.20.2

Чтобы завершить настройку, необходимо установить CNI (Container Networking Interface) — в моем примере это flannel:

```
kubectl apply -f
```

<https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml>

** краткий обзор и сравнение производительности CNI можно почитать в [статье на хабре](#).*

Узел управления кластером готов к работе.

8. Настройка worker (рабочей ноды)

Мы можем использовать команду для присоединения рабочего узла, которую мы получили после инициализации мастер ноды или вводим (на первом узле):

```
kubeadm token create --print-join-command
```

Данная команда покажет нам запрос на присоединения новой ноды к кластеру, например:

```
kubeadm join 192.168.0.15:6443 --token f7sihu.wmgzwxkvbr8500al \
--discovery-token-ca-cert-hash
sha256:6746f66b2197ef496192c9e240b31275747734cf74057e04409c33b1ad280321
```

Копируем его и используем на двух наших узлах. После завершения работы команды, мы должны увидеть:

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

На мастер ноде вводим:

```
kubectl get nodes
```

Мы должны увидеть:

NAME	STATUS	ROLES	AGE	VERSION
k8s-master1.dmosk.local	Ready	control-plane,master	18m	v1.20.2
k8s-worker1.dmosk.local	Ready	<none>	79s	v1.20.2
k8s-worker2.dmosk.local	Ready	<none>	77s	v1.20.2

Наш кластер готов к работе. Теперь можно создавать поды, развертывания и службы. Рассмотрим эти процессы подробнее.

9. Pods

Поды — неделимая сущность объекта в Kubernetes. Каждый Pod может включать в себя несколько контейнеров (минимум, 1). Рассмотрим несколько примеров, как работать с подами. Все команды выполняем на мастере.

10. Создание

Поды создаются командой kubectl, например:

```
kubectl run nginx --image=nginx:latest --port=80
```

** в данном примере мы создаем под с названием **nginx**, который в качестве образа *Docker* будет использовать **nginx** (последнюю версию); также наш под будет слушать запросы на порту **80**.*

Чтобы получить сетевой доступ к созданному поду, создаем port-forward следующей командой:

```
kubectl port-forward nginx --address 0.0.0.0 8888:80
```

** в данном примере запросы к кластеру *kubernetes* на порт **8888** будут вести на порт **80** (который мы использовали для нашего пода).*

Команда **kubectl port-forward** является интерактивной. Ее мы используем только для тестирования. Чтобы пробросить нужные порты в Kubernetes используются Services — об этом будет сказано ниже.

Можно открыть браузер и ввести адрес `http://<IP-адрес мастера>:8888` — должна открыться страница приветствия для NGINX.

11. Просмотр

Получить список всех подов в кластере можно командой:

```
kubectl get pods
```

Например, в нашем примере мы должны увидеть что-то на подобие:

```
NAME READY STATUS RESTARTS AGE
nginx 1/1 Running 0 3m26s
```

Посмотреть подробную информацию о конкретном поде можно командой:

```
kubectl describe pods nginx
```

12. Запуск команд внутри контейнера

Мы можем запустить одну команду в контейнере, например, такой командой:

```
kubectl exec nginx -- date
```

** в данном примере будет запущена команда **date** внутри контейнера **nginx**.*

Также мы можем подключиться к командной строке контейнера командой:

```
kubectl exec --tty --stdin nginx -- /bin/bash
```

13. Удаление

Для удаления пода вводим:

```
kubectl delete pods nginx
```

14. Использование манифестов

В продуктивной среде управление подами выполняется с помощью специальных файлов с описанием того, как должен создаваться и настраиваться под — манифестов. Рассмотрим пример создания и применения такого манифеста.

Создадим файл формата `yml`:

```
vi manifest_pod.yml
```

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: web-srv
```

```
  labels:
```

```
    app: web_server
```

```
    owner: dmosk
```

```
  description: web_server_for_site
```


spec:

containers:

- name: nginx

- image: nginx:latest

- ports:

- containerPort: 80

- containerPort: 443

- name: php-fpm

- image: php-fpm:latest

- ports:

- containerPort: 9000

- name: mariadb

- image: mariadb:latest

- ports:

- containerPort: 3306

** в данном примере будет создан под с названием **web-srv**; в данном поде будет развернуто 3 контейнера — **nginx**, **php-fpm** и **mariadb** на основе одноименных образов.*

Для объектов Kubernetes очень важное значение имеют метки или labels.

Необходимо всегда их описывать. Далее, данные метки могут использоваться для настройки сервисов и развертываний.

Чтобы применить манифест выполняем команду:

```
kubectl apply -f manifest_pod.yaml
```

Мы должны увидеть ответ:

```
pod/web-srv created
```

Смотрим поды командой:

```
kubectl get pods
```

Мы должны увидеть:

NAME	READY	STATUS	RESTARTS	AGE
------	-------	--------	----------	-----

web-srv	3/3	Ready	0	3m11s
---------	-----	-------	---	-------

** для **Ready** мы можем увидеть 0/3 или 1/3 — это значит, что контейнеры внутри пода еще создаются и нужно подождать.*

15. Deployments

Развертывания позволяют управлять экземплярами подов. С их помощью контролируется их восстановление, а также балансировка нагрузки. Рассмотрим пример использования Deployments в Kubernetes.

16. Создание

Deployment создаем командой со следующим синтаксисом:

```
kubectl create deploy <название для развертывания> --image <образ, который должен использоваться>
```

Например:

```
kubectl create deploy web-set --image nginx:latest
```

** данной командой мы создадим deployment с именем **web-set**; в качестве образа будем использовать **nginx:latest**.*

17. Просмотр

Посмотреть список развертываний можно командой:

```
kubectl get deploy
```

Подробное описание для конкретного развертывания мы можем посмотреть так:

```
kubectl describe deploy web-set
```

** в данном примере мы посмотрим описание **deployment** с названием **web-set**.*

18. Scaling

Как было написано выше, deployment может балансировать нагрузкой. Это контролируется параметром scaling:

```
kubectl scale deploy web-set --replicas 3
```

** в данном примере мы указываем для нашего созданного ранее deployment использовать 3 реплики — то есть Kubernetes создаст 3 экземпляра контейнеров.*

Также мы можем настроить автоматическую балансировку:

```
kubectl autoscale deploy web-set --min=5 --max=10 --cpu-percent=75
```

В данном примере Kubernetes будет создавать от 5 до 10 экземпляров контейнеров — добавление нового экземпляра будет происходить при превышении нагрузки на процессор до 75% и более.

Посмотреть созданные параметры балансировки можно командой:

```
kubectl get hpa
```

19. Редактирование

Для нашего развертывания мы можем изменить используемый образ, например:

```
kubectl set image deploy/web-set nginx=httpd:latest --record
```

** данной командой для deployment web-set мы заменим образ nginx на httpd; ключ **record** позволит нам записать действие в историю изменений.*

Если мы использовали ключ **record**, то историю изменений можно посмотреть командой:

```
kubectl rollout history deploy/web-set
```

Перезапустить deployment можно командой:

```
kubectl rollout restart deploy web-set
```

20. Манифест

Как в случае с подами, для создания развертываний мы можем использовать манифесты. Попробуем рассмотреть конкретный пример.

Создаем новый файл:

```
vi manifest_deploy.yaml
```

```
apiVersion: apps/v1
```

```
kind: Deployment
```

```
metadata:
```

```
  name: web-deploy
```

```
  labels:
```

```
    app: web_server
```

```
    owner: it-systems
```

```
    description: web_server_for_site
```

```
spec:
```

```
  replicas: 5
```

```
selector:
  matchLabels:
    project: myweb
template:
  metadata:
    labels:
      project: myweb
      owner: it-systems
      description: web_server_pod
  spec:
    containers:
      - name: myweb-httpd
        image: httpd:latest
        ports:
          - containerPort: 80
          - containerPort: 443
```

```
---
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: web-deploy-autoscaling
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: myweb-autoscaling
  minReplicas: 5
  maxReplicas: 10
  metrics:
    - type: Resource
      resource:
        name: cpu
        target:
          type: Utilization
          averageUtilization: 75
    - type: Resource
      resource:
        name: memory
        target:
          type: Utilization
          averageUtilization: 80
```

** в данном манифесте мы создадим deployment и autoscaling. Итого, мы получим 5 экземпляров подов для развертывания web-deploy, которые могут быть расширены до 10 экземпляров. Добавление нового будет происходить при*

превышении нагрузки на процессор более чем на 75% или потреблением оперативной памяти более чем на 80%.

Чтобы создать объекты с помощью нашего манифеста вводим:

```
kubectl apply -f manifest_deploy.yaml
```

Мы должны увидеть:

```
deployment.apps/web-deploy created
```

```
horizontalpodautoscaler.autoscaling/web-deploy-autoscaling created
```

Объекты web-deploy и web-deploy-autoscaling созданы.

21. Удаление

Для удаления конкретного развертывания используем команду:

```
kubectl delete services web-deploy
```

** в данном примере будет удалена служба для развертывания **web-deploy**.*

Удалить все службы можно командой:

```
kubectl delete services --all
```

22. Манифест

Как в случае с подами и развертываниями, мы можем использовать манифест-файлы. Рассмотрим небольшой пример.

```
vi manifest_service.yaml
```

```
apiVersion: v1
```

```
kind: Service
```

```
metadata:
```

```
  name: web-service
```

```
  labels:
```

```
    app: web_server
```

```
    owner: it-systems
```

```
    description: web_server_for_site
```

```
spec:
```

```
  selector:
```

```
    project: myweb
```

```
  type: NodePort
```

```
  ports:
```

```
    - name: app-http
```

```
      protocol: TCP
```

```
      port: 80
```

```
      targetPort: 80
```

```
    - name: app-smtp
```

```
      protocol: TCP
```

```
      port: 25
```

```
      targetPort: 25
```

** в данном примере мы создадим службу, которая будем связываться с развертыванием по лейбл **project: myweb**.*

23. Ingress Controller

В данной инструкции не будет рассказано о работе с Ingress Controller. Оставляем данный пункт для самостоятельного изучения.

Данное приложение позволяет создать балансировщик, распределяющий сетевые запросы между нашими сервисами. Порядок обработки сетевого трафика определяем с помощью Ingress Rules.

Существует не маленькое количество реализаций Ingress Controller — их сравнение можно найти в документе по ссылке в [Google Docs](#).

Для установки Ingress Controller Contour (среди множества контроллеров, он легко устанавливается и на момент обновления данной инструкции полностью поддерживает последнюю версию кластера Kubernetes) вводим:

```
kubectl apply -f https://projectcontour.io/quickstart/contour.yaml
```

24. Установка веб-интерфейса

Веб-интерфейс позволяет получить информацию о работе кластера в удобном для просмотра виде.

В большинстве инструкций рассказано, как получить доступ к веб-интерфейсу с того же компьютера, на котором находится кластер (по адресу 127.0.0.1 или localhost). Но мы рассмотрим настройку для удаленного подключения, так как это более актуально для серверной инфраструктуры.

Переходим на страницу веб-интерфейса в [GitHub](#) и копируем ссылку на последнюю версию файла yaml:

Installation

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.1.0/aio/deploy/recommended.yaml
```

** на момент обновления инструкции, последняя версия интерфейса была 2.1.0.*

Скачиваем yaml-файл командой:

```
wget
```

```
https://raw.githubusercontent.com/kubernetes/dashboard/v2.1.0/aio/deploy/recommended.yaml
```

*

где <https://raw.githubusercontent.com/kubernetes/dashboard/v2.1.0/aio/deploy/recommended.yaml> — ссылка, которую мы скопировали на портале GitHub.

Открываем на редактирование скачанный файл:

```
vi recommended.yaml
```

Комментируем строки для **kind: Namespace** и **kind: Secret** (в файле несколько блоков с kind: Secret — нам нужен тот, что с **name: kubernetes-dashboard-certs**):

```
...
```

```
#apiVersion: v1
```

```
#kind: Namespace
```

```
#metadata:
```

```
# name: kubernetes-dashboard
```

```
...
```

```
#apiVersion: v1
```

```
#kind: Secret
```

```
#metadata:
```

```
# labels:
```

```
# k8s-app: kubernetes-dashboard
# name: kubernetes-dashboard-certs
# namespace: kubernetes-dashboard
#type: Opaque
```

** нам необходимо закомментировать эти блоки, так как данные настройки в Kubernetes мы должны будем сделать вручную.*

Теперь в том же файле находим **kind: Service** (который с **name: kubernetes-dashboard**) и добавляем строки **type: NodePort** и **nodePort: 30001** (выделены жирным):

```
kind: Service
apiVersion: v1
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
```

spec:

```
type: NodePort
```

```
ports:
```

```
- port: 443
  targetPort: 8443
nodePort: 30001
```

```
selector:
```

```
k8s-app: kubernetes-dashboard
```

** таким образом, мы публикуем наш сервис на внешнем адресе и порту 30001.*

Для подключения к веб-интерфейсу не через локальный адрес, начиная с версии 1.17, обязательно необходимо использовать зашифрованное подключение (https). Для этого нужен сертификат. В данной инструкции мы сгенерируем самоподписанный сертификат — данный подход удобен для тестовой среды, но в продуктивной среде необходимо купить сертификат или получить его бесплатно в Let's Encrypt.

И так, создаем каталог, куда разместим наши сертификаты:

```
mkdir -p /etc/ssl/kubernetes
```

Сгенерируем сертификаты командой:

```
openssl req -new -x509 -days 1461 -nodes -out /etc/ssl/kubernetes/cert.pem -keyout
/etc/ssl/kubernetes/cert.key -subj "/C=RU/ST=SPb/L=SPb/O=Global Security/OU=IT
Department/CN=kubernetes.it-systems.local/CN=kubernetes"
```

** можно не менять параметры команды, а так их и оставить. Браузер все равно будет выдавать предупреждение о неправильном сертификате, так как он самоподписанный.*

Создаем namespace:

```
kubectl create namespace kubernetes-dashboard
```

** это та первая настройка, которую мы комментировали в скачанном файле **recommended.yaml**.*

Теперь создаем настройку для secret с использованием наших сертификатов:

```
kubectl create secret generic kubernetes-dashboard-certs --from-  
file=/etc/ssl/kubernetes/cert.key --from-file=/etc/ssl/kubernetes/cert.pem -n kubernetes-  
dashboard
```

** собственно, мы не использовали настройку в скачанном файле, так как создаем ее с включением в параметры пути до созданных нами сертификатов.*

Теперь создаем остальные настройки с помощью скачанного файла:

```
kubectl create -f recommended.yaml
```

Мы увидим что-то на подобие:

```
serviceaccount/kubernetes-dashboard created
```

```
service/kubernetes-dashboard created
```

```
secret/kubernetes-dashboard-csrf created
```

```
secret/kubernetes-dashboard-key-holder created
```

```
configmap/kubernetes-dashboard-settings created
```

```
role.rbac.authorization.k8s.io/kubernetes-dashboard created
```

```
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
```

```
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
```

```
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
```

```
deployment.apps/kubernetes-dashboard created
```

```
service/dashboard-metrics-scraper created
```

```
deployment.apps/dashboard-metrics-scraper created
```

Создадим настройку для админского подключения:

```
vi dashboard-admin.yaml
```

```
apiVersion: v1
```

```
kind: ServiceAccount
```

```
metadata:
```

```
  labels:
```

```
    k8s-app: kubernetes-dashboard
```

```
  name: dashboard-admin
```

```
  namespace: kubernetes-dashboard
```

```
---
```

```
apiVersion: rbac.authorization.k8s.io/v1
```

```
kind: ClusterRoleBinding
```

```
metadata:
```

```
  name: dashboard-admin-bind-cluster-role
```

```
  labels:
```

```
    k8s-app: kubernetes-dashboard
```

```
roleRef:
```

```
  apiGroup: rbac.authorization.k8s.io
```

```
  kind: ClusterRole
```

```
  name: cluster-admin
```

```
subjects:
```

```
- kind: ServiceAccount
```

```
  name: dashboard-admin
```

```
  namespace: kubernetes-dashboard
```

Создаем настройку с применением созданного файла:

```
kubectl create -f dashboard-admin.yaml
```

Теперь открываем браузер и переходим по ссылке `https://<IP-адрес мастера>:30001` — браузер покажет ошибку сертификата (если мы настраиваем по инструкции и сгенерировали самоподписанный сертификат). Игнорируем ошибку и продолжаем загрузку.

Kubernetes Dashboard потребует пройти проверку подлинности. Для этого можно использовать токен или конфигурационный файл:

Kubernetes Dashboard

- ☒ Token

Every Service Account has a Secret with valid Bearer Token that can be used. Bearer Tokens, please refer to the [Authentication](#) section.
- ☐ Kubeconfig

Please select the kubeconfig file that you have created to configure access to the cluster. For more information, please refer to the [Configure Access to Multiple Clusters](#) section.

На сервере вводим команду для создания сервисной учетной записи:

```
kubectl create serviceaccount dashboard-admin -n kube-system
```

Создадим привязку нашего сервисного аккаунта с Kubernetes Dashboard:

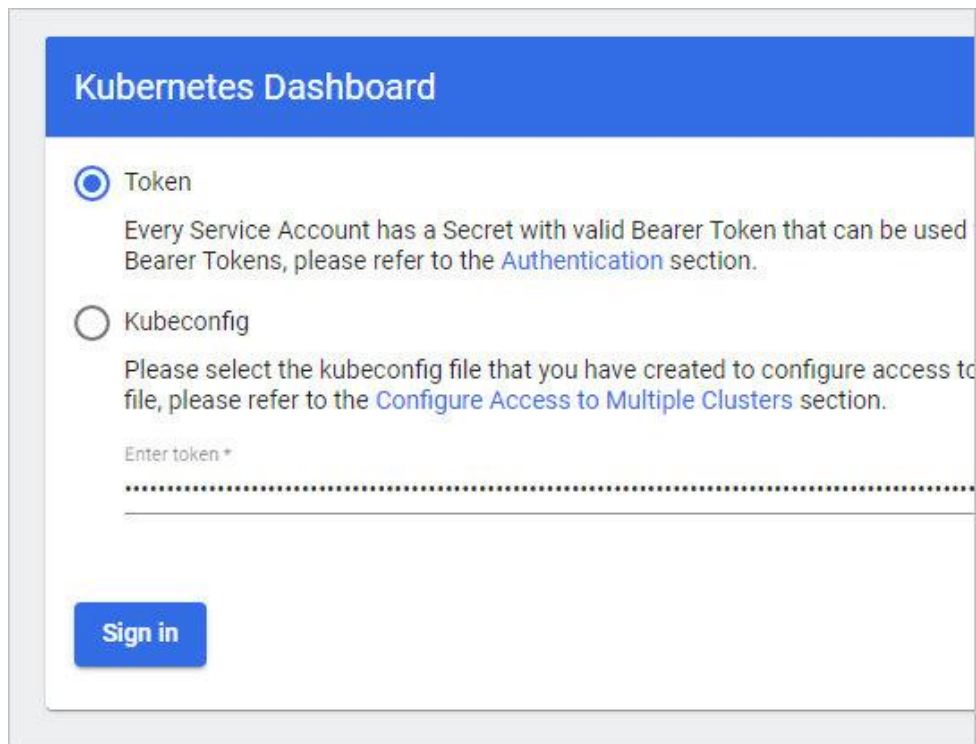
```
kubectl create clusterrolebinding dashboard-admin --clusterrole=cluster-admin --serviceaccount=kube-system:dashboard-admin
```

Теперь командой:

```
kubectl describe secrets -n kube-system $(kubectl -n kube-system get secret | awk 'NR==1{print $1}' | sed 's/^secret-//')
'/dashboard-admin/{ print $1 }'
```

получаем токен для подключения

Используя полученный токен, вводим его в панели авторизации:



Kubernetes Dashboard

☒ Token

Every Service Account has a Secret with valid Bearer Token that can be used Bearer Tokens, please refer to the [Authentication](#) section.

☐ Kubeconfig

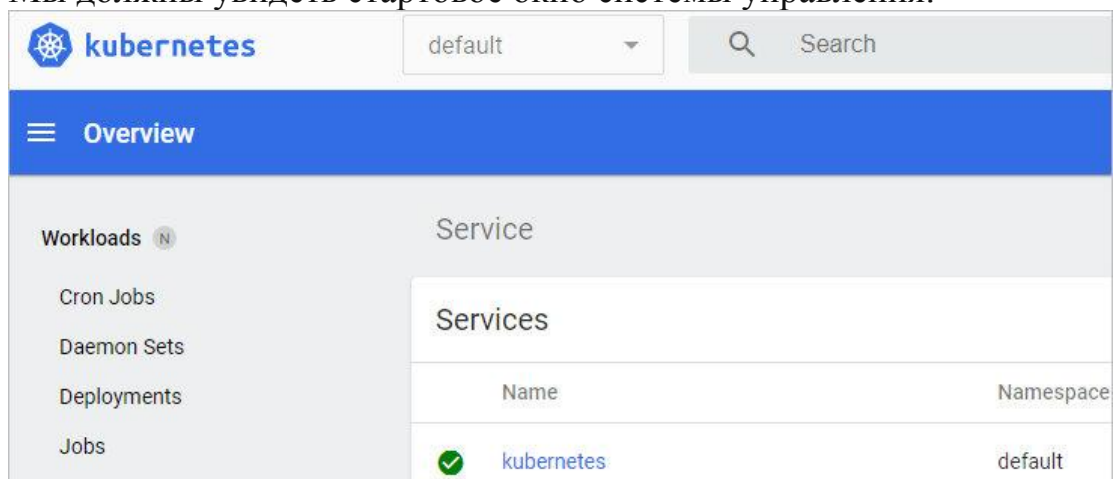
Please select the kubeconfig file that you have created to configure access to file, please refer to the [Configure Access to Multiple Clusters](#) section.

Enter token *

.....

Sign in

Мы должны увидеть стартовое окно системы управления:



kubernetes default Search

Overview

Workloads N

- Cron Jobs
- Daemon Sets
- Deployments
- Jobs

Service

Services

Name	Namespace
kubernetes	default

25. Удаление нод

При необходимости удалить ноду из нашего кластера, вводим 2 команды:

```
kubectl drain k8s-worker2.it-systems.local --ignore-daemonsets
```

```
kubectl delete node k8s-worker2.it-systems.local
```

** в данном примере мы удаляем ноду **k8s-worker2.it-systems.local**.*

Задание №4

Вопросы

1. Почему Kubernetes часто обозначают K8s?
2. Чем K8s отличается от Docker-Compose?
3. Что используется для описания состояния объектов в K8s?
4. Как связаны понятия Deployment, Replica Sets, Pods?
5. Для чего используются Namespace?
6. Что общего и в чем различие между Service и Ingress?
7. Что используется для подключения Pod к общим файлам?
8. Для чего нужен Helm и что используется для его описания?

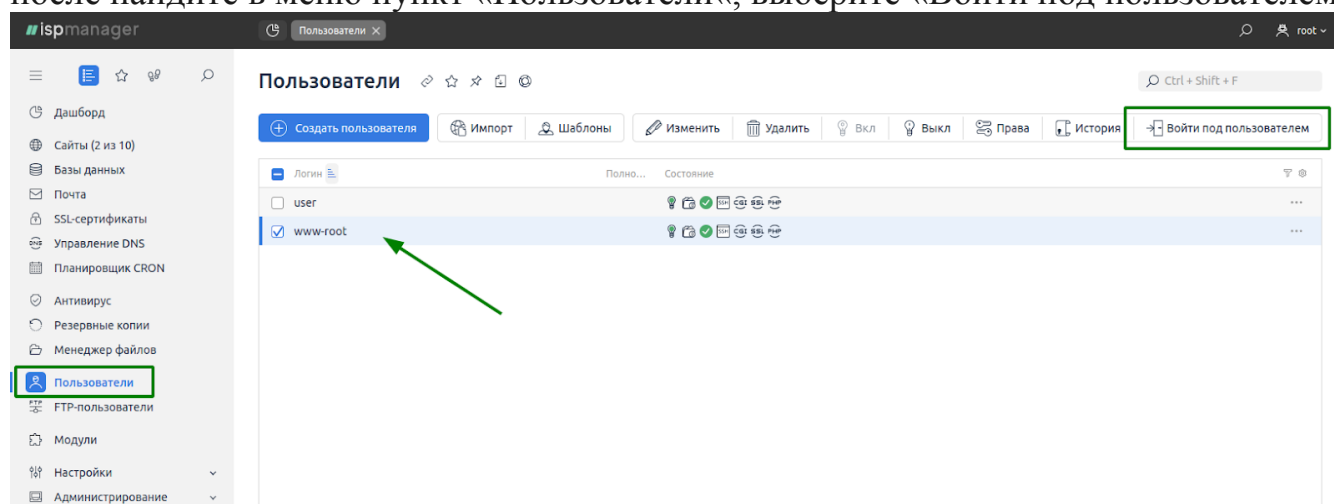
Задание №5

Установка SSL-сертификата на домен

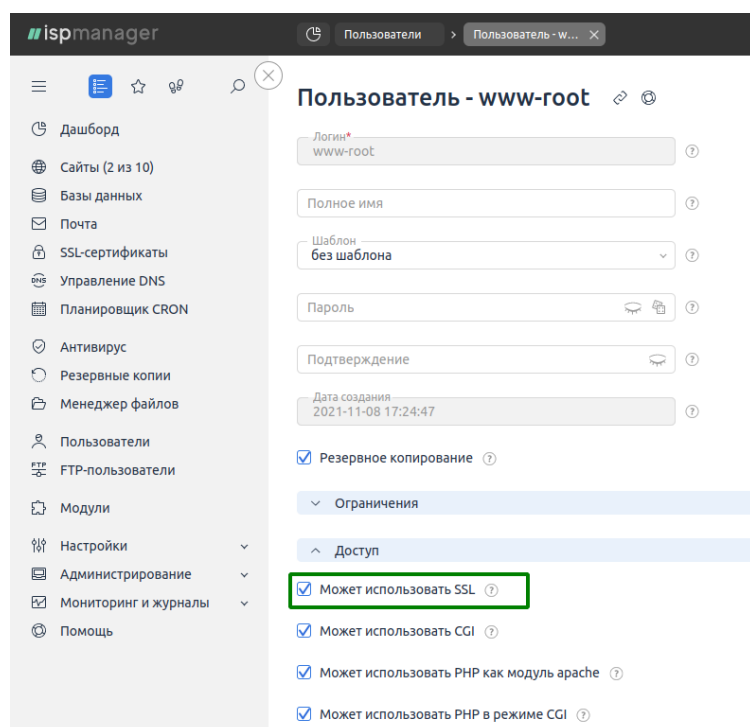
После получения необходимых файлов и сертификата, необходимо встроить этот сертификат на сервер, чтобы потом им подписать ваше доменное имя. Данный материал описывает установку сертификата с помощью менеджера ISPmanager, а также установку напрямую, не используя утилиты.

Использование ISPmanager 6

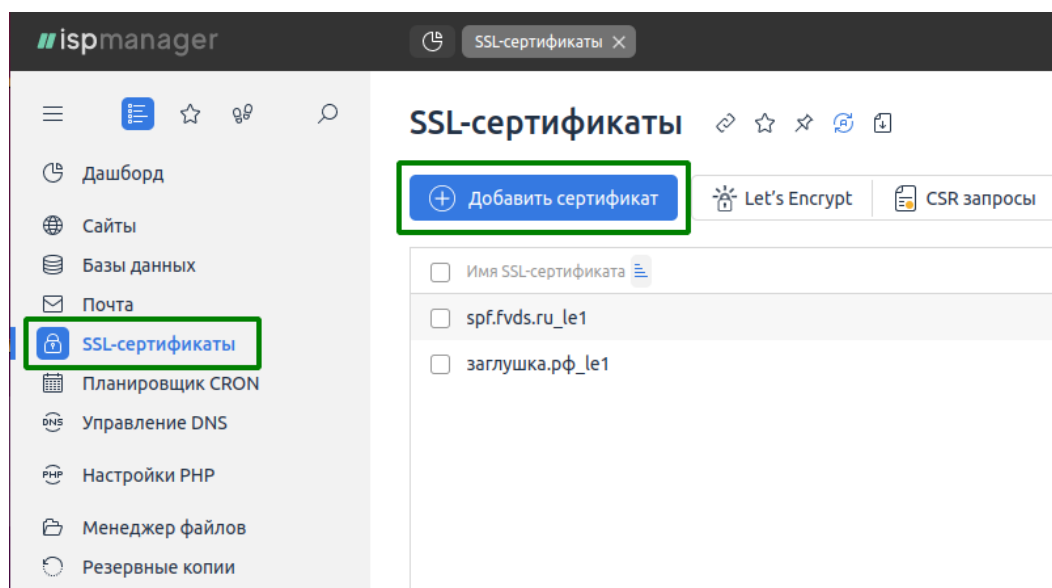
Авторизируйтесь в качестве пользователя, который владеет необходимым доменом. Чтобы это сделать, войдите в систему с правами суперпользователя (root), после найдите в меню пункт «Пользователи», выберите «Войти под пользователем».



Важно: вам необходимо подключить возможность использования SSL. В пункте «Пользователи», в настройках определенного пользователя найдите «Может использовать SSL», отметьте этот пункт.



В подменю «SSL-сертификаты» — «Добавить сертификат».



Тип сертификата — существующий.

Параметры:

Имя сертификата — имя формируется в зависимости от информации, которая содержится в сертификате + «_customX». Если у вас версия панели < 6.19, то вам необходимо вручную установить имя сертификата. SSL-Сертификат — поле, для загрузки файла .crt. Ключ сертификата — содержимое приватного ключа (файл .key). Также вам необходимо представить в панель файл (.ca/.ctrca). Это цепочка сертификатов (сертификат, которым подписан ваш исходный). Содержимое сертификатов просто скопируйте из файлов и вставьте в подходящие поля.

Например: цепочка сертификатов от GlobalSign

Вы получаете архив с файлами (файл .p7b сертификат и цепочка сертификата, файл с именем домена + открытый ключ + сертификат, корневой ключ цепочки сертификата, промежуточный ключ .crt).

Чтобы создать цепочки сертификата поместите в один файл .txt корневой ключ сертификата, после за ним, начиная с новой строки, промежуточный ключ и установите для него название [GlobalSign.ca](https://www.globalsign.com/). Откройте файлы с ключами цепочки через любой текстовый редактор. Скопируйте содержимое, обязательно проверьте целостность скопированного кода.

Когда вы добавили сертификат, переходите на страницу «Сайты», запустите сервер. Вы увидите установленный сертификат. Но если настройки сайта будут меняться, то файлы конфигурации будут перезаписаны автоматически.

Изменения в файлах, которые вы вносили руками, могут не сохраниться. В случае возникновения подобной ситуации вам нужно перейти в /var/www/httpd-

cert/ИМЯ_ПОЛЬЗОВАТЕЛЯ/ИМЯ_СЕРТИФИКАТА и повторно внести нужные изменения.

Проверка сертификата

Используйте ресурс: <https://www.ssllabs.com/ssltest/analyze.html>

You are here: [Home](#) > [Projects](#) > [SSL Server Test](#) > firstvds.ru

SSL Report: firstvds.ru (37.230.118.233)

Assessed on: Mon, 07 Nov 2022 08:55:26 UTC | [Hide](#) | [Clear cache](#) [Scan Another »](#)

Summary

Overall Rating

A

Certificate

Protocol Support

Key Exchange

Cipher Strength

Visit our [documentation page](#) for more information, configuration guides, and books. Known issues are documented [here](#).

Server sent invalid HSTS policy. See below for further information.

This site works only in browsers with SNI support.

This server supports TLS 1.3.

Certificate #1: RSA 2048 bits (SHA256withRSA)

Server Key and Certificate #1

Subject	firstvds.ru Fingerprint SHA256: 557d0160d95514aa57c16742c26db2cc9bb57a6883b6892187ca0c7809a5b51 Pin SHA256: YCFXcOKK8rTP1hQY70A/X5Tz2ewWJhA7xM7H0d/k=
Common names	firstvds.ru
Alternative names	firstvds.ru
Serial Number	12c332dd936b5ab967c2bdcf
Valid from	Wed, 19 Oct 2022 03:10:19 UTC
Valid until	Mon, 20 Nov 2023 03:10:18 UTC (expires in 1 year)
Key	RSA 2048 bits (e 65537)
Weak key (Debian)	No

Для проверки валидности работы установленного сертификата: <https://www.sslshopper.com/ssl-checker.html>

SSL Checker

Use our fast SSL Checker to help you quickly diagnose problems with your SSL certificate installation. You can verify the SSL certificate on your web server to make sure it is correctly installed, valid, trusted and doesn't give any errors to any of your users. To use the SSL Checker, simply enter your server's public hostname (internal hostnames aren't supported) in the box below and click the Check SSL button. If you need an SSL certificate, check out the SSL Wizard.

[More Information About the SSL Checker](#)

Server Hostname

https://firstvds.ru/

Check SSL

✔

firstvds.ru resolves to 37.230.118.233

✔

Server Type: ddos-guard

✔

The certificate should be trusted by all major web browsers (all the correct intermediate certificates are installed).

✔

The certificate was issued by GlobalSign. [Write review of GlobalSign](#)

✔

The certificate will expire in 377 days. [Remind me](#)

✔

The hostname (firstvds.ru) is correctly listed in the certificate.

Server

Common name: firstvds.ru

SAKs: firstvds.ru

Valid from: October 18, 2022 to November 19, 2023

Serial Number: 12c332dd936b5ab967c2bdcf

Signature Algorithm: sha256WithRSAEncryption

Issuer: AlphaSSL CA - SHA256 - G2

Chain

Common name: AlphaSSL CA - SHA256 - G2

Organization: GlobalSign nv-sa

Location: BE

Valid from: February 20, 2014 to February 20, 2024

Serial Number: 040000000101444ef36311

Signature Algorithm: sha256WithRSAEncryption

Issuer: GlobalSign Root CA

Установка SSL-сертификата без использования внешних панелей

В первую очередь, подключитесь к серверу по SSH.

Проверьте порт 443 — (netstat -napt | grep 443 или ss -tlpn | grep 443 или sockstat | grep 443 для FreeBSD).

Образец вывода: root httpd 83299 19 tcp4 188.120.225.20:443 *:*

Установка с использованием файла конфигурации веб-сервера

Apache

Путь для установки сертификата:

UBUNTU/DEBIAN — /etc/apache2/apache2.conf

Centos — /etc/httpd/conf/httpd.conf

FreeBSD — /usr/local/etc/apache22/httpd.conf

Создайте <VirtualHost>, чтобы осуществить SSL-соединение:

<VirtualHost 10.0.0.1:443>

DocumentRoot /var/www/user/data/www/domain.com

ServerName domain.com

SSLEngine on

SSLCertificateFile /path/to/domain.crt

SSLCertificateKeyFile /path/to/domain.key

SSLCACertificateFile /path/to/ca.crt

</VirtualHost>

Где:

- domain.com — доменное имя вашего домена
- 10.0.0.1 — IP адрес домена
- /var/www/user/data/www/domain.com — путь до домашней директории,

где расположен ваш домен

- /path/to/domain.crt — файл с сертификатом
- /path/to/domain.key — файл с ключом для сертификата
- /path/to/ca.crt — файл, содержащий корневой сертификат

После установки изменений, используйте команду:

apachectl -t

Если получено сообщение — Syntax OK, то можете перезапустить сервер — systemctl restart apache2 или systemctl restart httpd или apachectl restart

NGINX

Откройте файл конфигурации сервера. Путь: /etc/nginx/nginx.conf

Инициализируйте модуль SSL:

server {

listen 10.0.0.1:443;

server_name domain.com;

ssl on;

ssl_certificate /path/to/domain.crt;

ssl_certificate_key /path/to/domain.key ;

}

domain.com — доменное имя **10.0.0.1** — IP, который принадлежит домену
путь — /path/to/domain.crt (сертификат)/path/to/domain.key — ключ сервера
(путь)

Цепочка сертификата добавляется в файл с самим сертификатом.

После того, как вы изменили файл конфигурации, проверьте наличие ошибок в синтаксисе, а также ваши вставленные данные.

nginx -t — проверка конфигурации

nginx: the configuration file /etc/nginx/nginx.conf syntax is ok

nginx: configuration file /etc/nginx/nginx.conf test is successful

Если вы увидели подобный вывод, то SSL настроен корректно

systemctl restart nginx — перезапуск сервера

SSL-сертификаты (несколько) на едином IP

Браузер получит сертификат сервера, выставленный по умолчанию. Это не зависит от недостаточной настройки, так работает SSL. Чтобы разрешить работы нескольких сертификатов на одном IP, необходимо воспользоваться расширением SNI протокола TLS. Ссылка на документацию (<https://tools.ietf.org/html/rfc6066>). Данный модуль позволит установить SSL-handshake, во время которого сервер определит необходимый сертификат для использования. Большинство современных браузеров поддерживает это решение, но стоит обратить внимание также на версию библиотеки OpenSSL, она должна быть > 0.9.8f.

Полезные команды Openssl

- Создание ключа для SSL-сертификата:

openssl req -batch -noout -new -newkey rsa:2048 -nodes -keyout cert.key

- Сгенерировать CSR-запрос:

openssl req -new -key cert.key -out cert.csr

- Сделать ключ без пароля:

openssl rsa -in cert.key -out cert.key

- CSR данные:

openssl req -noout -text -in cert.csr

- Данные сертификата (проверить кем выдан, например):

openssl x509 -noout -text -in cert.crt

- Проверить, что ключ соответствует сертификату:

openssl x509 -noout -modulus -in cert.crt | openssl md5

openssl rsa -noout -modulus -in cert.key | openssl md5

Два последних значения должны совпадать, в нашем случае это md5.

- Узнать длину запроса:

echo ‘(‘ `openssl req -noout -modulus -in cert.csr | cut -d’=’ -f2 | wc -c` ‘-1)*4’ |

bc

- Проверить HTTPS:

openssl s_client -host ulanovka.ru -port 443

Задание №6

Администрирование FTP-серверов.

1. На виртуально машине разверните ftp-сервер;
2. Разрешите анонимный доступ для всех пользователей на данный ftp-сервер. Проверьте работу ftp-сервера с данной конфигурацией с вашей основной операционной системы;

3. Настройте разграничение прав доступа к определенным каталогам пользователей на ftp-сервере. Проверьте работу ftp-сервера с данной конфигурацией с вашей основной операционной системы;
4. Настройте смешанный режим доступа анонимных и зарегистрированных пользователей. Проверьте работу ftp-сервера с данной конфигурацией с вашей основной операционной системы;
5. Установить ssh-сервер на вашу операционную систему Linux. Настройте ssh с точки зрения безопасности;
6. На вашей основной операционной системе установить ssh-клиент (если основная операционная система Linux) или putty (если основная операционная система Windows). Проверьте работу ssh, настроив клиент соответствующим образом;
7. Установить веб сервер на Linux Ubuntu;
8. Создайте простую html-страничку. Разместите её на веб-сервере по веб-адресам: work.my и www.work.my.

Задание №7

Установка необходимых пакетов: apt-get install apache2-utils libpam-pwdfile

Настройка виртуальных пользователей vsftpd

Параметры для настройки должны находиться в файле /etc/vsftpd.conf.

listen=YES

listen_ipv6=NO

anonymous_enable=NO

local_enable=YES

write_enable=YES

local_umask=022

nopriv_user=vsftpd chroot_local_user=YES

allow_writeable_chroot=yes guest_username=vsftpd virtual_use_local_privs=YES

guest_enable=YES

user_sub_token=\$USER

local_root=/var/www/\$USER

hide_ids=YES

В файле /etc/pam.d/vsftpd должен храниться путь к файлу с пользователями:

auth required pam_pwdfile.so pwdfile /etc/ftpusers account required pam_permit.so

Создание виртуального пользователя test FTP-сервера: htpasswd -d /etc/ftpusers test

Создание домашнего каталога для созданного пользователя: mkdir

/var/www/test chown ftp:nogroup /var/www/test chmod +w /var/www/test

Сконфигурировать FTP-сервер для поддержки входа виртуальных пользователей.

Создать виртуального пользователя и домашний каталог для него. В качестве логина пользователя использовать свою фамилию.

С помощью FTP-клиента продемонстрировать работу виртуального пользователя

Задание №8

1. Каково назначение ftp-сервера?
2. Каким образом производится настройка vsftpd?
3. Каково назначение сетевого протокола SSH?
4. Какие основные параметры рекомендуется менять при настройке SSH с точки зрения его безопасности и почему?
5. Каково назначение Telnet? Почему Telnet не рекомендуется использовать?
6. Каково назначение Apache?
7. Какие основные конфигурационные файлы Apache существуют?

Задание №9

Установка и настройка системы защиты от DDoS-атак

Для эффективной защиты от распределённых сетевых атак типа «отказ в обслуживании» требуется своевременное обнаружение начала атаки. Чтобы обнаружить DDoS-атаку необходимо накапливать статистические данные о трафике сервера в сети, работающем в штатном режиме. При знании среднестатистических значений характеристик сервера, можно отследить появление аномалии трафика в сети.

Для выполнения данной работы необходимо создать две виртуальные машины или объединить в локальную сеть два компьютера. Одна машина будет жертвой, другая злоумышленником.

На машине жертве необходимо установить:

Web-сервер (Apache, NGINX);

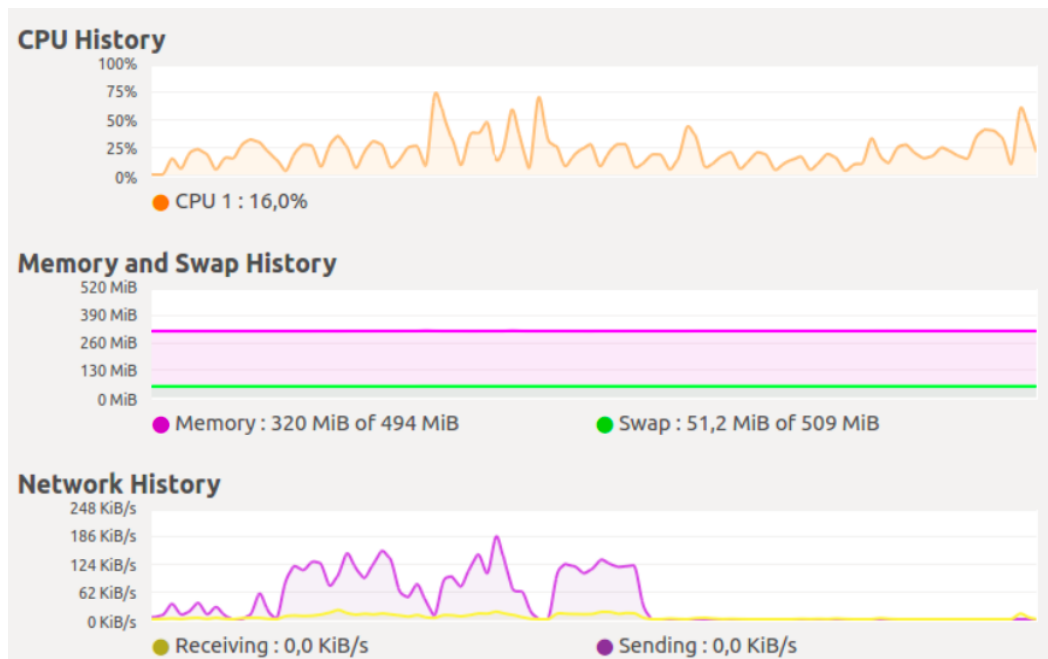
Средства мониторинга и анализа ресурсов сервера и сетевого трафика (iptraf, ksysguard или другие);

На машине злоумышленника необходимо установить:

Программное обеспечение и скрипты для проведения DDoS атак;

Уязвимые сервисы для усиления DDoS атак (dns, ntp, chargen);

На машине-жертве фиксируются системные характеристики, такие как: загрузки процессора, объем оперативной памяти и характеристики сетевого трафика, такие как: количество пакетов, открытые соединения.



Proto/Port	Pkts	Bytes	PktsTo	BytesTo	PktsFrom	BytesFrom
TCP/443	11160	8170815	4543	440587	6617	7730228
UDP/53	1268	164284	650	45740	618	118544
UDP/123	8	608	8	608	8	608
TCP/80	1941	1168863	989	76369	952	1092494
UDP/68	4	1808	2	1152	2	656
UDP/67	4	1808	2	656	2	1152

Скриншоты мониторинга и трафика

Изменение указанных характеристик не обязательно обозначает атаку на сервер, но показывает отклонение от среднестатистических показателей работы в штатном режиме. Для большинства DDoS-атак можно выделить аномалии и признаки, соответствующие только им.

В ходе данной лабораторной работы необходимо провести каждый из представленных типов DDoS атак, отследить аномалии в использовании ресурсов сервера и сетевом трафике, проверить признаки DDoS атаки и, если они подтвердились, защитить сервер от атаки с помощью настройки сервера, добавление правил для межсетевого экрана, фильтрации пакетов по сигнатурам.

Пошаговый алгоритм методики:

Обнаружение аномалии в наблюдаемых системных характеристиках и характеристиках сетевого трафика.

Проверить присутствие признаков атаки в трафике.

Внести изменения в настройки сервера под конкретную атаку и внести правила для фильтрации.

Для каждого типа атак в сети интернет необходимо найти по несколько программ и скриптов, которые реализуют данный вид атаки.

Для атаки HTTP flood: goldeneye, ddosim, DAVOSET, HULK, LOIC, BBH.

Для медленных атак: r-u-dead-yet, slowhttptest, pyloris, sockstress, torshammer.

Для UDP flood: LOIC, fudp, hping3.

Для SYN flood и атак на TCP-стек: sprut, sitekiller, mummy, hping3.

Для ICMP flood: hping3.

Для land атаки: fudp, hping3.

Для атак с использованием SSL: thc-ssl-dos.

Для amplification атак: saddam, chargen_amp.

Ознакомьтесь со справочной документацией используемых Вами программ и скриптов.

Дополнительная информация

Медленные атаки

Аномалия: Кратковременный скачок нагрузки процессора, повышение исходящего трафика и используемой оперативной памяти.

Признак: В TCP-стеке большое количество соединений со статусом ESTABLISHED.

Защита: Если установлен apache, то поставить перед ним кэширующий сервер NGINX. Установить и настроить балансировщик нагрузки. На apache можно установить mod_security — firewall с готовыми правилами от OWASP и mod_reqtimeout — установка таймаутов и минимальной скорости передачи данных для получения запросов.

Добавить правила межсетевого экрана:

ограничение количества соединений с одного IP-адреса;

```
# iptables -A INPUT -p tcp --syn --dport 80 -m connlimit --connlimit-above 30 -j DROP
```

заблокировать IP после 10 подключений к порту 80 в течение 30 секунд;

```
# iptables -I INPUT -p tcp --dport 80 -m state --state NEW -m recent --set
# iptables -I INPUT -p tcp --dport 80 -m state --state NEW -m recent --update --seconds 30 --hitcount 10 -j DROP
```

Также NGINX имеет функцию кэширования клиентского запроса перед отправкой на бэкэнд — client_body_buffer_size. Бэкэнд получит запрос, только когда он полностью загрузится.

Атака произвольными пакетами.

Аномалия: Резкое увеличение нагрузки процессора, входящего и исходящего-HTTP трафика.

Признак: На сервер приходит много однотипных пакетов, резко выросло количество обращений к ресурсоёмкому элементу сайта.

Защита: Борьба с HTTP flood ведётся с помощью усовершенствования работы веб-сервера и баз данных. Также меры включают в себя использование обработки подключений методом epoll, увеличение количества соединений, отключение тайм-аута на закрытие подключений keep-alive.

```
worker_processes 2; worker_rlimit_nofile 8000; events {
worker_connections 4000;
use epoll; }
```

Установить на NGINX модуль ngx_http_limit_req_module и заблокировать IP-адреса, которые стали получать ответ «Service unavailable».

```
http {
limit_req_zone $binary_remote_addr zone=zne:15m rate=3r/s; server {
location / {
```

```
limit_req zone=zne burst=5;  
}
```

Атака произвольными пакетами. UDP-Flood

Аномалия: Резкое увеличение нагрузки процессора, входящего UDP-трафика.

Признак: Со всех открытых UDP-портов идёт исходящий трафик.

Защита: Необходимо выставить ограничение на количество подключений к открытым портам и закрыть неиспользуемые порты с помощью межсетевых экранов.

Добавить правила межсетевого экрана:

ограничить количество подключений;

```
# iptables -I INPUT -p udp --dport 53 -j DROP -m iplimit --iplimit-above 1 •
```

разрешить подключение только доверенным IP-адресам;

```
# iptables -A OUTPUT -p udp --dport 53 -d 8.8.4.4 -j ACCEPT
```

блокировать все остальные порты.

```
# iptables -A OUTPUT -p udp -j DROP
```

Атака произвольными пакетами. SYN-Flood

Аномалия: Кратковременный скачок используемых ресурсов процессора и многократное увеличение входящего и исходящего трафика, и их количество равно.

Признак: В TCP-стеке появляется большое количество полуоткрытых соединений со статусом SYN_RECV.

Защита: Современные реализации TCP протокола и некоторые межсетевые экраны имеют механизм защиты от SYN flood атак.

Принцип его действия заключается в следующем:

Серверу отправляется запрос на установление соединения, механизм регистрирует его в таблице.

После ответа сервера о подтверждении запроса на соединение механизм отправляет пакет серверу с подтверждением соединения и отправляет ответный пакет клиенту. В этот момент в механизме запускается таймер, отсчитывающий время на ответ от клиента.

При получении пакета с подтверждением соединения механизм считает запрос валидным, остановит таймер и отправит его на сервер.

Если пакет не пришёл в установленное время, механизм отправляет серверу запрос на удаление информации о соединении.

Принцип действие механизма основан на контроле времени задержки ответа клиента. При слишком коротком тайм-ауте будут обрываться легитимные пользователи, при длинном будут накапливаться соединения, что может привести к серьёзным последствиям.

Также можно использовать SYN cookie, или ограничить количество запросов на новые подключения от конкретного пользователя за конкретный период времени. На Рисунке 1 представлен пример настройки ядра операционной системы для защиты от SYN-Flood с помощью команды sysctl.

```
test@test:~$ sudo sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
test@test:~$ sudo sysctl -w net.ipv4.tcp_max_syn_backlog=4096
net.ipv4.tcp_max_syn_backlog = 4096
test@test:~$ sudo sysctl -w net.ipv4.tcp_keepalive_intvl=10
net.ipv4.tcp_keepalive_intvl = 10
test@test:~$ sudo sysctl -w net.ipv4.tcp_keepalive_probes=5
net.ipv4.tcp_keepalive_probes = 5
test@test:~$ sudo sysctl -w net.ipv4.conf.default.rp_filter=1
net.ipv4.conf.default.rp_filter = 1
test@test:~$ sudo sysctl -w net.ipv4.tcp_synack_retries=1
net.ipv4.tcp_synack_retries = 1
test@test:~$ sudo sysctl -w net.ipv4.tcp_fin_timeout=10
net.ipv4.tcp_fin_timeout = 10
```

Атака произвольными пакетами. ICMP-Flood

Аномалия: Очень сильная загрузка процессора и сильное увеличение входящего и исходящего ICMP трафика, равного по значениям.

**** Признак:**** Множество пакетов передаётся по протоколу ICMP.

Защита:

Для противодействия атаке возможны следующие меры:

отключить ответы на ICMP-запросы;

#sysctl net.ipv4.icmp_echo_ignore_all=1

понижить приоритет обработки ICMP-сообщений;

отбросить или фильтровать ICMP-трафик межсетевым экраном;

iptables -A INPUT -p icmp -j DROP --icmp-type 8

увеличить очередь обрабатываемых подключений.

Атака с помощью SSL

Аномалия: Резкое повышение используемых системных ресурсов.

Признак: Появляется HTTPS трафик. Большое количество обращений к SSL серверу.

Защита: Можно установить правила разрыва соединения с пользователем, выполняющего функцию повторного подтверждения больше заданного количества раз в определённый период времени. Можно использовать контроллер доставки приложений (ADC), чтобы выгрузить SSL с сервера и использовать Web Application Firewall (WAF), для просмотра трафика на наличие атак. Но грамотно спланированные атаки могут превысить количество подключений к этим устройствам и нарушить их работу. Необходимо проводить поведенческий анализ. Целью этих методов является снижение объема трафика атаки пока ресурсы сервера не смогут эффективно бороться с ним. Данный тип атак не отличается поведением от легитимного трафика на сетевом уровне. Из-за этого борьба не всегда эффективна и механизмы ослабления склонны к ложным срабатываниям.

Amplification атаки

Аномалия: Огромное увеличение входящего трафика, которое может достигать до 600 Гб/с.

Признак: Увеличение входящих UDP-пакетов на порты, используемые уязвимыми сервисами. Например, NTP 123 порт, DNS 53 порт, CharGEN 19 порт.

Защита: Такие атаки фильтруются по порту источника и сигнатурам пакетов, так как для каждой amplification атаки используется определённая уязвимая команда сервиса, сигнатура которой известна.

В межсетевом экране необходимо добавить правила:

закрывать неиспользуемые UDP порты;

```
# iptables -A INPUT -p udp -j DROP
```

разрешить подключение к X UDP порту только с IP-адреса используемого сервиса;

```
# iptables -A INPUT -p udp --dport X -d x.x.x.x -j ACCEPT
```

пропускать только пакеты, которые имеют определённую сигнатуру 0x00=0x00000000.

```
# iptables -A INPUT -p udp --dport X -m u32 --u32 «0x00=0x00000000» -j ACCEPT
```

Задания к лабораторной работе:

Установить необходимое ПО и настроить сервера.

Сделать скриншоты мониторинга ресурсов сервера и сетевого трафика в штатном режиме.

Провести несколько атак из списка.

Сделать скриншоты мониторинга для каждой проведённой атаки.

Написать какие аномалии вы наблюдали в используемых ресурсах сервера, характеристиках сервера, и что является признаком атаки.

Защитить сервер от атак и сделать скриншоты результатов.

Задание №10

1. Принцип работы каждой из сетевых атак типа «отказ в обслуживании»?
2. Какие аномалии в сетевом трафике и ресурсах сервера вы заметили в каждой атаке?
3. Что является признаком конкретной атаки?
4. В чем отличие DDoS от DoS?
5. Что такое UDP-флуд?
6. Как защититься от различных атак DoS/DDoS?
7. Что такое SYN-флуд? Укажите решение защиты от SYN-флуда с помощью iptables.
8. Как определить, что осуществляется атака SYN-флуд?
9. Как осуществить защиту от HTTP-флуда?
10. Как работают атаки типа SlowLoris?
11. Как совершить DDoS-атаку на почтовый сервер и в дальнейшем реализовать его защиту?
12. Как работают атаки типа SSL? Как определить, что осуществляется атака?
13. Перечислите Amplification атаки с коэффициентами их усиления.
14. Опишите теоретически возможность расследования DoS/DDoS атак, установление личности атакующего, определение источников атак.

**Перечень источников,
дополнительной литературы, Интернет-ресурсов**

1. Электронные издания (электронные ресурсы)

1. Моргунов, А. В. Управление Веб-технологиями, сервисами и контентом : учебное пособие / А. В. Моргунов ; RU. — Новосибирск : СибГУТИ, 2021. — 88 с.
2. Кашкин, Е. В. Разработка динамических страниц на языке JavaScript с использованием библиотеки jQuery : учебно-методическое пособие / Е. В. Кашкин. — Москва : РТУ МИРЭА, 2020. — 86 с.
3. Никулова, Г. А. WEB-программирование. Серверные технологии: PHP : учебно-методическое пособие / Г. А. Никулова, В. Р. Субботин. — Липецк : Липецкий ГПУ, [б. г.]. — Часть 1 — 2017. — 59 с.
4. Нурмагомедова, Н. Х. WEB- технологии. Курс лекций : учебное пособие / Н. Х. Нурмагомедова, Г. Г. Исаева. — Махачкала : ДГПУ, 2022. — 81 с.
5. Тенгайкин, Е. А. Эксплуатация объектов сетевого администрирования. Безопасность функционирования информационных систем. Лабораторные работы : учебное пособие для спо / Е. А. Тенгайкин. — Санкт-Петербург : Лань, 2022. — 80 с.
6. JavaScript в HTML-документах : методические указания / составители А. А. Логачев, Н. Б. Смелова. — Санкт-Петербург : СПбГЛТУ, 2018. — 28 с.

Дополнительные источники (при необходимости)

7. Владимир Дронов «PHP 5/6, MySQL 5/6 и Dreamweaver CS4. Разработка интерактивных Web-сайтов» - БХВ-Петербург, 2017. - 544 с.
8. Дж. Поллок «JavaScript. Руководство разработчика» - Питер, 2017. - 537 с.
9. Джози Вернеке «Язык географической разметки KML» - ДМК Пресс, 2017. - 284 с.
10. Джон Дакетт «Основы веб-программирования с использованием HTML, XHTML и CSS» - Эксмо, 2016. - 768 с.
11. Дэвид Флэнаган «JavaScript. Подробное руководство» - Символ-плюс, 2013. - 119 с.
12. Е. Бенкен «PHP, MySQL, XML. Программирование для Интернета» - БХВ-Петербург, 2013. - 352 с.
13. Кит Вуд «Расширение библиотеки jQuery» - ДМК Пресс, 2014. - 400 с.
14. Кристиан Уэнц «PHP и MySQL. Карманный справочник» - Вильямс, 2015. - 256 с.
15. Лазаро Исси Коэн «Полный справочник по HTML, CSS и JavaScript» - ЭКОМ Паблишерз, 2016. - 233 с.
16. Люк Веллинг «Разработка веб-приложений с помощью PHP и MySQL» - Вильямс, 2013. - 848 с.
17. М. Кантелон «Node.js в действии» - Питер, 2015. - 441 с.
18. Максим Кузнецов «Головоломки на PHP для хакера» - БХВ-Петербург, 2016. - 464 с.
19. Максим Кузнецов «Самоучитель PHP 5/6» - БХВ-Петербург, 2013. - 167 с.
20. Н.В. Савельева «Основы программирования на PHP. Курс лекций» - Интернет-университет информационных технологий, 2015. - 264 с.
21. Петр Ташков «Веб-мастеринг HTML, CSS, JavaScript, PHP, CMS, AJAX, раскрутка» - Книга по Требованию, 2017. - 512 с.
22. Петцольд Чарльз, Эспозито Д. «Программирование для Microsoft Windows 8. Разработка приложений для Windows 8 на HTML5 и JavaScript (комплект из 2 книг)» - Питер, 2014. - 492 с.
23. Р. Никсон «Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript и CSS» - Питер, 2013. - 356 с.
24. Робин Никсон «Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript» - Питер, 2013. - 496 с.
25. Робин Никсон «Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5» - Питер, 2016. - 768 с.
26. Сэмми Пьюривал «Основы разработки веб-приложений» - Питер, 2014. - 659 с.
27. Тимур Машнин «Web-сервисы Java» - БХВ-Петербург, 2015. - 138 с.
28. Штефен Вальтер «Создание приложений для Windows 8 с использованием HTML5 и JavaScript» - ДМК Пресс, 2013. - 344 с.
29. Э. Уайт «PHP 5 на практике» - НТ Пресс, 2016. - 271 с.
30. Э. Фримен «Изучаем программирование на HTML5» - Питер, 2013. - 592 с.