Федеральное государственное бюджетное образовательное учреждение
высшего образования «Самарский государственный медицинский университет»
Министерства здравоохранения Российской Федерации

Платформа
Лидирующего исследовательского центра
сквозной цифровой технологии виртуальной и дополненной реальности

ИНСТРУКЦИЯ

по установке программного обеспечения

Самара, 2022 г.

Аннотация

Настоящая инструкция по установке программного обеспечения «Платформа Лидирующего исследовательского центра сквозной цифровой технологии виртуальной и дополненной реальности» (далее — Платформа ЛИЦ) является пошаговым руководством по развертыванию сервера Платформы ЛИЦ.

В документе приведены следующие сведения:

- требования к персоналу;
- требования к окружению;
- установка;
- аварийные ситуации и способы их устранения.

Оглавление

1. Требования к персоналу

Для успешного выполнения установки программного обеспечения Платформы ЛИЦ, персонал должен иметь следующие знания и навыки:

- навыки администрирования **Linux** (установка\удаление пакетов; проверка ресурсов сервера; работа с чтением лог файлов; проверка разрешений на файлы и папки; конфигурация сети, настройка iptables и т.п.);
- опыт работы с Docker (в т.ч. сборки контейнеров);
- базовые навыки администрирования веб-серверов **Apache/Nginx** (уметь настроить дефолтный сайт, настроить SSL, настроить уровни логирования);
- базовые навыки администрирования **БД (Postgres)**;
- хорошее понимание продуктивных интернет сервисов и технологий: DNS, FTP/SFTP, TCP, UDP и т.п.

2. Требования к окружению

- Операционная система Debian 9 и выше, Ubuntu 18.03 и выше;
- Docker с настроенной контейнеризацией на Linux.

3. Установка

Установку программного обеспечения следует выполнять по порядку с п.3.1 по п.3.5.18.

3.1. Настройте Consul

1. Создайте файл конфигурации `touch /etc/consul.d/consul.json` :

```
{
    "skip_leave_on_interrupt": true,
    "retry_join": [],
    "start_join": [],
    "datacenter": "lic_dc",
    "server": false,
    "ui_config": {"enabled":true},
    "bootstrap" : false,
    "client_addr": "0.0.0.0",
    "log_level": "INFO",
    "disable_host_node_id": false,
    "ports": {
        "http": 3500
    }
}
```

2. Скачайте Consul, распакуйте архив.
3. Скопируйте Consul:

```
mv consul /usr/local/bin/
cd /usr/local/bin/
chmod +x consul
```

4. Проверьте:

```
consul --version
```

5. Завершите установку:

```
consul -autocomplete-install
complete -C /usr/local/bin/consul consul
useradd --system --home /etc/consul.d --shell /bin/false consul
mkdir --parents /opt/consul
```

6. Создайте сервис запуска:

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

7. Отредактируйте `mc -e /etc/systemd/system/consul.service` для параметра `-bind` необходимо указать физический IP-адрес машины:

```
[Unit]
Description="HashiCorp Consul - SamSMU Service discovery"
Documentation=https://www.consul.io/
Requires=network-online.target
After=network-online.target
#ConditionFileNotEmpty=/etc/consul.d/consul.hcl

[Service]
#User=consul
#Group=consul
ExecStart=/usr/local/bin/consul  agent  -config-dir=/etc/consul.d/  -
bind=192.168.6.20 -node=prod
ExecReload=/usr/local/bin/consul reload
KillMode=process
Restart=on-failure
LimitNOFILE=65536

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

8. Запустите Consul:

```
systemctl enable consul
systemctl start consul
systemctl status consul
```

Ссылка на официальную документацию:

https://learn.hashicorp.com/consul/advanced/day-1-operations/deployment-guide

Примечание. Для восстановления настроек необходимо выполнить команду:
`consul backup restore <путь до файла бэкапа>/consul.snp`

3.2. Настройте Docker

1. CommonAccessService.docker.node.json

```
{
    "CommonAccessService": {
        "SelfAddress":  "192.168.6.20",  //прописать  физический  IP-адрес
локальной машины( localhost не подходит)
        "Port": 20350
    }
}
```

2. appsettings.docker.node.json (вместо http://192.168.6.20:3500/ укажите адрес до Consul из раздела 3.1.):

```
{
    "ServiceDiscovery": {
        "HttpEndpoint":  "http://192.168.6.20:3500",  //прописать  путь  до
consul
        "SelfAddress":  "192.168.6.20"  //прописать  физический  IP-адрес
локальной машины( localhost не подходит)
    },
    "DataMigration": true
}
```

3. consulbeat.yml (вместо http://192.168.6.20:3500/ укажите адрес до Consul из раздела 3.1.):

```
########################## Consulbeat ############################
consulbeat:
  # Defines how often an event is sent to the output
  period: 10s
  # Defines the consul url
  consul_url: "http://192.168.6.20:3500/" # прописать путь до consul
  fail_on_http_error: false
#----------------------- Logstash output ------------------------
output.logstash:
  hosts: ["plat-consulbeat-logstash:15055"]
#========================= Logging ==============================
logging.level: info
logging.selectors: ["*"]}
```

4. logstash.consulbeat.yml

```
---
## Default Logstash configuration from logstash-docker.
##                  from                 https://github.com/elastic/logstash-
docker/blob/master/build/logstash/config/logstash-oss.yml
#
http.host: "0.0.0.0"
path.config: /etc/logstash/pipeline
path.logs: /var/log/logstash
```

5. logstash.consulbeat.conf

```
input {
  beats {
    port => 15055
  }
}


# First filter
filter {
  #ignore log comments
  if [message] =~ "^#" {
    drop {}
  }

  grok {
    break_on_match => false
    patterns_dir => "./patterns"
    match => [
      "message" , "%{TIMESTAMP_ISO8601:timestamp} %{IPORHOST:serverip} %{WORD:verb}
%{NOTSPACE:request} %{NOTSPACE:querystring} %{NUMBER:port} %{NOTSPACE:auth}
%{IPORHOST:clientip} %{NOTSPACE:agent} %{NOTSPACE:referrer} %{NUMBER:response
} %{NUMBER:sub_response} %{NUMBER:sc_status} %{NUMBER:responsetime}",
      "message" , "%{TIMESTAMP_ISO8601:timestamp} %{IPORHOST:serverip} %{WORD:verb}
%{NOTSPACE:request} %{NOTSPACE:querystring} %{NUMBER:port} %{NOTSPACE:auth}
%{IPORHOST:clientip} %{NOTSPACE:agent} %{NUMBER:response} %{NUMBER:sub_respon
se} %{NUMBER:sc_status} %{NUMBER:responsetime}",
      "message" , "%{TIMESTAMP_ISO8601:timestamp} %{IPORHOST:serverip} %{WORD:verb}
%{NOTSPACE:request} %{NOTSPACE:querystring} %{NUMBER:port} %{NOTSPACE:auth}
%{IPORHOST:clientip} %{NOTSPACE:agent} %{NUMBER:response} %{NUMBER:sub_respon
se} %{NUMBER:sc_status}",
      "querystring", "(?:\&SERVER-ROUTED=(?<routed_server>[\.,\w]+)\&?)",
      "querystring", "(?:\&SERVER-STATUS=(?<routed_status>[\d]+)\&*)",
      "routed_server", "^(?:(?<service_environment>[\w]+))\.(?:(?<service_name>[\w,\-
,_]+))\.[\w]+$"
    ]
  }
  date {
    match => [ "timestamp", "yyyy-MM-dd HH:mm:ss", "ISO8601" ]
    target => "@timestamp"
    locale => "en"
    timezone => "UTC"
  }
}


output {
  elasticsearch {
    hosts =>  ["http://es1:9200"]
    index => "consulbeat-plat--%{+YYYY.MM.dd}"
    template_name => "consulbeat-plat"
    template_overwrite => false
    manage_template => true
  }
}
```

6. logstash.logs.yml

```
---
## Default Logstash configuration from logstash-docker.
## from https://github.com/elastic/logstash-
docker/blob/master/build/logstash/config/logstash-oss.yml
#
http.host: "0.0.0.0"
path.config: /etc/logstash/pipeline
path.logs: /usr/share/logstash/logs
C/C++ detected
```

7. logstash.logs.conf

```
input {
   beats {
        port => "15055"
    }
}

filter {
  json {
    source => "message"
    target => "parsedJson"
  }
  mutate {
    rename => {
      "[parsedJson][Date]" => "record_time"
      "[parsedJson][Level]" => "Level"
      "[parsedJson][Host]" => "Host"
      "[parsedJson][Machine]" => "routed_server"
      "[parsedJson][Env]" => "service_environment"
      "[parsedJson][Metrics][StatusCode]" => "routed_status"
      "[parsedJson][Metrics][RequestMethod]" => "http_method"
      "[parsedJson][Metrics][RequestPath]" => "url"
      "[parsedJson][Metrics][Elapsed]" => "responsetime"
      "[parsedJson][Metrics][Responded]" => "respondedtime"
      "[parsedJson][Metrics][ServiceName]" => "service_name"
      "[parsedJson][Metrics][Headers]" => "headers_1"
    }
  }

  if [headers_1] !~ /.+/
  {
     if ![headers_1][X-Real-IP]
     {
         ruby {  code => "event.get('headers_1').each {|hash|
event.set(hash['Key'], hash['Value']) }" }
     }else
     {
         mutate {
            add_field => { "X-Real-IP" => "%{[headers_1][X-Real-IP]}" }
         }
     }
  }

  mutate {
     rename => { "X-Real-IP" => "real_ip" }
  }

  geoip { source => "real_ip" }

  mutate {
    convert => [ "routed_status","integer" ]
    convert => [ "responsetime","float" ]
    convert => [ "respondedtime","float" ]
  }
  date {
    match => ["record_time", "yyyy-MM-dd HH:mm:ss.SSSS"]
    timezone => "UCT"
    target => "@timestamp"
  }

  if [url][Value]
  {
    mutate { add_field => { "new_url" => "%{[url][Value]}" } }
  }
  else
  {
     mutate { rename => { "[url]" => "new_url" } }
  }
```

```
grok {
     match => { "[new_url]" =>
"^/(?<api_direct>[\w]+)/(?<api_controller>[\w\-\d]+)(?<api_method>/[\w\-
\d]+)*" }
     add_tag => [ "success_grok" ]
  }

  mutate {
    remove_field => [ "url" ]
  }

  mutate {
    rename => { "new_url" => "url" }
  }

  if "success_grok" in [tags]
  {
  }else  {  drop { }   }

  if [api_controller] == "HealthCheck" {
      drop { }
  }
}

output {
  elasticsearch {
    hosts => "es1:9200"
    index => "logs-plat-%{+YYYY.MM.dd}"
  }
}
```

8. filebeat.logs.yml

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/logs/Timings/*/*.log
    - /var/logs/Timings/*/*/*.log
    - /var/logs/Timings/*/*/*/*.log
    - /var/logs/Timings/*/*/*/*/*.log
    - /var/logs/Timings/*/*/*/*/*/*.log
  exclude_lines: ['"RequestPath":"/sys/HealthCheck"']
logging.metrics.enabled: false
output.logstash:
  hosts: ["plat-logstash-logs:15055"]
tags: ["production","docker"]
```

3.3. Создайте docker-secrets

1. plat_connectionstring_production

```
User
ID=plat_user_admin;Password=123123123;Host=postgres;Port=5432;Database=plat_prod;
Pooling=true;
```

2. ConnectionStrings_catalog_ru_ru_production

```
User
ID=plat_user_admin;Password=123123123;Host=postgres;Port=5432;Database=plat_prod;
Pooling=true;
```

3. plat_hangfire_reports_production

```
User
ID=plat_user_admin;Password=123123123;Host=postgres;Port=5432;Database=plat_hf_re
ports_prod;Pooling=true;
```

4. plat_hangfire_marketplaces_production

```
User
ID=plat_user_admin;Password=123123123;Host=postgres;Port=5432;Database=plat_hf_ma
rketplaces_prod;Pooling=true;
```

5. pgbouncer_users

```
"plat_user_admin" "123123123"
```

3.4. Создайте в докере Volumes FileStorage

3.5. Установите модули:

3.5.1. База данных — PostgreSQL

Создайте файл `/usr/inst/docker-compose.psql.yml` и выполните команду

```
docker-compose /usr/inst/docker-compose.psql.yml up -d
```

```
version: "3.8"

services:
  postgres-prod-db:
    image: postgres:13
    command: postgres -c shared_buffers=256MB -c max_connections=600
    volumes:
      - postgresdb_cache:/tmp/cache
      - db_local:/var/lib/postgresql/data

#  pgbouncer:
# fix: чтобы не править строки подключения
  postgres:
    image: edoburu/pgbouncer
    environment:
      - DB_HOST=postgres-prod-db
      - MAX_CLIENT_CONN=2000
      - LISTEN_PORT=5432
      - AUTH_FILE=/run/secrets/pgbouncer_users
      - AUTH_TYPE=md5
      - POOL_MODE=session
      - DEFAULT_POOL_SIZE=100
      - IGNORE_STARTUP_PARAMETERS=extra_float_digits
      - DB_USER=plat_user_admin
    secrets:
      - pgbouncer_users
    volumes:
      - pgbouncer-log-prod:/postgresql/logs
      - pgbouncer-run-prod:/var/run/postgresql

secrets:
  pgbouncer_users:
    external: true
    name: pgbouncer_users_prod

volumes:
  pgbouncer-log-prod:
  pgbouncer-run-prod:
  postgresdb_cache:
  db_local:
```

3.5.2. Сервис для хранения логов — Elasticsearch

Создайте файл `/usr/inst/docker-compose.es.yml` и выполните команду

```
docker-compose /usr/inst/docker-compose.es.yml up -d
```

```
version: "3.8"

services:
  es01:
    image: docker.elastic.co/elasticsearch/elasticsearch:7.13.0
    container_name: es01
    environment:
      - node.name=es01
      - cluster.name=es-docker-cluster-prod
      - cluster.initial_master_nodes=es01
      - node.max_local_storage_nodes=3
      - "ES_JAVA_OPTS=-Xms2g -Xmx2g"
    ulimits:
      memlock:
        soft: -1
        hard: -1
    volumes:
      - elasticsearch_data:/usr/share/elasticsearch/data

  kib01:
    image: docker.elastic.co/kibana/kibana:7.1.1
    container_name: kib01
    depends_on:
      - es01
    ports:
      - 18601:5601
    environment:
      SERVERNAME: localhost
      ELASTICSEARCH_URL: http://es01:9200
      ELASTICSEARCH_HOSTS:   http://es01:9200
      ELASTICSEARCH_USERNAME: admin
      ELASTICSEARCH_PASSWORD: 123456

  cerebro:
    image: lmenezes/cerebro:latest
    container_name: cerebro
    ports:
      - 18000:9000
    volumes:
      - cerebro_d:/opt/cerebro

volumes:
  cerebro_d:
    driver: local
  elasticsearch_data:
    driver: local
```

3.5.3. Сборщик логов Logstash+filebeat

1. Создайте папку `mkdir /usr/share/logs/Timings`
2. Создайте файл `/usr/inst/docker-compose.logstash.yml` и выполните команду
   `docker-compose /usr/inst/docker-compose.logstash.yml up -d`

```
version: "3.7"

services:
  plat-logstash-logs:
    image: docker.elastic.co/logstash/logstash-oss:7.1.1
    environment:
      LS_JAVA_OPTS: "-Xmx1g -Xms1g"
    labels:
      traefik.enable: "false"
       configs:
      - source: logstash.logs.yml
        target: /etc/logstash/config/logstash.yml
      - source: logstash.logs.conf
        target: /etc/logstash/pipeline/logstash.conf

  filebeat:
    image: docker.elastic.co/beats/filebeat:7.9.1
    hostname: "{{.Node.Hostname}}-filebeat"
    user: root
    configs:
      - source: filebeat.logs.yml
        target: /usr/share/filebeat/filebeat.yml
    volumes:
      - /usr/share/logs/Timings/:/var/logs/Timings/:ro
    command: ["--strict.perms=false"]

configs:
  consulbeat.yml:
    external: true
  logstash.logs.yml:
    external: true
  logstash.logs.conf:
    external: true
  filebeat.logs.yml:
    external: true
```

3.5.4. Настройка системы мониторинга — Graphana

1. Создайте файл `/usr/inst/docker-compose.graphana.yml` и выполните команду
   `docker-compose /usr/inst/docker-compose.graphana.yml up -d`

```
version: "3.7"

services:
  grafana:
    image: grafana/grafana
    ports:
      - 3000:3000
    volumes:
      - grafana_data:/var/lib/grafana

volumes:
  grafana_data:
```

2. Создайте источники данных:
    1. consulbeat.plat
        0. Name: consulbeat.plat
        1. URL: http://es1:9200
        2. index name: [consulbeat-plat--]YYYY.MM.DD
        3. pattern: Daily

4. timefield: @timestamp
5. version: 7.0+
6. Max connection shard request: 20

2. Plat.card
0. Name: Plat.card
1. URL: http://es1:9200
2. index name: [logs-plat-]YYYY.MM.DD
3. pattern: Daily
4. timefield: @timestamp
5. version: 7.0+
6. Max connection shard request: 20

3. Создайте любые 3 notification channel
4. Создайте dashboard через импорт файла:Plat.Services-1654723764463.json

3.5.5. Настройка consulbeat — Healthcheck beatlogger

Создайте файл /usr/inst/docker-compose.consulbeat.yml и выполните команду

```
docker-compose /usr/inst/docker-compose.consulbeat.yml up -d
```

```
version: "3.7"

services:
  consulbeat:
    hostname: "plat-consulbeat-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/consulbeat-5:latest-linux"
    command: './consulbeat-linux-amd64 -c /etc/consulbeat/consulbeat.yml -e
-d * '
    restart: always
        configs:
      - source: consulbeat.yml
        target: /etc/consulbeat/consulbeat.yml

  plat-consulbeat-logstash:
    image: docker.elastic.co/logstash/logstash-oss:7.1.1
    ports:
      - 15055:15055
    environment:
      LS_JAVA_OPTS: "-Xmx1g -Xms1g"
    labels:
      traefik.enable: "false"
         configs:
      - source: logstash.consulbeat.yml
        target: /etc/logstash/config/logstash.yml
      - source: logstash.consulbeat.conf
        target: /etc/logstash/pipeline/logstash.conf

configs:
  consulbeat.yml:
    external: true
  logstash.consulbeat.yml:
    external: true
  logstash.consulbeat.conf:
    external: true
C# detected
```

3.5.6. Сервис прав доступа

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/Accesses
mkdir /usr/share/logs/Timings/Accesses
```

2. Создайте файл `/usr/inst/docker-compose.access.yml` и выполните команду `docker-compose /usr/inst/docker-compose.access.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-accesses-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.accesses.serviceweb:latest-linux"
    command: "--ServiceDiscovery:Port=20350"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: CommonAccessService.docker.node.json
        target: /app/CommonAccessService.docker.node.json
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/Accesses:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/Accesses:/usr/logs/Timings
    ports:
      - 20350:80

configs:
  CommonAccessService.docker.node.json:
    external: true
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.7. Сервис общих справочников

1. Создайте на локальной машине пути:

   ```
   mkdir /usr/share/logs/Globals
   mkdir /usr/share/logs/Timings/Globals
   ```

2. Создайте `docker secrets plat_connectionstring_production` и положите строку подключения к БД
3. Создайте файл `/usr/inst/docker-compose.Globals.yml` и выполнить команду `docker-compose /usr/inst/docker-compose.Globals.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-globals-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.globals.serviceweb:latest-linux"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
      - DB_CONNECTIONS_STRING="cat /run/secrets/plat_connectionstring"
    command: "--ServiceDiscovery:Port=20355 --
ConnectionStrings:ais=/run/secrets/plat_connectionstring_production"
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/Globals:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/Globals:/usr/logs/Timings
    ports:
      - 20355:80
    secrets:
      - plat_connectionstring_production

secrets:
  plat_connectionstring_production:
    external: true

configs:
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.8. Сервис потоковых данных

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/FileStorages
mkdir /usr/share/logs/Timings/FileStorages
```

2. Создайте файл `/usr/inst/docker-compose.filestorages.yml` и выполните команду `docker-compose /usr/inst/docker-compose.filestorages.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-filestorages-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.filestorages.serviceweb:latest-
linux"
    command: "--ServiceDiscovery:Port=20354"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - FileStorage:/IExercise
      - /usr/share/logs/FileStorages:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/FileStorages:/usr/logs/Timings
    ports:
      - 20354:80

configs:
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
C# detected
```

3.5.9. Сервис заметок

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/Records
mkdir /usr/share/logs/Timings/Records
```

2. Создайте файл `/usr/inst/docker-compose.Records.yml` и выполните команду `docker-compose /usr/inst/docker-compose.Records.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-records-web-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.records.serviceweb:latest-linux"
    command: "--ServiceDiscovery:Port=20358"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - /usr/share/fs:/FileStorage
      - /usr/share/logs/Records:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/Records:/usr/logs/Timings
    ports:
      - 20358:80

configs:
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.10. Сервис настроек внешних ПО

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/Exercises
mkdir /usr/share/logs/Timings/Exercises
```

2. Создайте файл `/usr/inst/docker-compose.Exercises.yml` и выполните команду `docker-compose /usr/inst/docker-compose.Exercises.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-exercises-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.exercises.serviceweb:latest-linux"
    command: "--ServiceDiscovery:Port=20327"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/Exercises:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/Exercises:/usr/logs/Timings
    ports:
      - 20327:80

configs:
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.11. Сервис форума

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/forum/web
mkdir /usr/share/logs/Timings/forum/web
```

2. Создайте файл `/usr/inst/docker-compose.forum.web.yml` и выполните команду
   `docker-compose /usr/inst/docker-compose.forum.web.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "reviais-forum-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.reviais.forum.serviceweb:latest-linux"
    command: "--ServiceDiscovery:Port=20370"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/forum/web:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/forum/web:/usr/logs/Timings
    ports:
      - 20370:80

configs:
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.12. Сервис заказов, обращений

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/Appointments
mkdir /usr/share/logs/Timings/Appointments
```

2. Создайте файл `/usr/inst/docker-compose.appointments.yml` и выполните команду `docker-compose /usr/inst/docker-compose.appointments.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-appointments-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.appointments.serviceweb:latest-
linux"
    command: "--ServiceDiscovery:Port=20351"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/Appointments:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/Appointments:/usr/logs/Timings
    ports:
      - 20351:80

configs:
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
C# detected
```

3.5.13. Сервис покупок, билинга

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/marketplaces/bg
mkdir /usr/share/logs/Timings/marketplaces/bg
```

2. создать файл `/usr/inst/docker-compose.marketplaces.yml` и выполните команду `docker-compose /usr/inst/docker-compose.marketplaces.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-marketplaces-web-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.marketplaces.serviceweb:latest-
linux"
    command: "--ServiceDiscovery:Port=20373"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/marketplaces/web:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/marketplaces/web:/usr/logs/Timings
    ports:
      - 20373:80

  background:
    hostname: "plat-marketplaces-bg-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.marketplaces.background:latest-
linux"
    command: "--ServiceDiscovery:Port=20380 --console"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/marketplaces/bg:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/marketplaces/bg:/usr/logs/Timings
    ports:
      - 20380:80
    secrets:
      - source: plat_hangfire_marketplaces_production
        target: ConnectionStrings_hangfire_production


configs:
  appsettings.docker.node.json:
    external: true

secrets:
  plat_hangfire_marketplaces_production:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.14. Сервис каталога

1. Создайте на локальной машине пути:

   ```
   mkdir /usr/share/logs/samsmu.plat.catalog
   mkdir /usr/share/logs/Timings/samsmu.plat.catalog
   ```

2. Создайте `docker secrets ConnectionStrings_catalog_ru_ru_production` и положить строку подключения как для сервиса общих справочников
3. Создайте файл `/usr/inst/docker-compose.catalog.yml` и выполните команду `docker-compose /usr/inst/docker-compose.catalog.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-catalog-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.catalog.serviceweb:latest-linux"
    command: "--ServiceDiscovery:Port=20352 --
Background:ServiceName='Catalog.Service.ru_ru'"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/samsmu.plat.catalog:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/samsmu.plat.catalog:/usr/logs/Timings
    ports:
      - 20352:80
    secrets:
      - source: ConnectionStrings_catalog_ru_ru_production
        target: ConnectionStrings_ais_production

secrets:
  ConnectionStrings_catalog_ru_ru_production:
    external: true

configs:
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.15. Сервис посетителей

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/actors
mkdir /usr/share/logs/Timings/actors
```

2. Создайте файл `/usr/inst/docker-compose.actors.yml` и выполните команду `docker-compose /usr/inst/docker-compose.actors.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-actors-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.actors.serviceweb:latest-linux"
    command: "--ServiceDiscovery:Port=20359"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/actors:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/actors:/usr/logs/Timings
    ports:
      - 20359:80

configs:
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.16. Сервис отчетов

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/reports/web
mkdir /usr/share/logs/Timings/reports/web
mkdir /usr/share/logs/reports/bg
mkdir /usr/share/logs/Timings/reports/bg
mkdir /usr/share/tmp/ReviReports
```

2. Создайте файл `/usr/inst/docker-compose.reports.yml` и выполните команду `docker-compose /usr/inst/docker-compose.reports.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-reports-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.reports.serviceweb:latest-linux"
    command: "--ServiceDiscovery:Port=20721"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/reports/web:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/reports/web:/usr/logs/Timings
    ports:
      - 20721:80

  background-observer:
    hostname: "plat-reports-bg-observer-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.reports.background:latest-linux"
    command: "--ServiceDiscovery:Port=20723 --
BackgroundSettings:OverrideQueues='Observer' --
Background:ServiceName='Reports.Observer.Background.Service' --console"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/reports/bg:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/reports/bg:/usr/logs/Timings
      - /usr/share/tmp/ReviReports:/tmp/ReviReports
    ports:
      - 20723:80
    secrets:
      - source: plat_hangfire_reports_production
        target: ConnectionStrings_hangfire_production

configs:
  appsettings.docker.node.json:
    external: true

secrets:
  plat_hangfire_reports_production:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.17. Сервис рейтингов, опросов

1. Создайте на локальной машине пути:

```
mkdir /usr/share/logs/Screenings/web
mkdir /usr/share/logs/Timings/Screenings/web
mkdir /usr/share/logs/Screenings/bg
mkdir /usr/share/logs/Timings/Screenings/bg
```

2. Создайте файл `/usr/inst/docker-compose.screenings.yml` и выполните команду `docker-compose /usr/inst/docker-compose.screenings.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-screenings-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.screenings.serviceweb:latest-linux"
    command: "--ServiceDiscovery:Port=20361"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/Screenings/web:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/Screenings/web:/usr/logs/Timings
    ports:
      - 20361:80

  background:
    hostname: "plat-screenings-background-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.screenings.background:latest-linux"
    command: "--ServiceDiscovery:Port=22444 --console"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - /usr/share/fs:/FileStorage
      - /usr/share/logs/Screenings/bg:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/Screenings/bg:/usr/logs/Timings
    ports:
      - 22444:80
    secrets:
      - source: plat_hangfire_screenings_production
        target: ConnectionStrings_hangfire_production

configs:
  appsettings.docker.node.json:
    external: true

secrets:
  plat_hangfire_screenings_production:
    external: true

volumes:
  FileStorage:
    external: true
```

3.5.18. Сервис пользователей системы

1. Создайте на локальной машине пути:

   ```
   mkdir /usr/share/logs/Users
   mkdir /usr/share/logs/Timings/Users
   ```

2. Создайте файл `/usr/inst/docker-compose.Users.yml` и выполните команду
   `docker-compose /usr/inst/docker-compose.Users.yml up -d`

```
version: "3.7"

services:
  serviceweb:
    hostname: "plat-users-serviceweb-lx-{{.Node.Hostname}}"
    image: "{dockerhub_repository}/samsmu.plat.users.serviceweb:latest-linux"
    command: "--ServiceDiscovery:Port=20362"
    environment:
      - ASPNETCORE_ENVIRONMENT=production
    restart: always
    configs:
      - source: appsettings.docker.node.json
        target: /app/appsettings.docker.node.json
    volumes:
      - FileStorage:/FileStorage
      - /usr/share/logs/Users:/usr/logs/ServiceWeb
      - /usr/share/logs/Timings/Users:/usr/logs/Timings
    ports:
      - 20362:80

configs:
  appsettings.docker.node.json:
    external: true

volumes:
  FileStorage:
    external: true
C# detected
Настройка событий
```

4. Аварийные ситуации и способы их устранения

Не работает протокол TLS

Если у вас операционная система Ubuntu 20 или выше, вы можете столкнуться с проблемой, когда данные по протоколу TLS не передаются.

Чтобы устранить эту проблему, необходимо выполнить следующие действия:

1. добавьте в файл `etc/ssl/openssl.cnf` следующий код:

```
openssl_conf = default_conf

[ default_conf ]
ssl_conf = ssl_sect

[ ssl_sect ]
system_default = system_default_sect

[ system_default_sect ]
MinProtocol = TLSv1
DEFAULT@SECLEVEL = 1
```

2. отредактируйте конфигурацию `nginx` и внесите изменения в список шифров, чтобы добавить псевдошифр `@SECLEVEL=1`.

Пример:

```
ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3;
ssl_ciphers "EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH";
```

станет

```
ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3;
# seclevel for TLS 1.0 and 1.1
ssl_ciphers "EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH:@SECLEVEL=1";
```