

Схема сети	2
Часть 1	3
1 Настройка сетевых интерфейсов InfoWatch ARMA NGFW	3
2 Настройка трансляции адресов	4
3 Настройка IDS/IPS Suricata	4
3.1 Тестирование обнаружения вредоносного подключения	5
3.2 Тестирование эксплоита для CVE-2021-41773	6
3.3 Написание своего правила	7
4 Создание отказоустойчивого кластера	8
4.1 Настройка ведущего устройства	8
4.2 Настройка резервного устройства	9
4.3 Тестирование работы кластера	10
5 Настройка сбора статистики по трафику	11
5.1 Настройка Flow-accounting на NGFW	12
5.2 Настройка NetFlow-коллектора nprobe	12
5.3 Настройка NetFlow-монитора ntopng	13
5.4 Результат работы	13
Часть 2	15
1 Установка RedOS Serv и шифрование диска	15
2 Результат работы контейнеров docker-compose	16
3 Результат работы Zabbix-dashboard	16
4 Результат работы Django	17
5 Результат работы pgAdmin и подключения Django и Zabbix	18
6 Тестирование создания и восстановления бэкапов	20

Схема сети

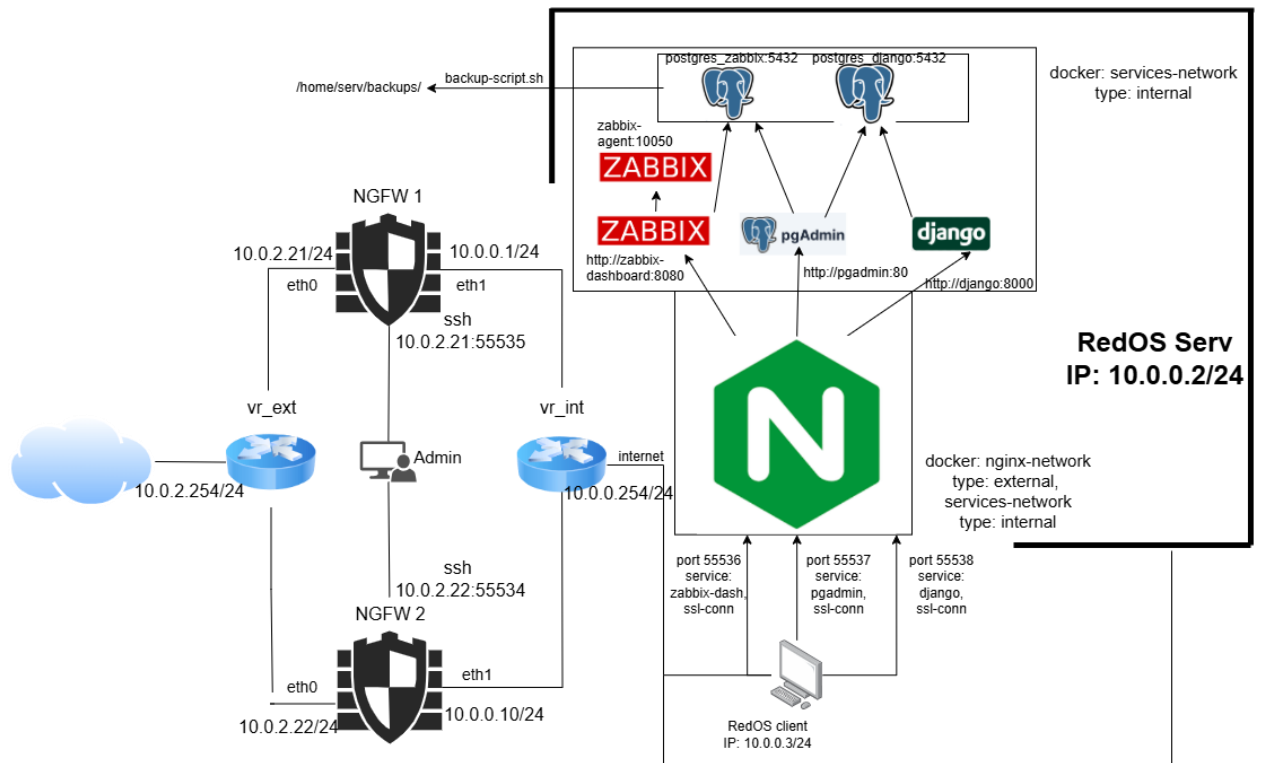


Рисунок 1 – Схема сети

Часть 1

1 Настройка сетевых интерфейсов InfoWatch ARMA NGFW

После установки ARMA и выполнения команды `install image` на виртуальную машину VirtualBox зададим админу другой пароль

```
set system login user admin authentication plaintext-  
password xC5i20YMEK36EB
```

Далее настроим сетевые интерфейсы (режим `configure`).

```
set interfaces ethernet eth0 address dhcp
```

```
set interfaces ethernet eth1 address 10.0.0.1/24
```

Далее зададим DNS-сервер

```
set system name-server 8.8.8.8
```

И настроим подключение по ssh-порту

```
set service ssh port 55535
```

```
set service ssh listen-address 10.0.2.21
```

Интерфейсы:

`eth0` – для подключения к внешней сети

`eth1` – для подключения к локальной сети.

Для включения IPv4-форвардинга добавляем параметр `net.ipv4.ip_forward=1` в файл `/etc/sysctl.conf`, после применяем `sysctl -p`. Настройки конфигурации интерфейсов приведены на рисунке 2.

```
admin@ngfwos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface    IP Address      MAC              VRF      MTU    S/L    Description
-----
eth0          10.0.2.21/24    08:00:27:de:a8:42 default  1500   u/u
              10.0.2.254/24
eth1          10.0.0.1/24     08:00:27:4b:e2:26 default  1500   u/u
              10.0.0.254/24
lo            127.0.0.1/8     00:00:00:00:00:00 default  65536  u/u
              ::1/128
```

Рисунок 2 – Конфигурация интерфейсов

2 Настройка трансляции адресов

Настроим трансляцию адресов (Source NAT), добавим соответствующие правила для преобразования внутренних IP-адресов в публичный внешний адрес. Для этого введём следующие команды:

```
set nat source rule 2 outbound-interface name eth0
set nat source rule 2 protocol all
set nat source rule 2 source address 10.0.0.0/24
set nat source rule 2 translation address masquerade
```

Проверим работу NAT (рисунок 3), сделав traceroute до сайта bmstu.ru с машины RedOS Client (шлюз по умолчанию выбрал 10.0.0.1).

```
[client@localhost ~]$ traceroute bmstu.ru
traceroute to bmstu.ru (195.19.50.250), 30 hops max, 60 byte packets
 1 _gateway (10.0.0.1)  0.407 ms  0.360 ms  0.332 ms
 2 10.0.2.1 (10.0.2.1)  1.296 ms  1.383 ms  1.358 ms
 3 * * *
```

Рисунок 3 – Проверка работоспособности NAT

3 Настройка IDS/IPS Suricata

Для начала загрузим базу правил через команду

```
sudo suricata-update
```

Далее в режиме конфигурирования настроим suricata.

```
set suricata enable yes
```

```

set suricata stats enabled yes

set suricata outputs fast enabled yes

set suricata netmap interface eth0 copy-mode tap

set suricata update-rules local-storage LOCAL path
/var/lib/suricata/rules

configure

save

suricata update-rules local-storage LOCAL

```

На рисунке 4 показан импорт правил из директории /var/lib/suricata/rules.

```

admin@ngfwos:/var/lib/suricata/rules$ suricata update-rules local-storage LOCAL
Try to get rules from storage "LOCAL"...
Loaded 58042 rules.
Disabled 0 rules.
Enabled 0 rules.
Modified 0 rules.
Dropped 0 rules.

```

Рисунок 4 – Импорт правил

3.1 Тестирование обнаружения вредоносного подключения.

В базе правил найдём какой-нибудь вредоносный IP-адрес для тестирования обнаружения подключения к нему. Правило с sid **2404301** классифицирует подключение к IP **149.28.156.183** как вредоносное (сервис содержит банковский троян), правило изображено на рисунке 5.

```

admin@ngfwos:/var/lib/suricata/rules$ cat suricata.rules | grep 149.28.156.183
alert ip $HOME_NET any -> [149.28.156.183] any (msg:"ET CNC Feodo Tracker Reported CnC Server group 2"; reference:url,doc.emergingthreats.net/bin/
view/Main/BotCC; reference:url,feodotracker.abuse.ch; threshold: type limit, track by_src, seconds 3600, count 1; flowbits:set,ET.Evil; flowbits:s
et,ET.BotccIP; classtype:trojan-activity; sid:2404301; rev:7526; metadata:affected_product Windows_XP_Vista_7_8_10_Server_32_64_Bit, attack_target
Client_Endpoint, deployment Perimeter, tag Banking_Trojan, signature_severity Major, created_at 2014_11_04, updated_at 2025_04_14;)

```

Рисунок 5 – Одно из вредоносных правил Suricata

С ПК RedOS Client попробуем подключиться по http к вредоносному IP

```
curl http://149.28.156.183
```

В файле **fast.log** сработал детект данного правила (рисунок 6)

```
admin@ngfwos:/var/lib/suricata/rules$ cat /var/log/suricata/fast.log | grep 2404301
04/15/2025-16:30:08.840196  [**] [1:2404301:7526] ET CNC Feodo Tracker Reported CnC Server group 2 [**] [Classification: A Network Trojan was detected] [Priority: 1] {TCP} 10.0.2.21:42802 -> 149.28.156.183:80
04/15/2025-17:00:20.521671  [**] [1:2404301:7526] ET CNC Feodo Tracker Reported CnC Server group 2 [**] [Classification: A Network Trojan was detected] [Priority: 1] {TCP} 10.0.0.3:49314 -> 149.28.156.183:80
04/15/2025-17:37:51.636128  [**] [1:2404301:7526] ET CNC Feodo Tracker Reported CnC Server group 2 [**] [Classification: A Network Trojan was detected] [Priority: 1] {TCP} 10.0.2.21:38962 -> 149.28.156.183:80
admin@ngfwos:/var/lib/suricata/rules$
```

Рисунок 6 – Детект правила № 2404301

3.2 Тестирование эксплоита для CVE-2021-41773.

Предположим, что у компании есть сервер, работающий на 80 порту. Развернём Apache-сервер на RedOS Client.

Далее поднимем Ubuntu-машину во внешней сети для тестирования эксплоита. Её адрес – **10.0.2.6**. Для проброса 80 порта во внутреннюю сеть компании необходимо использовать DNAT:

```
set nat destination rule 3 inbound-interface name eth0
set nat destination rule 3 protocol tcp
set nat destination rule 3 destination port 80
set nat source rule 3 translation address 10.0.0.3
set nat source rule 3 translation port 80
```

Проверим доступность Apache с внешней Ubuntu-машины (Рисунок 7).

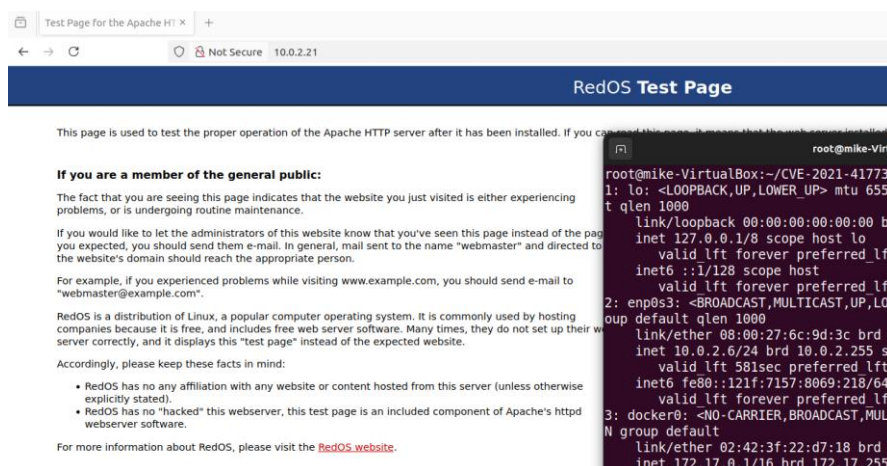


Рисунок 7 – Доступ до Apache с внешней машины.

Запустим эксплоит (Рисунок 8).



Рисунок 10 – Скачивание ВПО.

Далее переведём Suricata в режим IPS и заменим ключевое слово alert на drop. Правило, детект и блокирование загрузки файлов приведены на рисунке 11.1 и 11.2.

```
admin@ngfwos:/var/lib/suricata/rules$ cat custom.rules | grep 1000034
drop http $EXTERNAL_NET any -> $HOME_NET any (msg: "ALERT - Download of malwar_for_ITPlanet.exe detected"; flow: established, to_client; http.response_body; content: "malwar_for_ITPlanet.exe"; nocase; classtype:trojan-activity; sid:1000034; rev:1;)
```

Рисунок 11.1 – Правило для обнаружения/блокирования скачивания

```
admin@ngfwos:/var/lib/suricata/rules$ cat /var/log/suricata/fast.log | grep 1000034
04/15/2025-18:15:01.813463  [**] [1:1000034:1] ALERT - Download of malwar_for_ITPlanet.exe detected [**] [Classification: A Network Trojan was detected] [Priority: 1] {TCP} 185.173.93.71:54535 -> 10.0.2.21:41024
04/15/2025-18:22:12.648909  [wDrop] [**] [1:1000034:1] ALERT - Download of malwar_for_ITPlanet.exe detected [**] [Classification: A Network Trojan was detected] [Priority: 1] {TCP} 185.173.93.71:54535 -> 10.0.2.21:33940
04/15/2025-18:22:13.415468  [wDrop] [**] [1:1000034:1] ALERT - Download of malwar_for_ITPlanet.exe detected [**] [Classification: A Network Trojan was detected] [Priority: 1] {TCP} 185.173.93.71:54535 -> 10.0.2.21:33958
```

Рисунок 11.2 – Детекты правила 1000034

4 Создание отказоустойчивого кластера.

4.1 Настройка ведущего устройства

Настроим основное устройство NGFW1. Для этого создадим и настроим виртуальный маршрутизатор vr_ext.

```
set high-availability vrrp group vr_ext

edit high-availability vrrp group vr_ext

set interface eth0

set vrid 100

set address 10.0.2.254/24

set priority 100
```



```
commit
```

```
save
```

Далее создадим и настроим виртуальный маршрутизатор `vr_int`.

```
set high-availability vrrp group vr_int
```

```
edit high-availability vrrp group vr_int
```

```
set interface eth1
```

```
set vrid 100
```

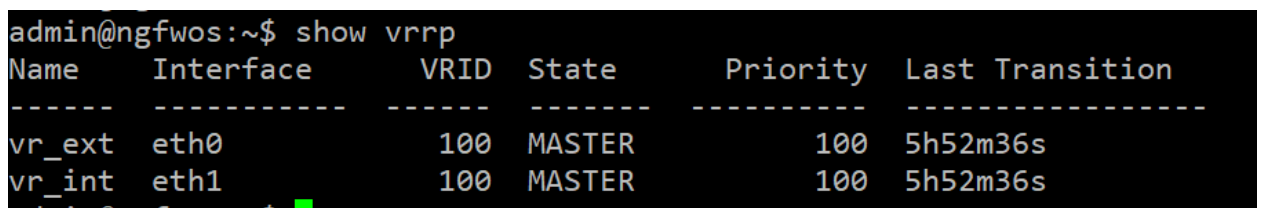
```
set address 10.0.0.254/24
```

```
set priority 100
```

```
commit
```

```
save
```

На рисунке 12 изображены настройки VRRP ведущего устройства.



Name	Interface	VRID	State	Priority	Last Transition
vr_ext	eth0	100	MASTER	100	5h52m36s
vr_int	eth1	100	MASTER	100	5h52m36s

Рисунок 12 – настройка VRRP ведущего устройства

4.2 Настройка резервного устройства

Поднимем виртуальную машину в той же подсети и с теми же параметрами, что и основное устройство. Изменим следующие параметры:

Внешний адрес: **10.0.2.22/24**.

Внутренний адрес: **10.0.0.10/24**.

Создадим и настроим виртуальный маршрутизатор `vr_ext`.

```
set high-availability vrrp group vr_ext
```

```
edit high-availability vrrp group vr_ext
```

```

set interface eth0

set vrid 50

set address 10.0.2.254/24

set priority 50

commit

save

```

Создадим и настроим виртуальный маршрутизатор `vr_int`.

```

set high-availability vrrp group vr_int

edit high-availability vrrp group vr_int

set interface eth1

set vrid 50

set address 10.0.0.254/24

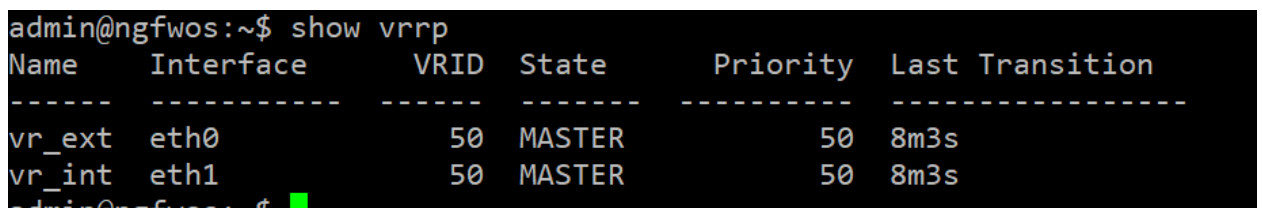
set priority 50

commit

save

```

На рисунке 13 изображены настройки VRRP резервного устройства.



```

admin@ngfwos:~$ show vrrp
Name      Interface  VRID  State    Priority  Last Transition
-----
vr_ext    eth0       50    MASTER   50        8m3s
vr_int    eth1       50    MASTER   50        8m3s

```

Рисунок 13 – настройка VRRP резервного устройства

4.3 Тестирование работы кластера

Для тестирования работоспособности кластера поднимем NGFW1 и NGFW2 и перенастроим шлюз по умолчанию машины RedOS Client на `vr_int` (**10.0.0.254**). Запустим непрерывное обращение к сайту `ident.me` (тест соединения) и `traceroute` до `ya.ru` (Рисунок 14).

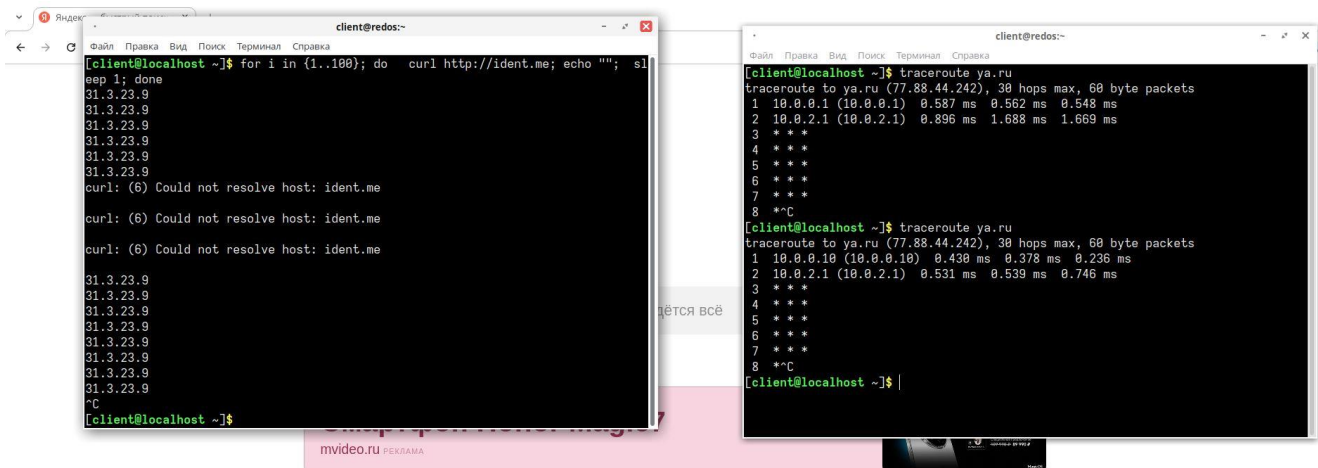


Рисунок 14 – Тестирование работоспособности кластера

Далее отключим основной NGFW, не прекращая обращения к ident.me и запустим traceroute снова. Видим на рисунке 14, что обращения возобновились, однако шлюз сменился на резервный.

5 Настройка сбора статистики по трафику

Для реализации сбора статистики по трафику с использованием технологии NetFlow необходимо реализовать схему, изображённую на рисунке 15.

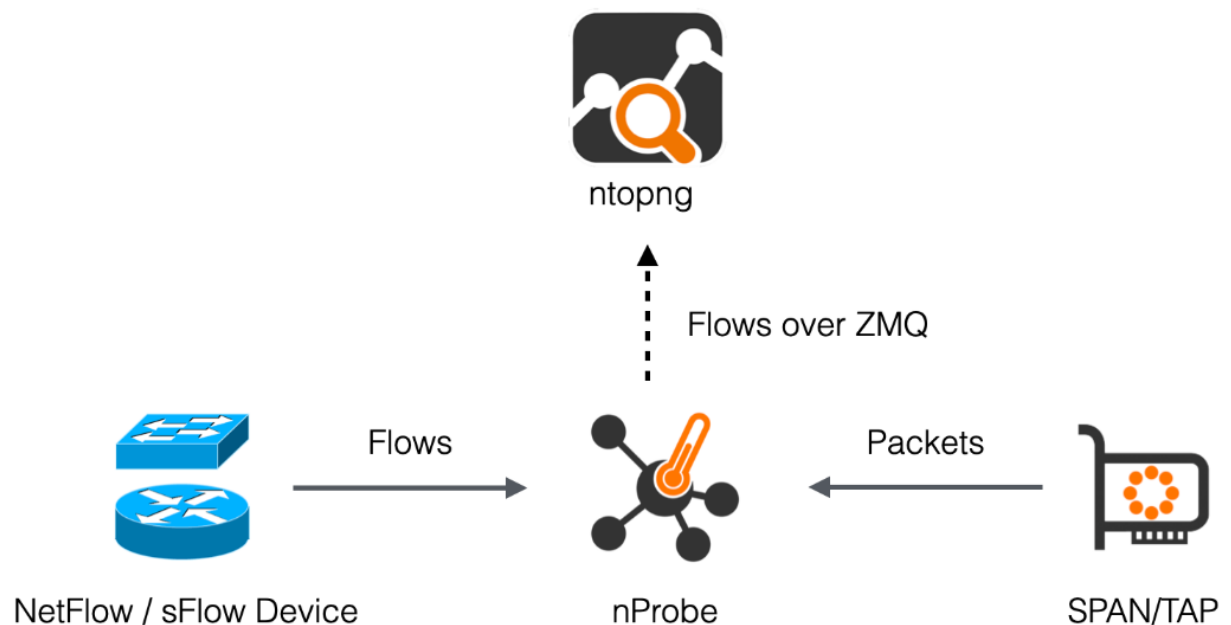


Рисунок 15 – схема для потоков NetFlow

5.1 Настройка Flow-accounting на NGFW

```
set system flow-accounting interface eth0

set system flow-accounting netflow version 9

set system flow-accounting netflow server 10.0.2.6 port 2055

set system flow-accounting netflow source-address 10.0.2.21

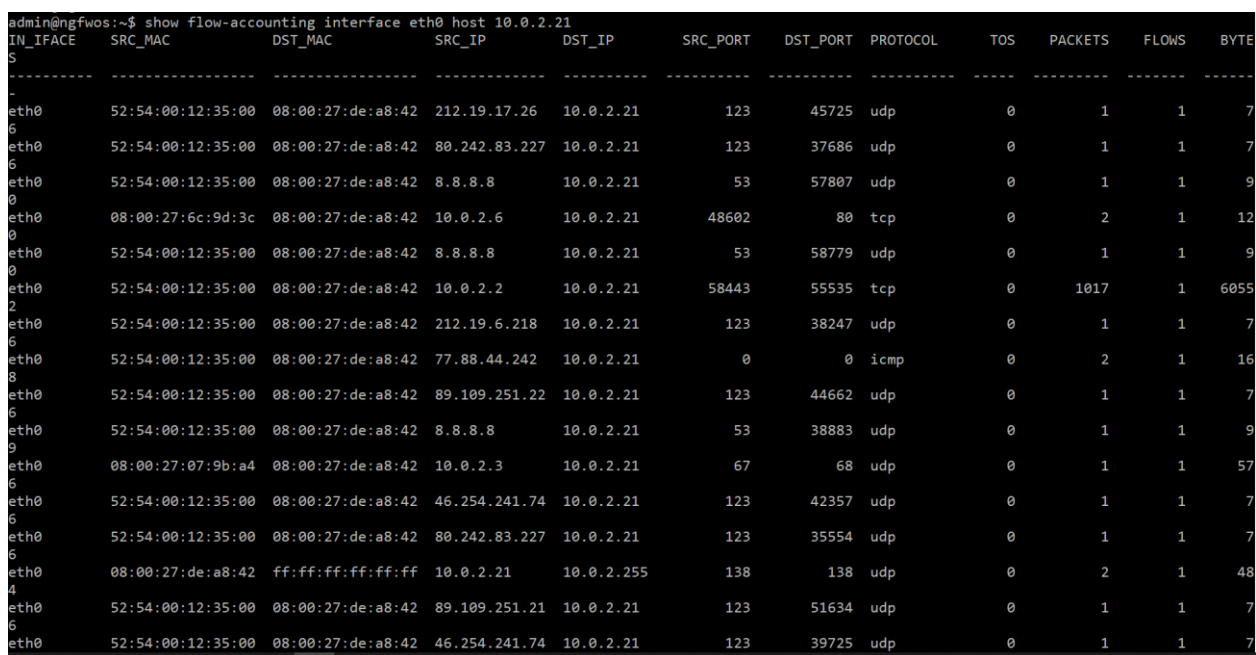
set system flow-accounting netflow engine-id 0

set system flow-accounting netflow sampling-rate 1

set system flow-accounting netflow timeout expiry-interval 30

set system flow-accounting netflow max-flows 8192
```

На рисунке 16 изображена потоки Netflow.



IN_IFACE	SRC_MAC	DST_MAC	SRC_IP	DST_IP	SRC_PORT	DST_PORT	PROTOCOL	TOS	PACKETS	FLOWS	BYTE
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	212.19.17.26	10.0.2.21	123	45725	udp	0	1	1	7
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	80.242.83.227	10.0.2.21	123	37686	udp	0	1	1	7
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	8.8.8.8	10.0.2.21	53	57807	udp	0	1	1	9
eth0	08:00:27:6c:9d:3c	08:00:27:de:a8:42	10.0.2.6	10.0.2.21	48602	80	tcp	0	2	1	12
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	8.8.8.8	10.0.2.21	53	58779	udp	0	1	1	9
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	10.0.2.2	10.0.2.21	58443	55535	tcp	0	1017	1	6055
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	212.19.6.218	10.0.2.21	123	38247	udp	0	1	1	7
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	77.88.44.242	10.0.2.21	0	0	icmp	0	2	1	16
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	89.109.251.22	10.0.2.21	123	44662	udp	0	1	1	7
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	8.8.8.8	10.0.2.21	53	38883	udp	0	1	1	9
eth0	08:00:27:07:9b:a4	08:00:27:de:a8:42	10.0.2.3	10.0.2.21	67	68	udp	0	1	1	57
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	46.254.241.74	10.0.2.21	123	42357	udp	0	1	1	7
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	80.242.83.227	10.0.2.21	123	35554	udp	0	1	1	7
eth0	08:00:27:de:a8:42	ff:ff:ff:ff:ff:ff	10.0.2.21	10.0.2.255	138	138	udp	0	2	1	48
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	89.109.251.21	10.0.2.21	123	51634	udp	0	1	1	7
eth0	52:54:00:12:35:00	08:00:27:de:a8:42	46.254.241.74	10.0.2.21	123	39725	udp	0	1	1	7

Рисунок 16 – схема для потоков NetFlow

5.2 Настройка NetFlow-коллектора nprobe.

Для начала импортируем репозиторий для установки nprobe и ntopng.

```
apt-get install software-properties-common wget

add-apt-repository universe

wget https://packages.ntop.org/apt/22.04/all/apt-ntop.deb
```

```
apt install ./apt-ntop.deb
```

Обновим apt и установим софт

```
apt-get update
```

```
apt-get install pfring-dkms nprobe ntopng n2disk cento ntap
```

Далее в файле конфигурации /etc/nprobe/nprobe.conf пропишем

```
--ntopng="tcp://127.0.0.1:5556"
```

```
-T="@NTOPNG@"
```

```
-i=none
```

```
-n=none
```

```
--collector-port=2055
```

```
--verbose
```

И запустим сервис:

```
systemctl start nprobe
```

5.3 Настройка NetFlow-монитора ntopng.

В файле конфигурации /etc/ntopng/ntopng.conf пропишем:

```
-i="tcp://127.0.0.1:5556"
```

```
--local-networks="10.0.2.0/24"
```

```
-w=3456
```

И запустим сервис:

```
systemctl start ntopng
```

5.4 Результат работы.

Настроим в Settings -> Preferences отображение NetFlow. В качестве исследуемого трафика выберем VRRP. Сбор статистики приведены на рисунках 18 и 19.

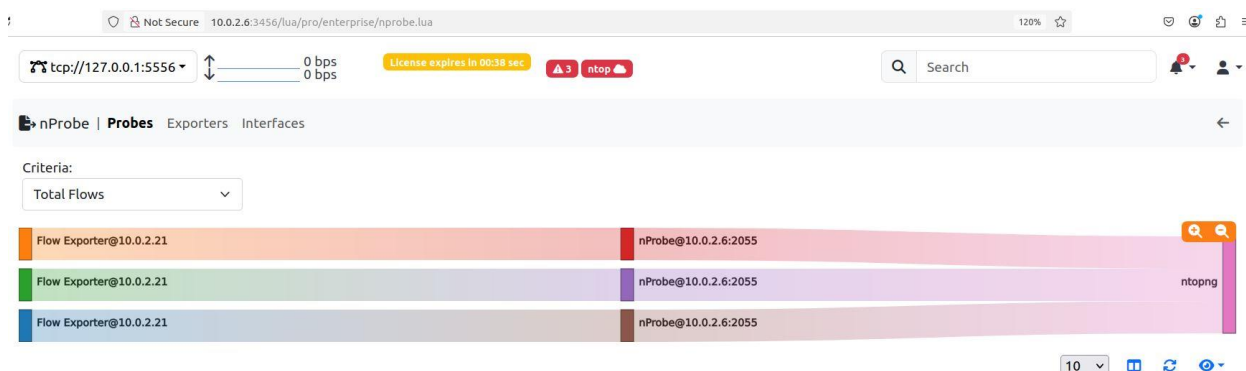


Рисунок 18 – изображение потоков NetFlow

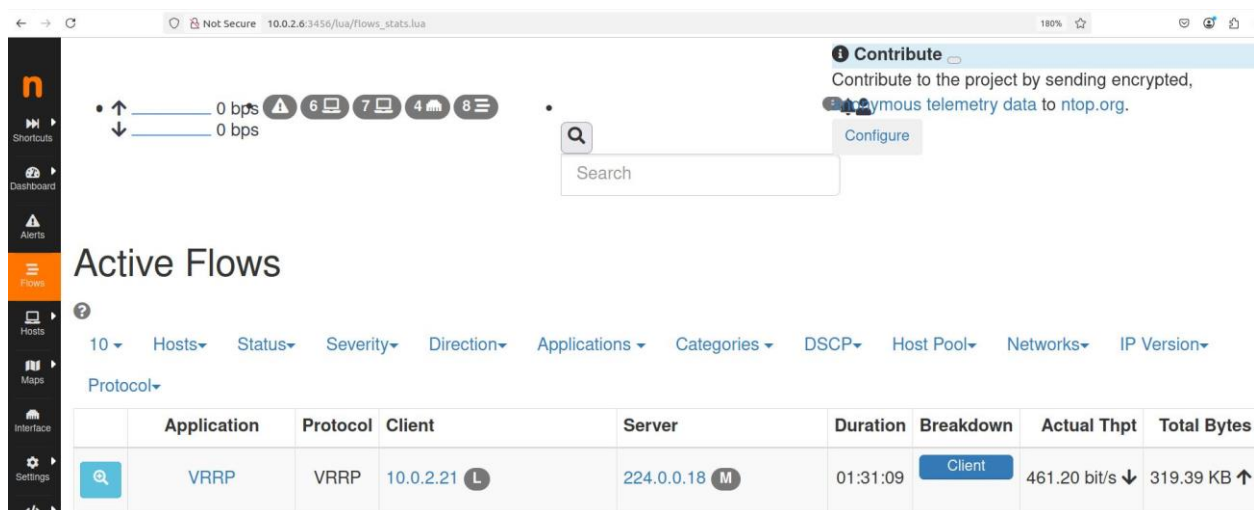


Рисунок 19 – Мониторинг VRRP

Часть 2

1 Установка RedOS Serv и шифрование диска

Для установки сервера выберем опцию минимальный сервер. В устройствах включим шифрование дисков. Настройка, изображённая на рисунке 20 создаёт зашифрованный LUKS-раздел.

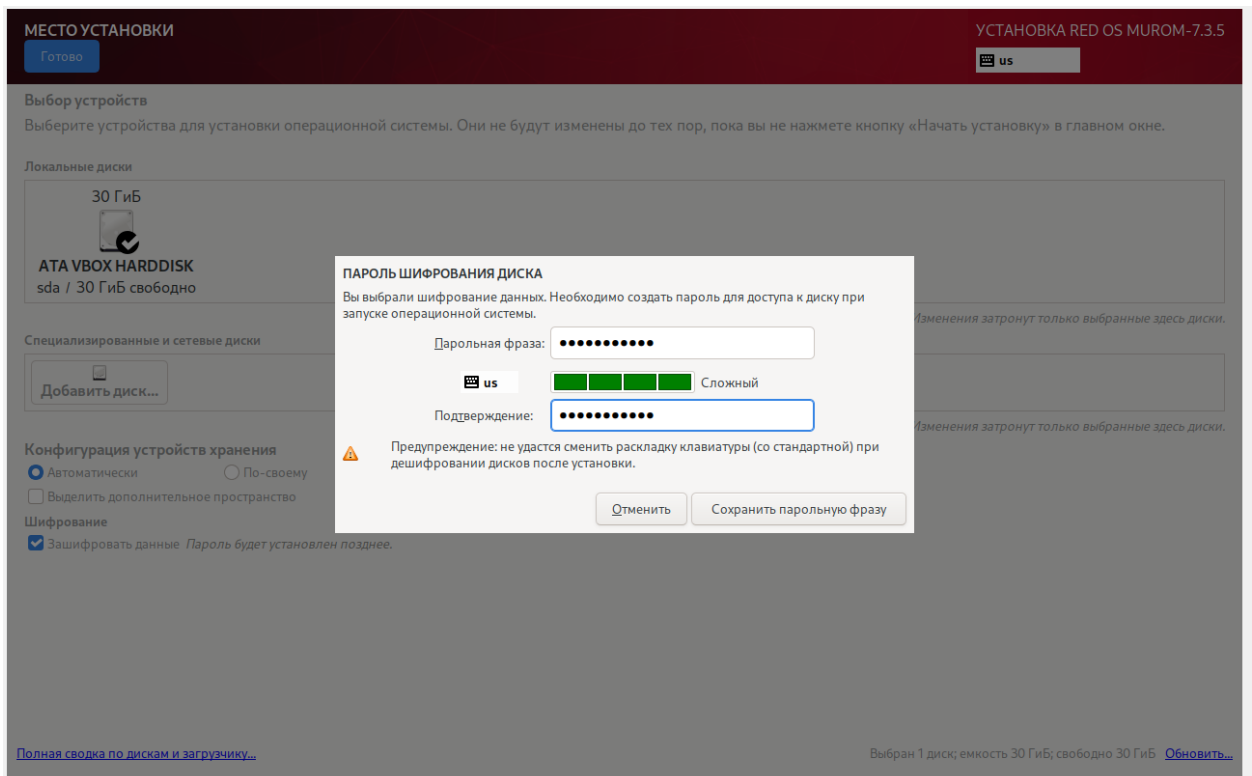


Рисунок 20 – Шифрование данных

Перед загрузкой системы будет запрашиваться пароль для доступа к разделу (Рисунок 21). Протестируем защиту, добавив vdi-диск к другой машине RedOS и примонтировав раздел в систему. В этом случае пароль также запрашивается.

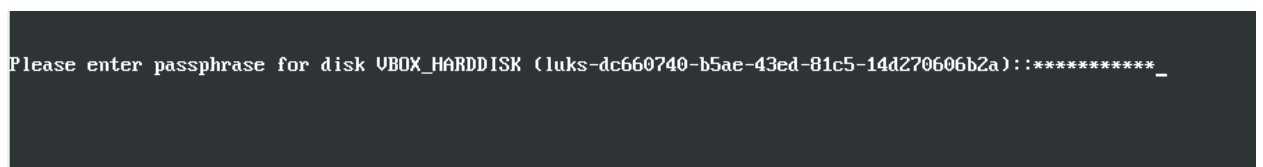


Рисунок 21 – Запрос пароля

2 Результат работы контейнеров docker-compose

На рисунке 22 видно, что контейнеры подняты, все находятся в состоянии healthy.

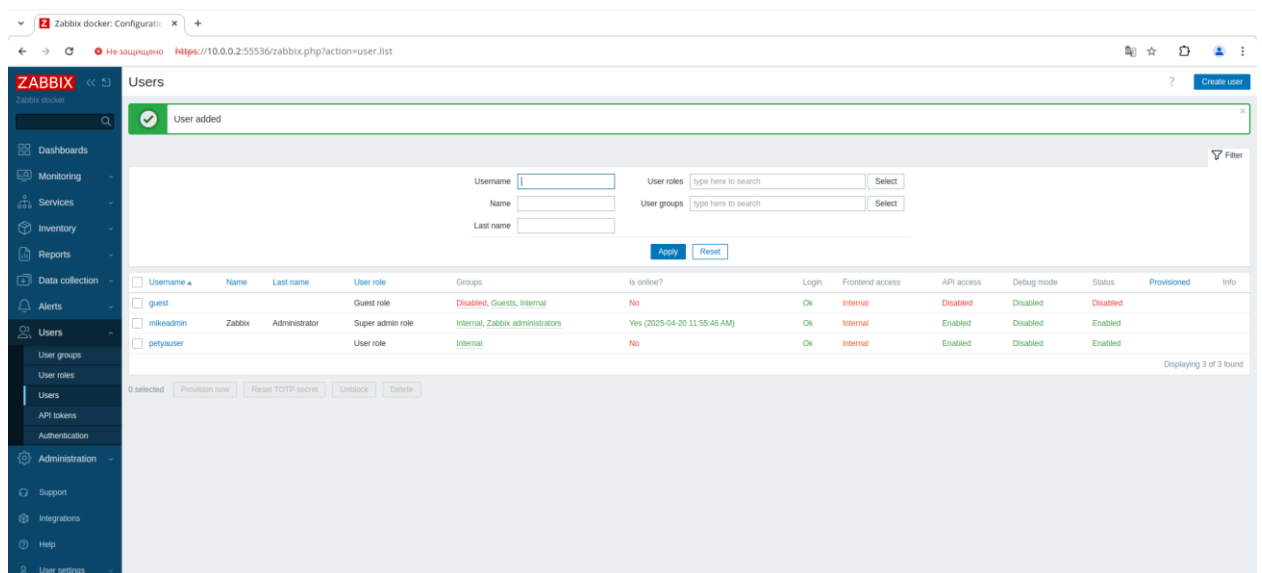
```
[root@localhost ITPlanet]# docker-compose ps -a
```

NAME	IMAGE	COMMAND	SERVICE	CREATED
STATUS	PORTS			
itplanet-django-1	itplanet-django	"python manage.py ru..."	django	About a minute ago
p About a minute (healthy)				
itplanet-nginx-1	nginx:1.27	"/docker-entrypoint..."	nginx	About a minute ago
p About a minute (healthy)	80/tcp, 0.0.0.0:55536->55536-55538/tcp, :::55536-55538->55536-55538/tcp			
itplanet-pgadmin-1	dpape/pgadmin4:9.0	"/entrypoint.sh"	pgadmin	About a minute ago
p About a minute (healthy)				
itplanet-postgres_django-1	postgres:17	"docker-entrypoint.s..."	postgres_django	About a minute ago
p About a minute (healthy)				
itplanet-postgres_zabbix-1	postgres:17	"docker-entrypoint.s..."	postgres_zabbix	About a minute ago
p About a minute (healthy)				
itplanet-zabbix-agent-1	zabbix/zabbix-agent:alpine-7.0-latest	"/usr/bin/docker-ent..."	zabbix-agent	About a minute ago
p About a minute (healthy)				
itplanet-zabbix-dashboard-1	zabbix/zabbix-web-nginx-pgsql:alpine-7.0-latest	"docker-entrypoint.sh"	zabbix-dashboard	About a minute ago
p About a minute (healthy)				
itplanet-zabbix-server-1	zabbix/zabbix-server-pgsql:alpine-7.0-latest	"/usr/bin/docker-ent..."	zabbix-server	About a minute ago
p About a minute (healthy)				

Рисунок 22 – Работа контейнеров

3 Результат работы Zabbix-dashboard

Для большей безопасности сменим имя и пароль дефолтной учётной записи, а также создадим пользовательскую. На рисунке 23 видны настройки этих учётных записей.



Username	Name	Last name	User role	Groups	Is online?	Login	Frontend access	API access	Debug mode	Status	Provisioned	Info
guest			Guest role	Disabled, Guests, Internal	No	Ok	Internal	Disabled	Disabled	Disabled		
mikadmin	Zabbix	Administrator	Super admin role	Internal, Zabbix administrators	Yes (2025-04-20 11:55:46 AM)	Ok	Internal	Enabled	Disabled	Enabled		
petyauser			User role	Internal	No	Ok	Internal	Enabled	Disabled	Enabled		

Рисунок 23 – Настройки учётных записей

Подключение Zabbix-агента изображено на рисунке 24.

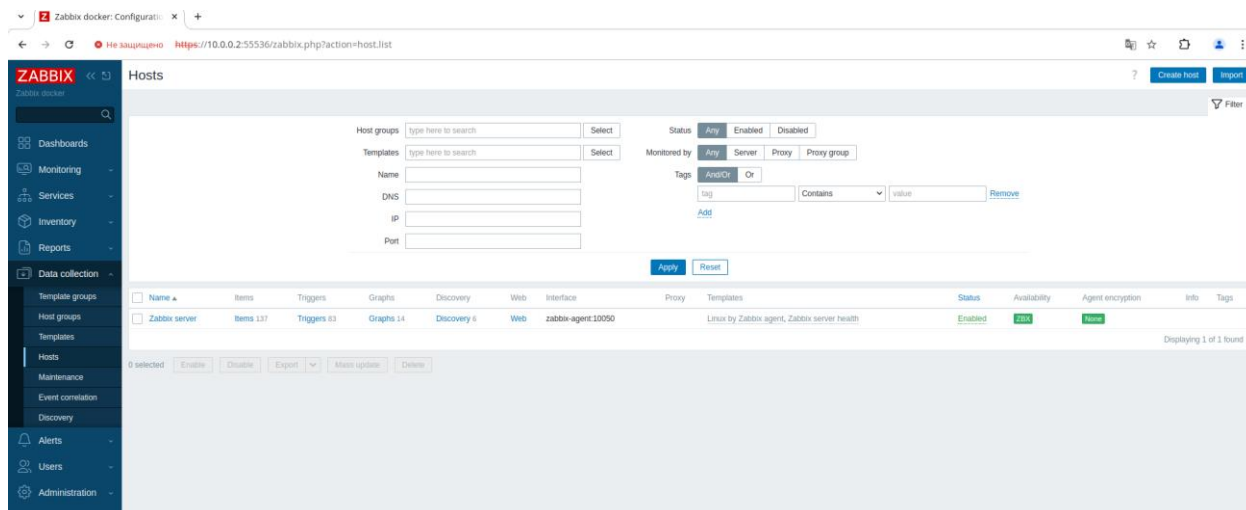


Рисунок 24 – Работа Zabbix-агента

4 Результат работы Django.

На рисунке 25 видим работу Django.

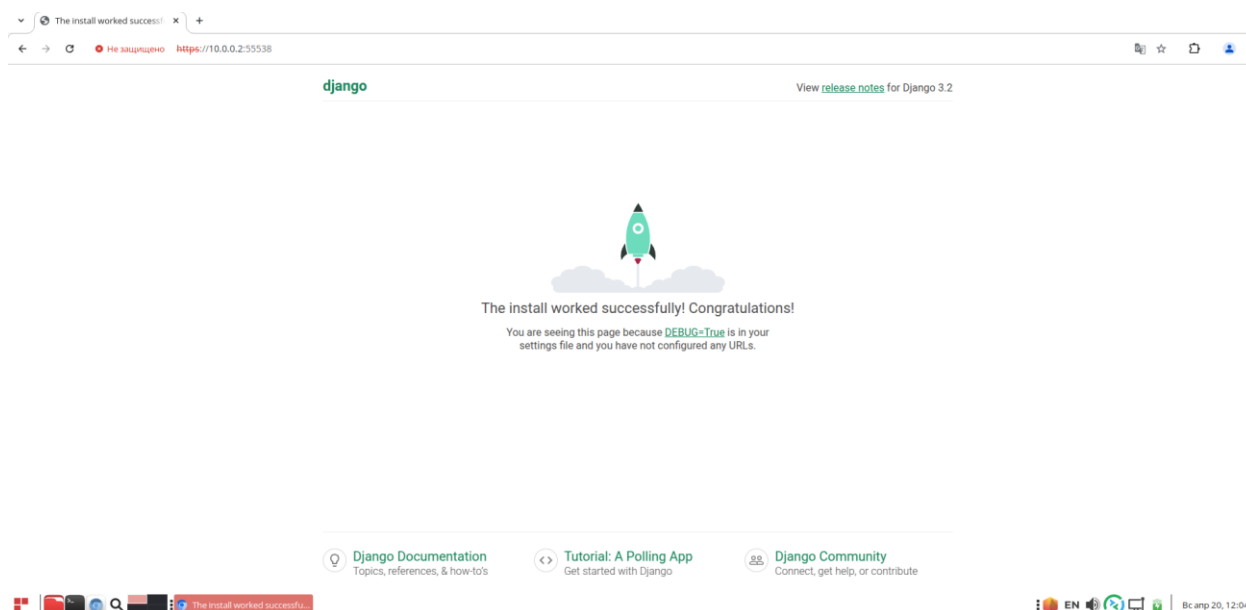


Рисунок 25 – Работа Django

Применим миграцию к БД и создадим суперпользователя с именем mikeadmin

```
docker-compose exec django bash
```

```
python manage.py migrate
```

```
python manage.py createsuperuser
```

На рисунке 26 изображён его профиль

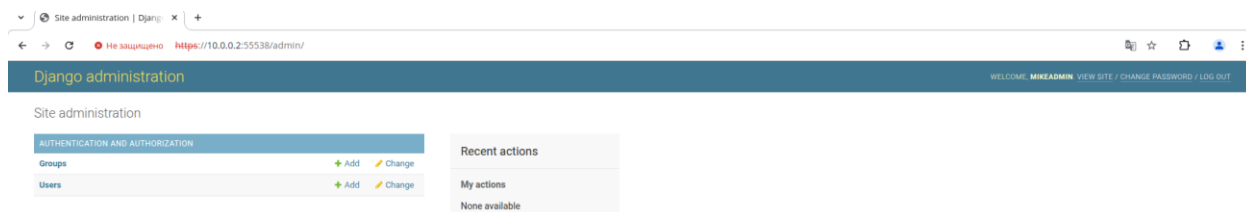


Рисунок 26 – Профиль суперпользователя.

5 Результат работы pgAdmin и подключения Django и Zabbix.

На рисунке 27 изображена структура БД djangodb. На рисунке 28 виден запрос на извлечение всех пользователей Django, где записан созданный суперпользователь mikeadmin. Таким образом, подключение Django к БД протестировано.

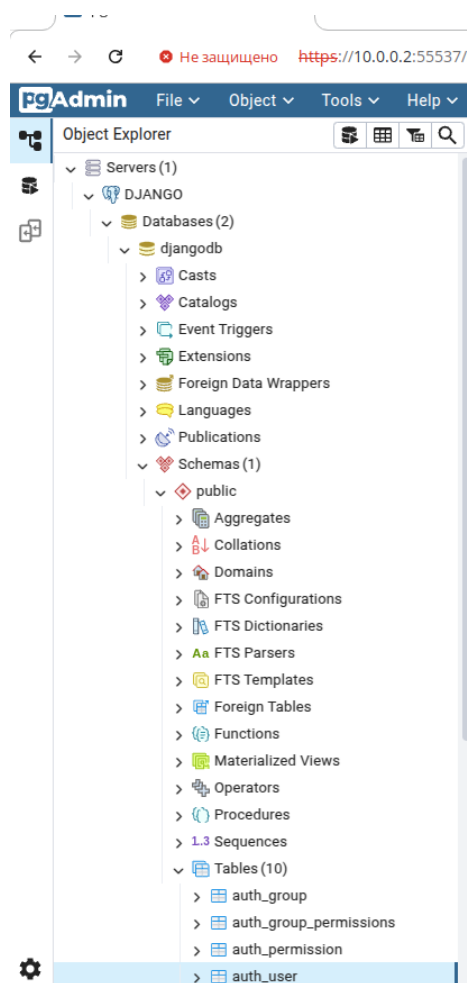


Рисунок 27 – структура БД djangodb

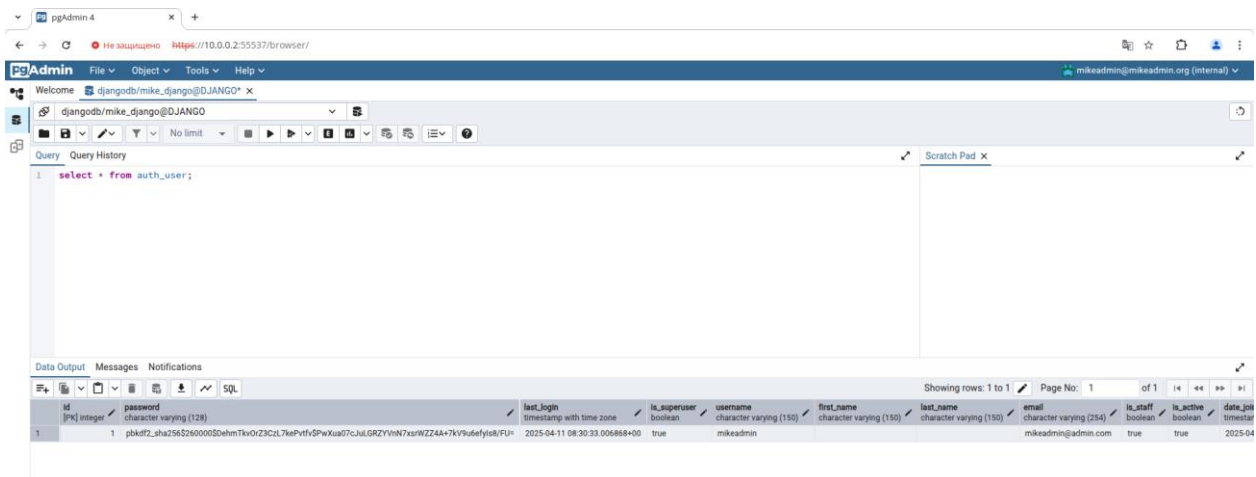


Рисунок 28 – Запись mikeadmin в таблице auth_user

На рисунке 29 изображена структура БД zabbixdb. На рисунке 30 виден запрос к этой БД.

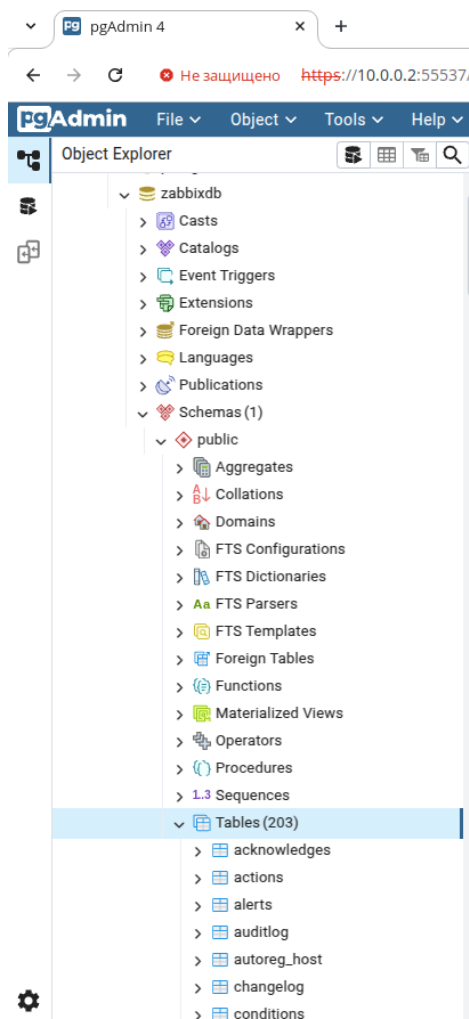


Рисунок 29 – структура БД zabbixdb

The screenshot shows the pgAdmin 4 web interface. A query `select * from dashboard;` has been executed, resulting in a table with 10 rows. The table columns are: `dashboardid` (PK, bigint), `name` (character varying (255)), `userid` (bigint), `private` (integer), `templateid` (bigint), `display_period` (integer), `auto_start` (integer), and `uuid` (character varying (32)).

	dashboardid [PK] bigint	name character varying (255)	userid bigint	private integer	templateid bigint	display_period integer	auto_start integer	uuid character varying (32)
1	1	Global view	1	0	[null]	30	1	
2	2	Zabbix server health	1	1	[null]	30	1	
3	3	Apache performance	[null]	1	10265	30	1	a328c9e713424465a8e1adec7322b0...
4	4	Apache performance	[null]	1	10264	30	1	c27c68fb9c234a09b4023076b45affc1
5	5	Docker overview	[null]	1	10318	30	1	7eb6472d07ac4c379e6b730b59a125...
6	7	Redis performance	[null]	1	10310	30	1	ee4c29eb7a0f443fafb7e7d3b9df7b24
7	8	RabbitMQ overview	[null]	1	10300	30	1	a886a871a00d47e28b8648cadae3bc...
8	9	RabbitMQ node status	[null]	1	10301	30	1	ce0af043ed2c4e7c988674c9ecb787d6
9	10	Network interfaces	[null]	1	10285	30	1	19dac6b780aa49558bf4a3782ba4b3...

Total rows: 342 Query complete 00:00:00.299

Рисунок 30 – Содержимое таблицы dashboard

6 Тестирование создания и восстановления бэкапов.

С помощью скриптов **backup_script.sh** и **encrypt_script.sh** выполняется gpg-шифрование и резервное копирование volumes с БД (Рисунок 31) в директорию **/home/serv/backups/**, а также на сервер **10.0.0.3** в директорию **/home/client/backups/** по ssh. Для этого сгенерируем ключи для беспарольного подключения по ssh.

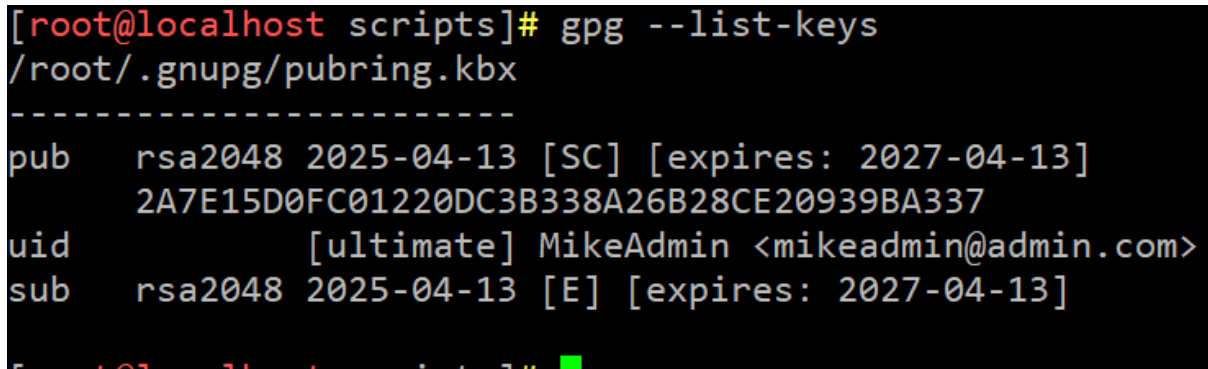
```
local    itplanet_django-postgres
local    itplanet_pgadmin
local    itplanet_zabbix-postgres
```

Рисунок 31 – Volumes для бэкапов

Настроим ключи gpg

```
gpg --full-generate-key
```

На рисунке 32 изображён ключ для шифрования бэкапов.



```
[root@localhost scripts]# gpg --list-keys
/root/.gnupg/pubring.kbx
-----
pub      rsa2048 2025-04-13 [SC] [expires: 2027-04-13]
          2A7E15D0FC01220DC3B338A26B28CE20939BA337
uid            [ultimate] MikeAdmin <mikeadmin@admin.com>
sub      rsa2048 2025-04-13 [E] [expires: 2027-04-13]
```

Рисунок 32 – Сгенерированный ключ

На случай отказа docker-сервера экспортируем публичный и приватный ключ на сервер 10.0.0.3

```
gpg --export-secret-key -a " MikeAdmin" > private.key
```

```
gpg --export -a " MikeAdmin" > public.key
```

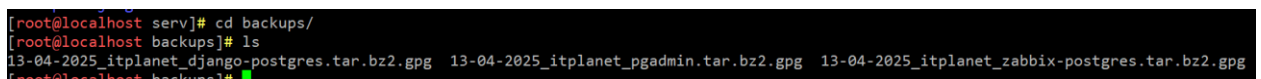
Добавим выполнение скрипта backup_script.sh в crontab:

```
crontab -e
```

```
0 4 * * * /root/scripts/backup_script.sh
```

Создание, шифрование и сохранение бэкапов будет выполняться каждый день в 04:00.

На рисунках 33 и 34 изображены директории с шифрованными бэкапами на docker-сервере и на удалённом сервере 10.0.0.3 соответственно.



```
[root@localhost serv]# cd backups/
[root@localhost backups]# ls
13-04-2025_itplanet_django-postgres.tar.bz2.gpg  13-04-2025_itplanet_pgadmin.tar.bz2.gpg  13-04-2025_itplanet_zabbix-postgres.tar.bz2.gpg
```

Рисунок 33 – Бэкапы на RedOS Serv

```
client@redos:~/backups
Файл Правка Вид Поиск Терминал Справка
[client@localhost ~]$ cd /home/client/
[client@localhost ~]$ ls
backups index.html ovpn Видео Документы Загрузки Изображения Музыка Общедоступные 'Рабочий стол' Шаблоны
[client@localhost ~]$ cd backups/
[client@localhost backups]$ ls
13-04-2025_itplanet_django-postgres.tar.bz2.gpg 13-04-2025_itplanet_pgadmin.tar.bz2.gpg 13-04-2025_itplanet_zabbix-postgres.tar.bz2.gpg
[client@localhost backups]$
```

Рисунок 34 – Бэкапы на RedOS Client

Восстановим для теста бэкап БД djangodb. Для этого имитируем удаление volumes командой

```
docker-compose down -v
```

Расшифруем соответствующий volume (рисунок 35).

```
[root@localhost backups]# gpg -d 13-04-2025_itplanet_django-postgres.tar.bz2.gpg > 13-04-2025_itplanet_django-postgres.tar.bz2
gpg: encrypted with 2048-bit RSA key, ID 81D177E9E1B5EFB6, created 2025-04-13
"MikeAdmin <mikeadmin@admin.com>"
[root@localhost backups]# ls
13-04-2025_itplanet_django-postgres.tar.bz2 13-04-2025_itplanet_pgadmin.tar.bz2.gpg
13-04-2025_itplanet_django-postgres.tar.bz2.gpg 13-04-2025_itplanet_zabbix-postgres.tar.bz2.gpg
```

Рисунок 35 – Расшифровка volume

Далее с помощью команды ниже примонтируем бекапный volume (рисунок 35).

```
docker run --rm -v itplanet_django-postgres:/volume -v /home/serv/backups:/backup alpine sh -c "rm -rf /volume/* /volume/..?* /volume/.[!..]* ; tar -C /volume/ -xjf /backup/13-04-2025_itplanet_django-postgres.tar.bz2"
```

```
[root@localhost backups]# docker run --rm -v itplanet_django-postgres:/volume -v /home/serv/backups:/backup alpine sh -c "rm -rf /volume/* /volume/..?* /volume/.[!..]* ; tar -C /volume/ -xjf /backup/13-04-2025_itplanet_django-postgres.tar.bz2"
[root@localhost backups]# docker volume ls
```

Рисунок 35 – Восстановление бэкапа

Теперь БД djangodb приведена к виду как на рисунке 27 и 28.