Оглавление

[Введение 2](#_Toc189475002)

[1. О роли администратора. 2](#_Toc189475003)

[2. Создание и настройка VPN-сервера 5](#_Toc189475004)

[3. Развёртывание удостоверяющего центра для выдачи сертификатов 8](#_Toc189475005)

[4. Настройка мониторинга 11](#_Toc189475006)

[5. Создание системы резервного копирования 15](#_Toc189475007)

[6. План развития инфраструктуры 17](#_Toc189475008)

[Заключение 19](#_Toc189475009)

# Введение

В данном проекте будет разработана необходимая и достаточная для работы компании инфраструктура VPN сервера. Были внедрены решения, позволяющие оптимизировать и автоматизировать большую часть процессов необходимых для ее работы, настроена система мониторинга и система резервного копирования данных.

Обзорная модель инфраструктуры представлена на рис. 1. На нем видно, что вся инфраструктура представлена в виде 4х серверов работающих в облаке: VPN Server – сервер, с установленным vpn компонентом openvpn; Monitoring – сервер, с установленным на него инструментом мониторинга Prometheus и системой оповещения alertmanager; Certification Authority (CA) – сервер с установленным на него компонентом easy-rsa, предназначенным для работы с инфраструктурой ключей; Remote repository – сервер , на котором записываются резервные копии важных компонентов данной архитектуры, на него установлен компонент aptly, для создания локальной репозитории.

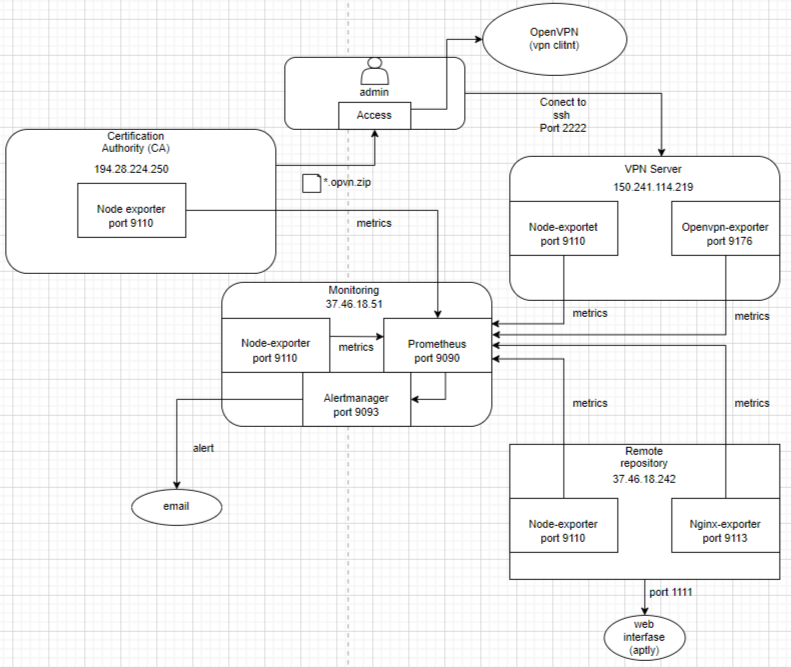


Рисунок 1 – инфраструктура, нарисованная в <https://app.diagrams.net/>

В роли хостинга, предоставляющего услуги, аренды серверов был выбран dshp. Выбор был сделан по причинам сравнительно низкой цены за возможную аренду зарубежного сервера - 200 руб. в месяц независимо от количества используемого трафика, что целесообразно, если использовать vpn относительно часто. В рамках пакета, хостинг предоставляет публичный зарубежный IP, собственную систему мониторинга в своем web интерфейсе, возможность подключиться к серверу напрямую через vps, возможность переустановки ОС и многое другое.

Заранее укажу, что все упомянутые в проекте артефакты будут приведены в общей папке проекта, чтобы не повторяться в дальнейшем.

# О роли администратора.

В ходе составления плана инфраструктуры было принято решения, что на всех серверах необходимо выделить роль “admin”. Изначально планировалось оптимизировать все процессы, чтобы сам пользователь заходил на сервер и запуском простой команды инициировал процесс создания своего сертификата для openvpn клиента, после чего сертификат пришел бы ему на почту. Но данных подход, к сожалению, имеет несколько проблем. Во-первых, это не безопасно, т.к. тогда каждому пользователю нужно давать доступ по сертификату, это приведет к тому, что нужно передавать открытый ключ во много «лишних рук». Во-вторых, для реализации такого подхода необходимо редактировать несколько скриптов, включенных в пакет (например, easy-rsa), чтобы тот не требовал пастфразы, где они использованы. Это конечно можно сделать, но это опять-таки противоречит безопасности и к тому же трудоемко. Гораздо проще сделать роль администратора.

Данная роль включает в себя: коммуникацию между пользователем и сервером, запуск необходимых скриптов, работа по устранению алертов, переустановка сервера из бэкапа в случаи полной недоступности. ВАЖНО!!! Для того, чтобы ограничить «злоупотребление полномочиями», на всех серверах введено логирование по sudo.

Администратору предоставляется доступ в облако, где он должен будет добавить текст своего публичного ключа, полученного с помощью команды, в раздел «SSH-ключи»:

ssh-keygen -t rsa -b 4096

После чего, Админ сможет подключиться к серверам напрямую по ssl сертификату. Это необходимо сделать, т.к. аутентификация по root и паролю отключена. Несмотря на это, администратору будет предоставлен пароль, который он получится по почте «transport1234», но как только он подключится, его необходимо поменять по команде (Эта информация указана у него в инструкции, которая лежит на сервере):

sudo passwd admin

Рассмотрим пример работы администратора в инфраструктуре. Ему поступает запрос по почте от пользователя, который содержит login пользователя и с информацией, которая подтвердит наличие сотрудника в штате. После проверок, что сотрудник действительно работает, админ, должен залогиниться на vpn сервер:

ssh admin@150.241.114.219 -p 222

На сервере в каталоге ~/Инструкции находится инструкция «Использование», необходимо с ней ознакомиться:

cat ~/Инструкции/Использование

После чего, администратором будет запущен скрипт, ./start.sh, который сделает запрос на подпись пользователя user и его открытый ключ, после чего, они будут отправлены на сервер удостоверяющего центра (ca) 194.28.224.250: полученные артефакты «user.req и user.key.

Далее необходимо перейти на сервер ca и, после ознакомления с инструкцией по работе (также в ~/Инструкции/Использования), уже там запустить скрипт ./start.sh, который сделает сертифика user.opvn и оформит его в zip архив, запросив при этом пароль у администратора. Данный zip архив можно скачать с помощью:

scp -P 222 admin@194.28.224.250:/home/admin/.clients/files/user.ovpn.zip C:\(path)

После чего, передать в запароленном виде пользователю, уточнив какая у него операционная система и направив инструкцию по настройке для пользователя.

На всех серверах, есть инструкции по обработке алертов и минимальные вводные по серверу и как его использовать. Лучше зайти изначально до работы на все 4 сервера, и ознакомившись с инструкциями (cat ~/Инструкции), поменять пароль.

На сервере monitoring <http://37.46.18.51:9090/alerts> (веб версия) помимо этого необходимо быть залогиненным, желательно 24/7, чтобы вовремя среагировать на ошибку. Но также, там действует система оповещения алертменеджер <http://37.46.18.51:9093/#/alerts> , которая будет присылать на почту уведомления о неисправности. Для успешной настройки алертменеджера необходимо сделать команды:

sudo sed -i 's/test@company.net/Ваша ПОЧТА/g' /etc/prometheus/ alertmanager.yml

sudo sed -i 's/test\_passwd/ВАШ ПАРОЛЬ/g' /etc/prometheus/alertmanager.yml

Данные команды заменят почту и ее пароль из конфигурационного файла, на то, что будет использована вами.

Важно, если вы используете в качестве почтового клиента не яндекс, то нужно будет поменять еще одну строчку в конфигурационном файле:

sudo vim /etc/prometheus/alertmanager.yml

smarthost: 'smtp.yandex.ru:587' заменить, например, на smarthost: 'smtp.gmail.com:587'.

Касаемо последнего сервера: роль админ, здесь подразумевает проверку, того, что произведены успешные бэкапы в заданный день, и в случае неработоспособности сервера восстановление его через deb пакеты.

Далее более подробно рассмотрим каждый из модулей инфраструктуры, и начнем с самого vpn сервера.

# Создание и настройка VPN-сервера

На этом сервере 150.241.114.219 установлена служба openvpn, которая запускается при наличии серверного сертификата server.crt, приватного ключа server.key, правильно составленного конфигурационного файла server.conf, сертификата удостоверяющего центра ca.crt и tls файла ta.key. Все эти файлы должны быть перенесены в директорию /etc/openvpn/server.

Подключение по ssh настроено таким образом, что к серверу невозможно подключиться пользователю root; невозможно подключиться с использованием пароля, только при наличии открытого ключа на устройстве; нельзя подключиться по пустому паролю; нельзя подключиться, используя ipv6; изменен порт на 222. Данные модификации сделаны исключительно для безопасности и для защиты сервера от несанкционированного доступа и дудос атак. Настройки файла /etc/ssh/sshd\_config предоставлены в архиве проекта. Кроме того, в файл «*/etc/default/grub* внесены строки, ограничивающие подключение только по протоколу ipv4:

GRUB\_CMDLINE\_LINUX\_DEFAULT="quiet splash ipv6.disable=1"  
GRUB\_CMDLINE\_LINUX="net.ifnames=0 biosdevname=0 ipv6.disable=1"

Также настроено время с помощью sudo timedatectl set-timezone Europe/Moscow.

Чтобы в дальнейшем не повторяться, данные настройки sshd\_config, «/etc/default/grub и времени были применены на всех серверах в этом проекте.

На сервере установлен фаервол iptables и настроены политики в соответствии с файлом iptables.sh, который также предоставлен в проекте.

Перейдем к сути, на этом сервере 2 основных скрипт ~/start.sh и ~/backup.sh .

При запуске start.sh происходит создание приватного ключа и запроса на подпись, как это продемонстрировано на рис. 3. После чего происходит отправление файлов на сервер удостоверяющего центра для подписания реквеста roma.req, и дальнейшую генерацию сертификата roma.ovpn. Отмечу, что все действия происходят без необходимости ввода пароля, за счет ssl ключей, которые хранятся в /home/admin/.ssh. На всех остальных серверах также установлены пары приватный/публичный ключ, поэтому утилита scp не требует пароля.

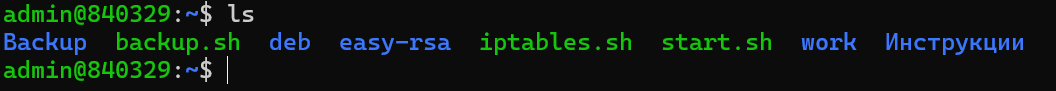


Рисунок 2 – домашний каталог пользователя admin

Еще на сервере установлен планировщик задач cron, таким образом, что раз в неделю происходит сканирования компьютера rkhanter , обновление пакетов в репозитории и раз в месяц 15го числа, создание бэкапа (аналогичная ситуация и на других серверах).

Теперь проверим скрипт производящий бэкап. На рис. 4 видно, что скрипт backup.sh завершился без ошибок и был направлен на удаленную репозиторию для хранения

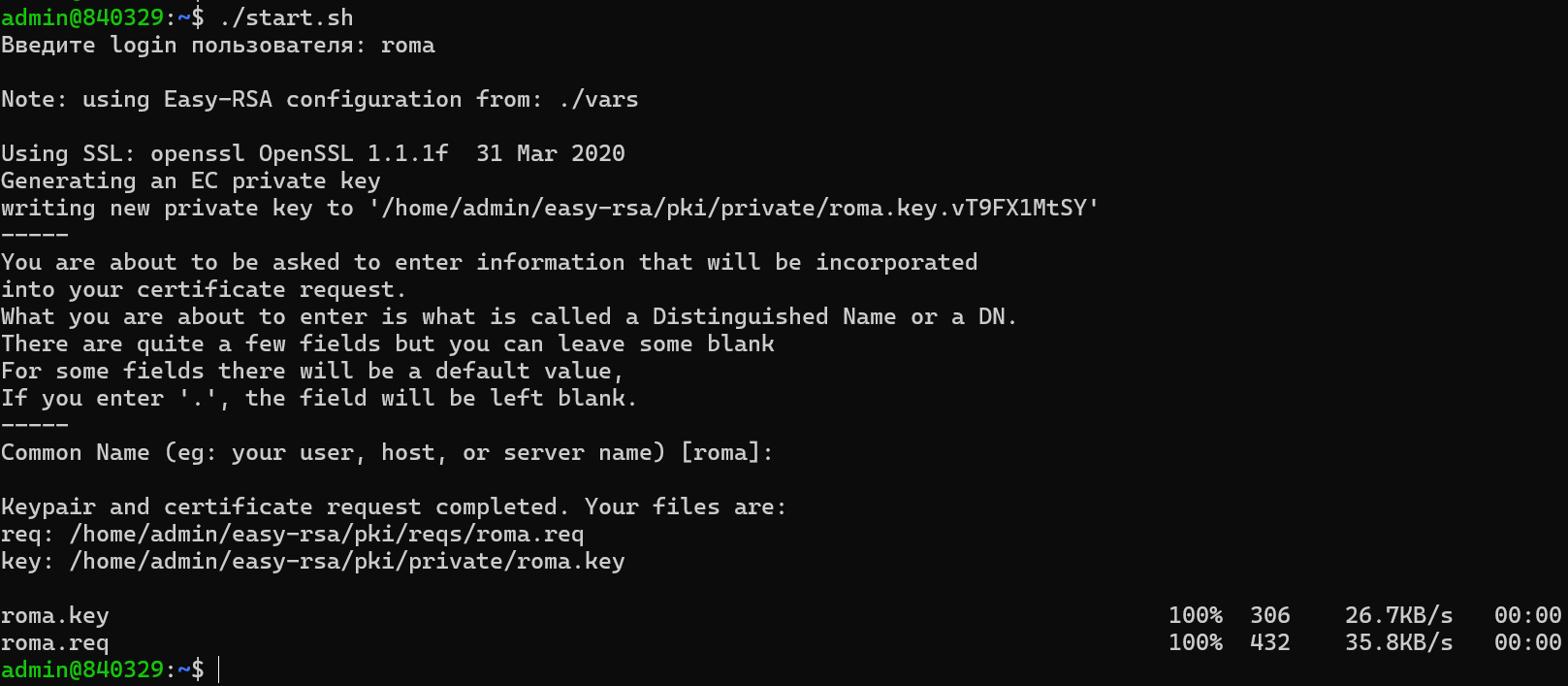


Рисунок 3 – Пример запуска скрипта start.sh для пользователя roma

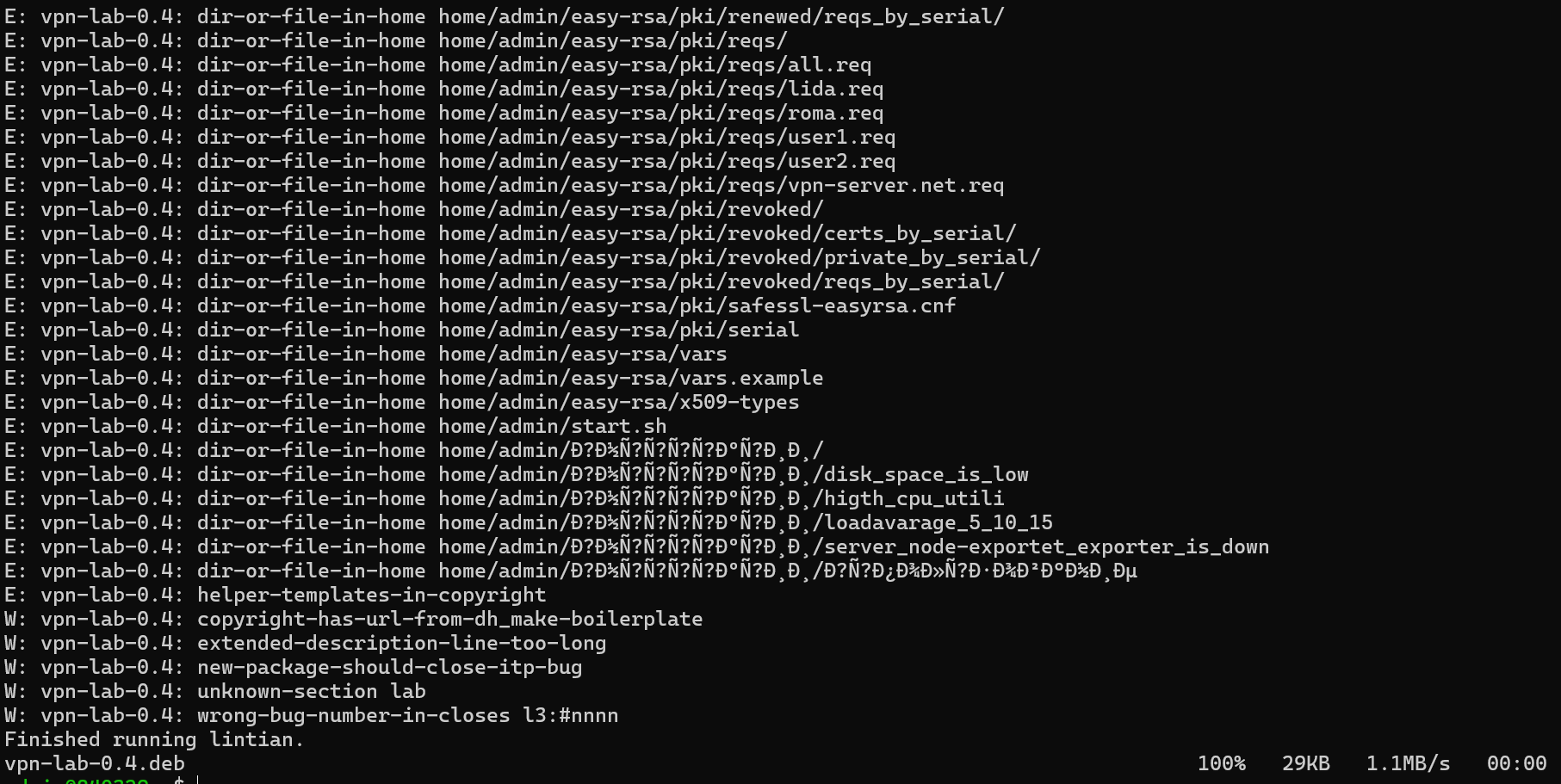


Рисунок 4 – Пример запуска backup.sh

Забегая вперед, скажу, что roma.req был успешно подписан на CA, и с помощью всех других артефактов, был сгенерирован roma.ovpn. Давайте убедимся в его работоспособности. На рис. 5 видно, что сервер запущен и изменил IP адрес, и теперь он соответствует тому, который нам предоставляет хостинг, а значит все работает, как надо.

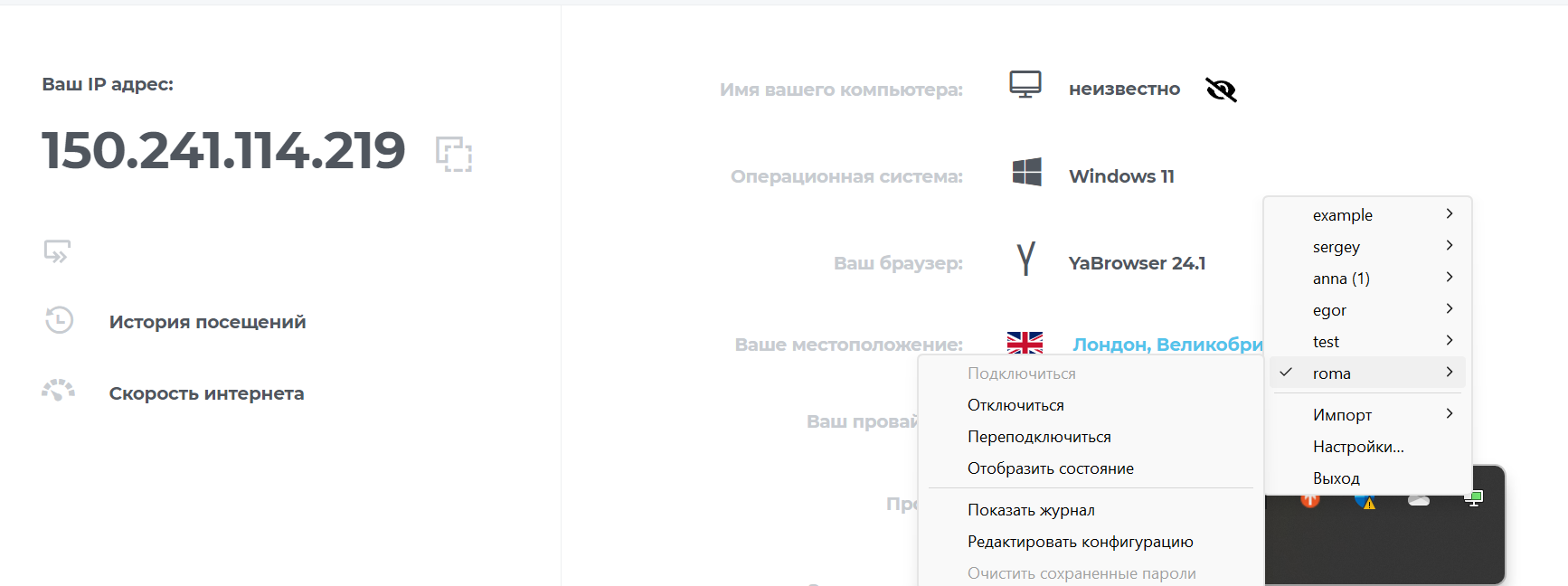


Рисунок 5 – Проверка работоспособности сертификата

# Развёртывание удостоверяющего центра для выдачи сертификатов

На данном сервере 194.28.224.250 располагается компонент, обеспечивающий работу с инфраструктурой открытых ключей easy-rsa.

Также как и на vpn сервере работают 2 скрипта: start.sh и backup.sh. Первый настроен таким образом, что в случае запуска, спрашивает у администратора логин пользователя и ищет ключ и реквест в подкаталогах easy-rsa. После чего, происходит подпись реквеста и генерация сертификата клиента. В этом процессе задействован корневой сертификат удостоверяющего центра ca.crt, расположенный по адресу ~/easy-rsa/pki. Потребуется пастфраза, которая была указана при создании ca.crt, ее необходимо передать администратору, так, чтобы никто другой не узнал (например, с помощью keepas). Важно отметить, что данный файл является ключевым компонентом сервера СА, поэтому его потеря приведет к перенастройке сервера и, что хуже, перевыпуску всех имеющихся сертификатов.

Далее, с помощью скрипта ~/.clients/make\_config.sh создает сертификат \*.ovpn и архивирование его в zip.



Рисунок 4 -домашняя директория пользователя

На сервере также настроен планировщик задач cron, таким образом, что раз в неделю происходит сканирования компьютера rkhanter, обновление пакетов в репозитории и раз в месяц 15го числа, создание бэкапа.

Рассмотрим пример работы скрипта start.sh (рис. 3).

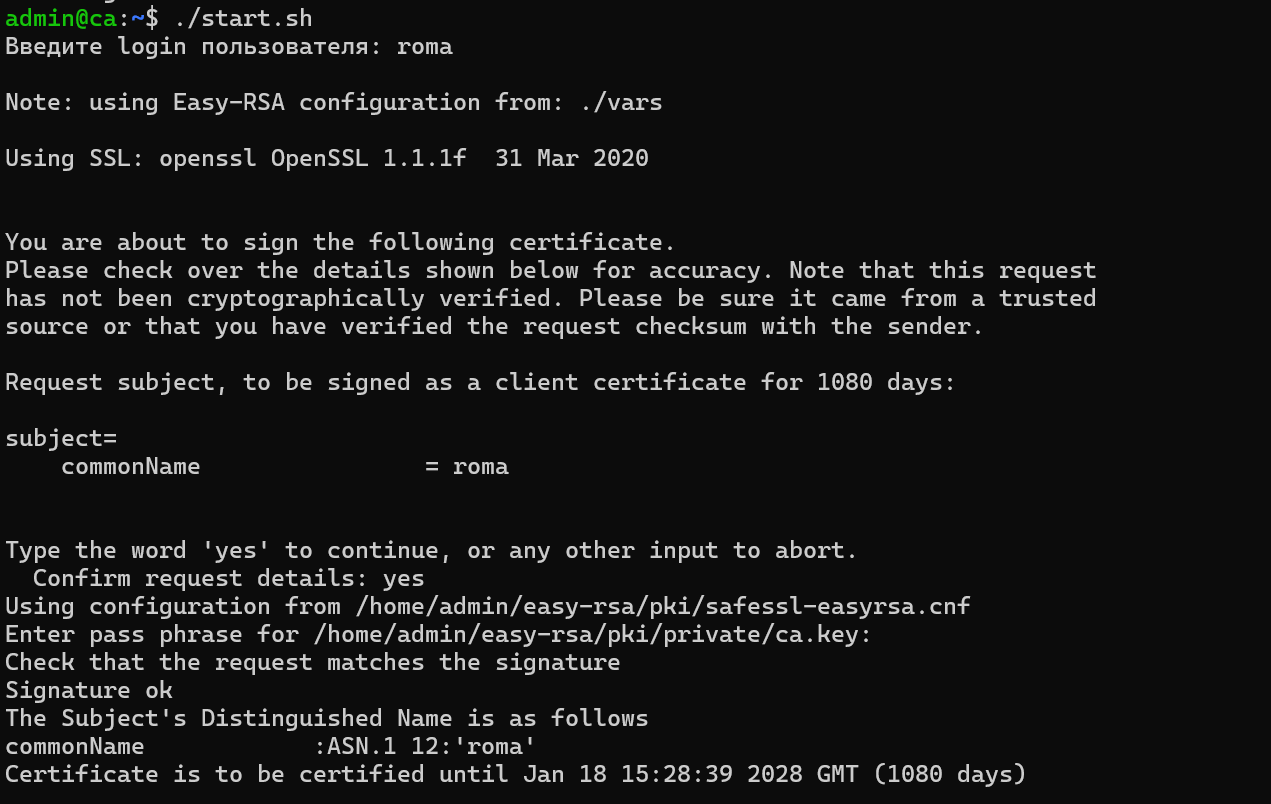


Рисунок 5.1 – пример запуска скрипта start.sh для пользователя roma

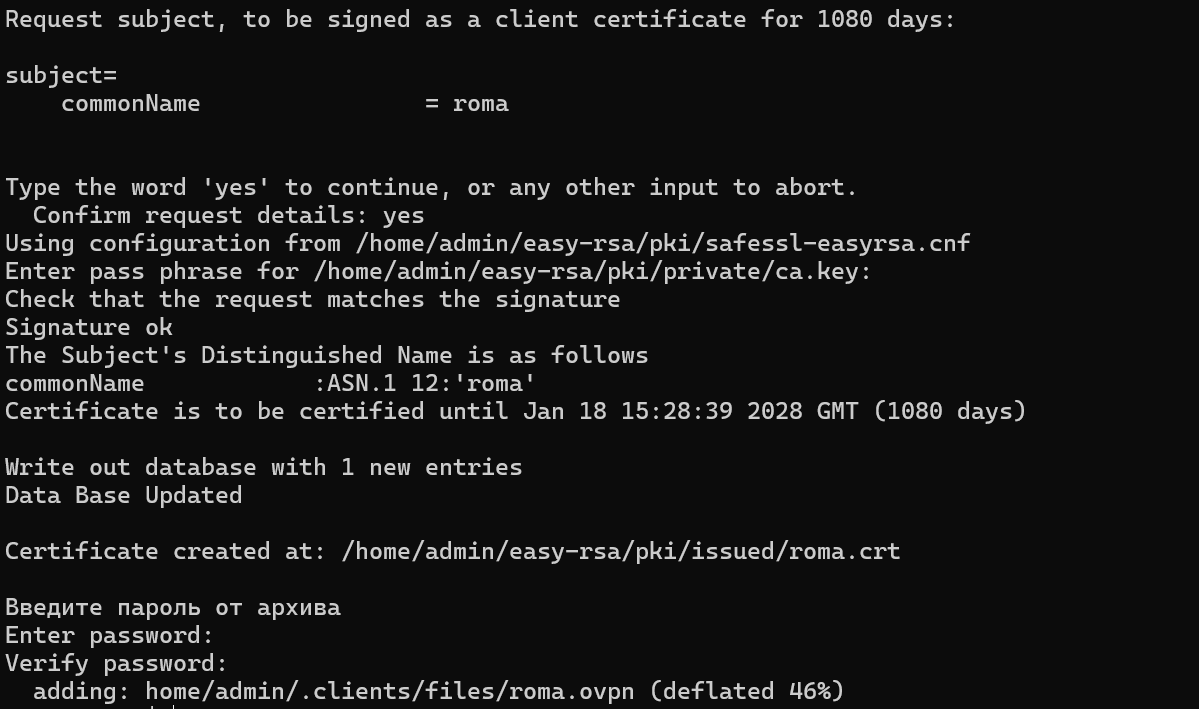


Рисунок 5.2 – окончание скрипта

После выполнения скрипта, в директории ~/.clients/files появятся 2 файла: roma.ovpn и roma.ovpn.zip. Чтобы повысить безопасность, загрузка на гостевой ПК осуществляется только заархивированного файла с помощью команды:

scp -P 222 [admin@150.241.114.219:/home/admin/.clietns/files/”username”.ovpn.zip](mailto:admin@150.241.114.219:/home/admin/.clietns/files/) C:\”path”

Все это представлено в файле инструкции, которые есть на каждом, из серверов проекта ~/Инструкции/Использование. Также в этом файле можно найти информацию по базовым командам в easy-rsa (например, отзыв сертификата).

Как видно, было описано ранее, все работает без всяких проблем, поэтому можем переходить к следующему пункту.

# Настройка мониторинга

Ранее, я описал, что в роли компонента инфраструктуры, отвечающего за мониторинг, выступает Prometheus. Как же он работает?

На каждом сервере были установлены экспортеры, которые собирают метрики, либо состоянию самого сервера (node-exporter), либо состоянию служб (в проекте использованы nginx-exporter и openvpn-exporter). Как только метрики будут получены и переведены в необходимый формат их направляют на сервер мониторинга 37.46.18.51, который примет решение об инициализации алерта, исходя из заданных на нем правил. В случае возникновения ошибок, происходит уведомление администратора по почте (рис. 6) и перевод заданной метрики состояние “firing” (рис. 7).

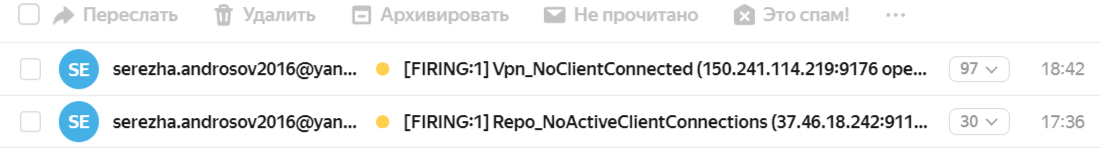


Рисунок 7 – оповещение об алерте по почте.

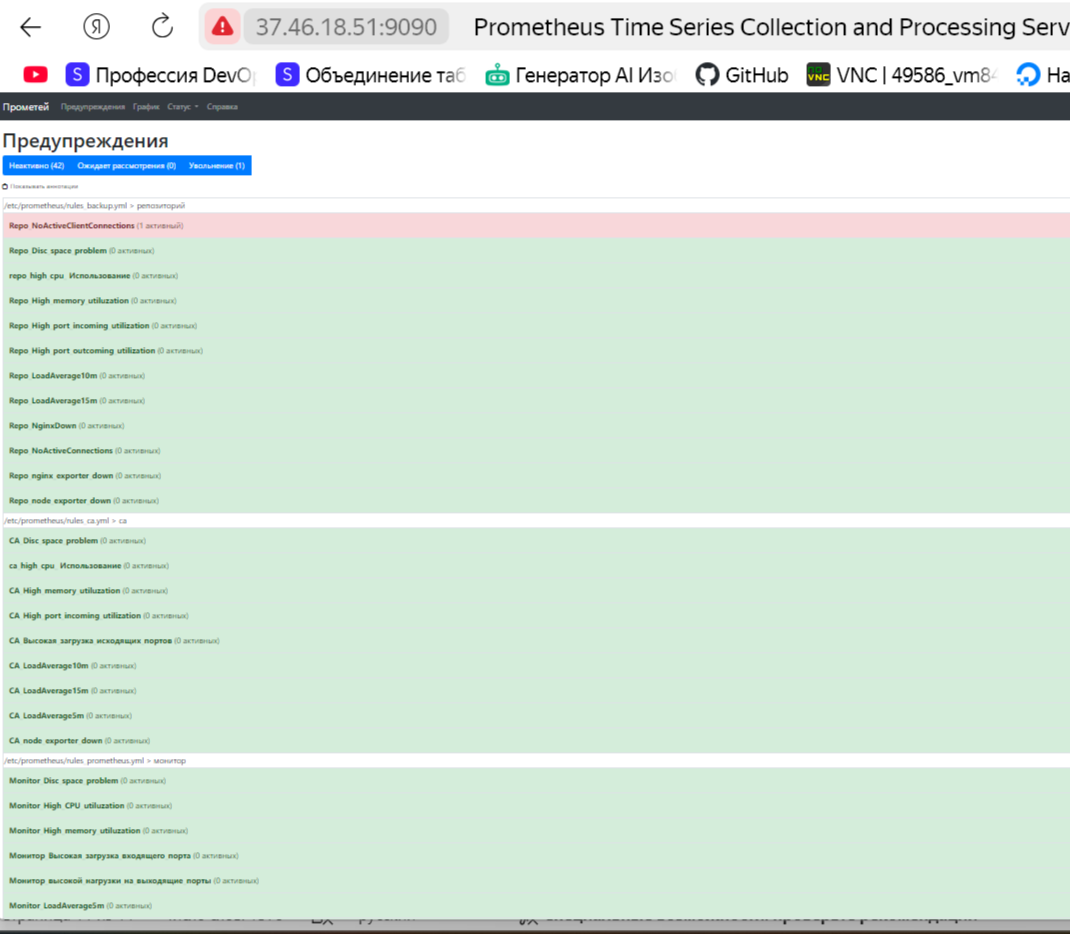


Рисунок 8.1 – Веб интерфейс Prometheus по всем алертам

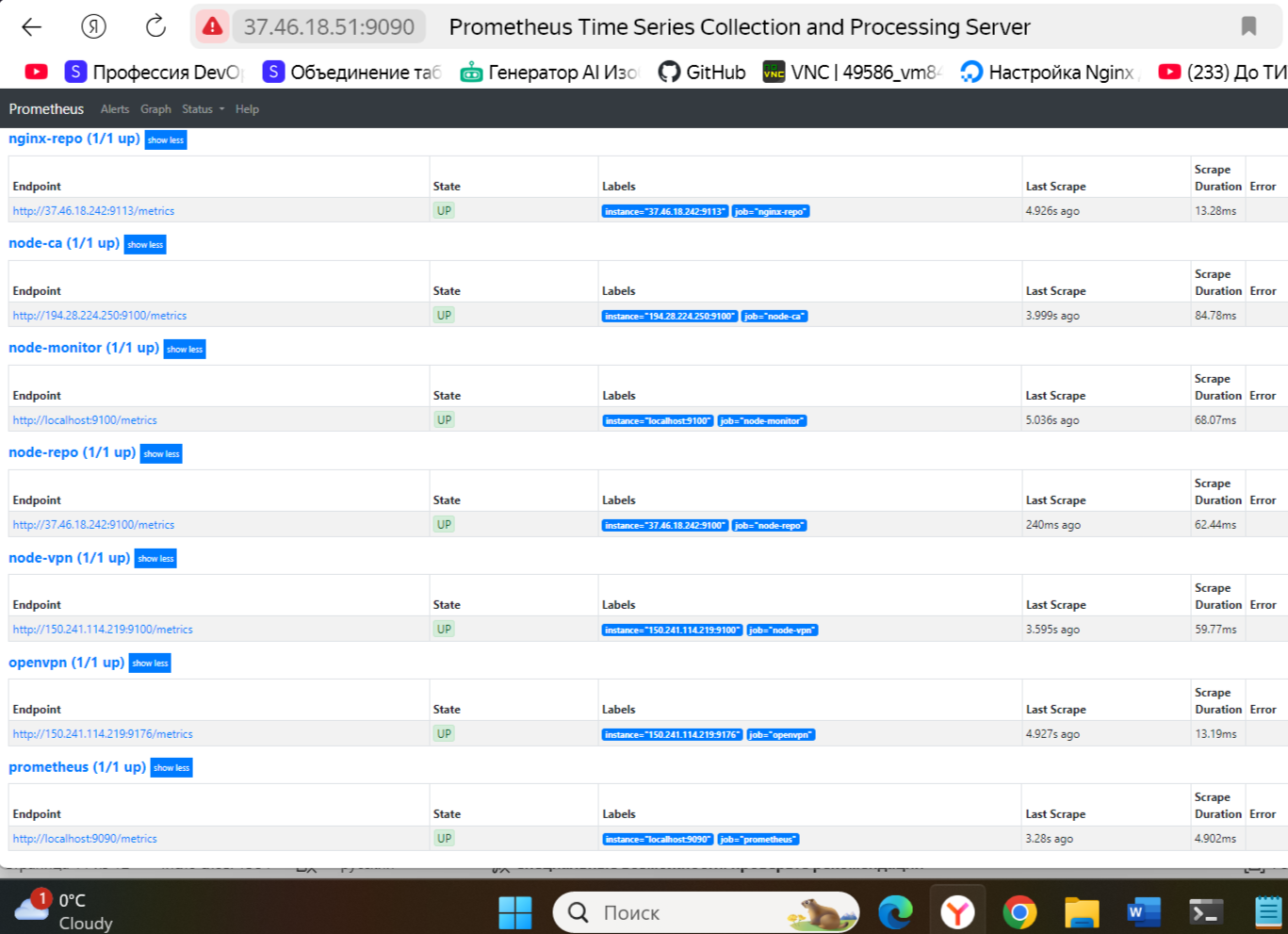


Рисунок 8.2 – Веб интерфейс Prometheus по всем инстансам

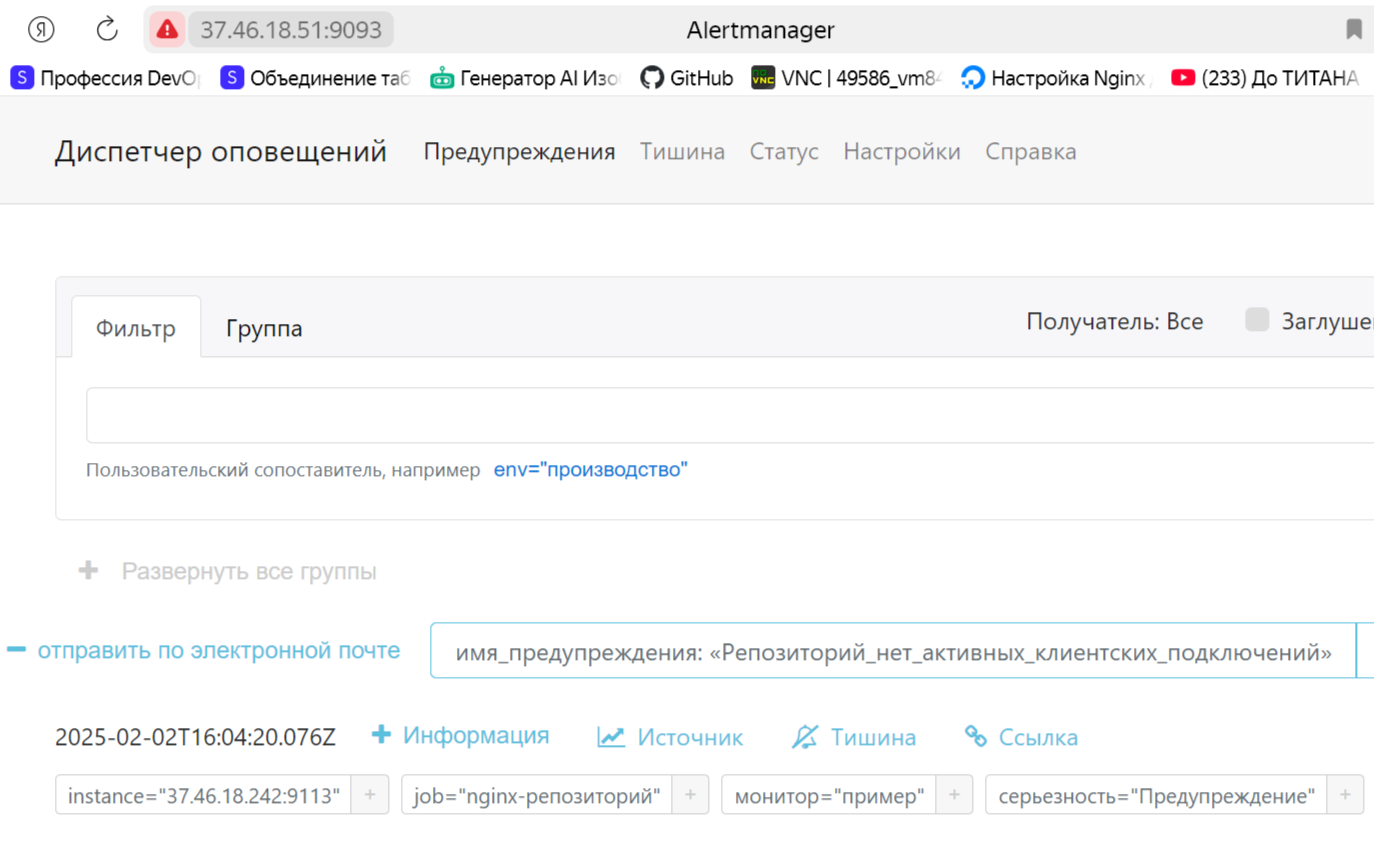


Рисунок 9 -Веб интерфейс alertmanager

В роли основных файлов конфигурации сервере мониторинга выступают prometheus.yml и alertmanager.yml. В файле прометеуса описана, за какими инстансами происходит наблюдение (ip сервера и рабочий порт экспортера), какой интервал сбора данных, какие файлы использовать в качестве правил (rules\*.yml) создания алертов и настроено подключение алертменеджера. В файле alertmanager.yml, настроено по какому правилу происходит информирование об ошибках, на какой адрес и по какому уровню.

Мониторинг осуществляется за всеми основными метриками, такими как место на диске, загрузка процессора, лоуд эвередж, доступность службы и тд. Данные сервисы были неоднократно протестированы и работают исправно.

# Создание системы резервного копирования

В нашей инфраструктуре за резервное копирование отвечает сервер 37.46.18.242 repo. На нем установлен в домашнем каталоге пользователя admin создана директория Backup, в которую приходят deb пакеты со всех остальных инстансов. Также есть скрипт backup.sh, который запускается на основании задачи в cron. Эта программа проверяет раз месяц содержимое каталога, в которые приходит бэкап на **новый** пакет. Если такой найден, то скрипт опубликует его в локальной репозитории lab, где уже хранятся более ранние версии пакетов и, по итогу, произойдет логирование результата. Если такого файла нет, то значит в указанный день (16 каждого месяца) что-то пошло не так, и администратору необходимо вручную запустить backup.sh. Важно!!! Данный скрипт запускается именно от root, поэтому необходимо использовать sudo. То есть, что требуется от admin?

16-го числа залогиниться на сервере с помощью команды:

ssh admin@ 37.46.18.242 -p 222

Проверить содержимое файла backup.log (рис. 10) на наличие свежей записи:

cat ~/Backup/”название пакета”/backup.log

Если такой найдено не было запустить выполнение скрипта backup.sh:

sudo /Backup/”название пакета”/backup.sh

Результат можно проверить в web интерфейсе по адресу 37.46.18.242:1111 (рис. 11).

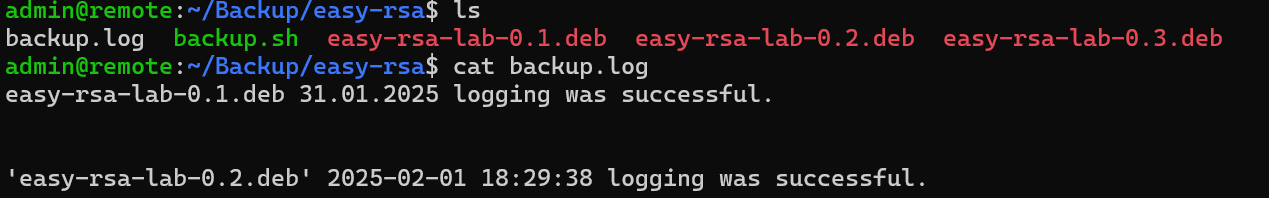


Рисунок 10 – Проверка файла backup.log. Пояснение: 1ая строка записана в ручную, т.к. первая версия пакета была сделана без скрипта.

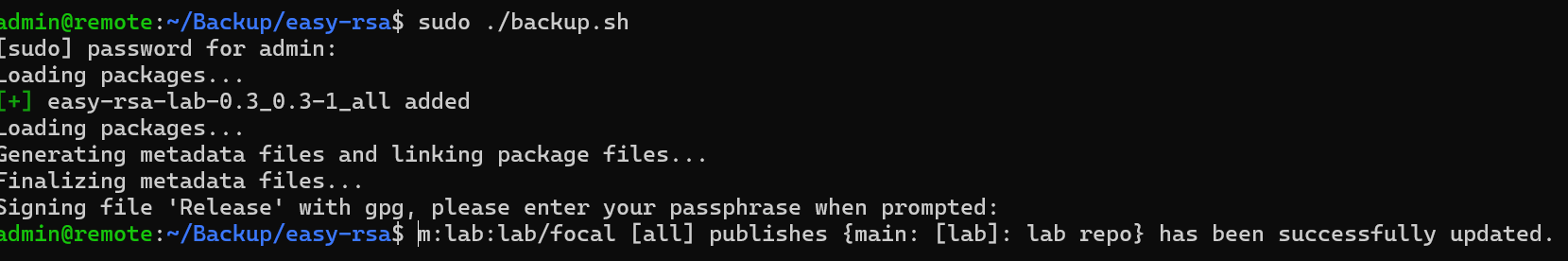


Рисунок 11.1 – публикация пакета в локальный репозиторий с помощью backup.sh

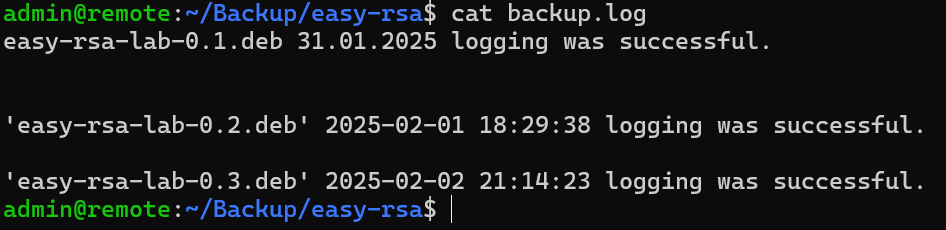


Рисунок 11.2 – Изменение содержимого лога

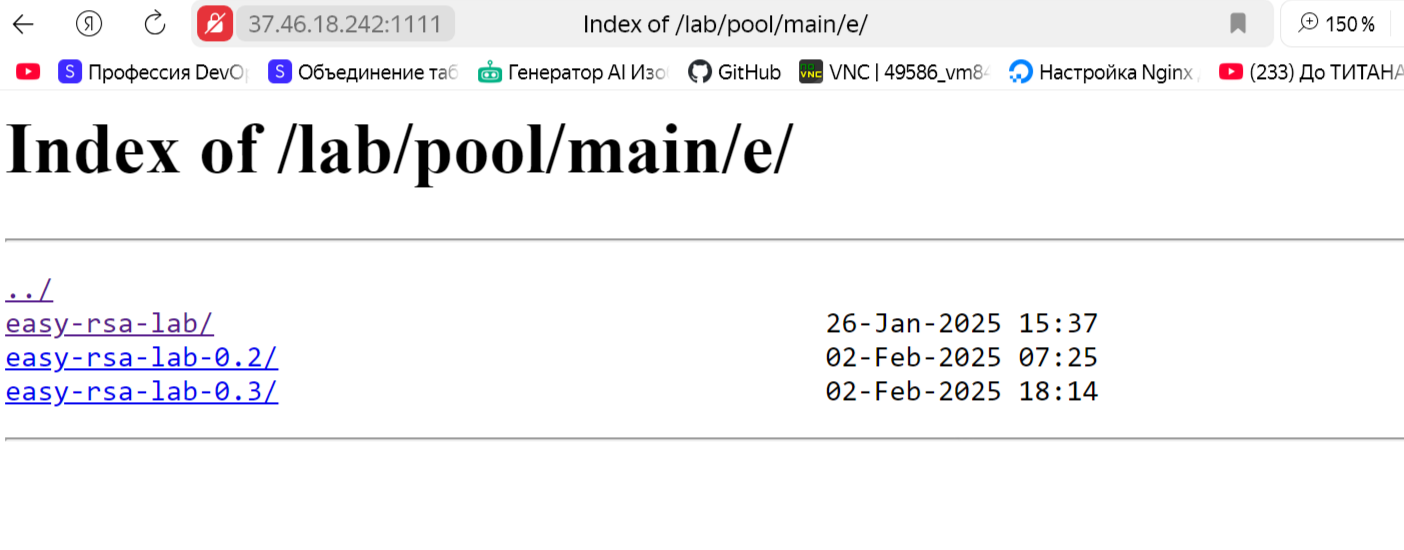


Рисунок 12 - Веб интерфейс локального репозитория

Как видно из рис. 12 публикация пакетов происходит успешно, при желании можно скачать оттуда файл для дальнейшего развертывания на новом сервере.

Касаемо самих пакетов, то было протестировано их использование на новых ВМ. Проблемы возникли только с файлом crontab, так как он вступал в конфликт с уже имеющимся файлом и, очевидно, его замена не является целевым решением и может привести к куче проблем, гораздо проще, на мой взгляд, просто установить правила вручную. Все эти файлы прикреплены к этому проекту и необходимо только скопировать из них необходимые строки.

# План развития инфраструктуры

На текущий момент инфраструктура проекта состоит из четырех серверов, каждый из которых выполняет критически важные функции: VPN-сервер, система мониторинга на базе Prometheus, удостоверяющий центр и репозиторий для хранения данных в формате deb-пакетов. Несмотря на то, что все компоненты работают стабильно и большая часть процессов автоматизирована, текущая архитектура выглядит громоздкой из-за использования четырех отдельных виртуальных частных серверов (VPS). Это приводит к увеличению затрат на обслуживание, усложнению управления и потенциальным проблемам с масштабируемостью.

1. **Оптимизация ресурсов**: Уменьшение количества используемых VPS за счет консолидации сервисов. Первым шагом станет объединение сервисов на меньшем количестве серверов. Например, можно рассмотреть возможность размещения Prometheus и удостоверяющего центра на одном сервере, так как они не требуют значительных ресурсов и могут работать в контейнерах (например, Docker). Это позволит сократить количество VPS до трех, а в перспективе — до двух.
2. **Упрощение управления**: Централизация управления и мониторинга инфраструктуры. Для упрощения управления и повышения переносимости сервисов рекомендуется перевести все компоненты на контейнеры (Docker или Podman). Это позволит:
   1. Упростить развертывание и обновление сервисов.
   2. Изолировать сервисы друг от друга, что повысит безопасность.
   3. Уменьшить зависимость от конкретной операционной системы.
3. **Внедрение оркестрации. Для управления контейнерами и автоматизации процессов развертывания можно использовать системы оркестрации, такие как Kubernetes или Docker Swarm. Это особенно полезно для обеспечения отказоустойчивости и масштабируемости. Например, если нагрузка на репозиторий deb-пакетов увеличится, Kubernetes позволит автоматически масштабировать этот сервис.**
4. **Автоматизация развертывания и управления.** Для дальнейшего повышения уровня автоматизации можно использовать инструменты Infrastructure as Code (IaC), такие как Terraform или Ansible. Это позволит:
   1. Управлять конфигурацией серверов через код.
   2. Быстро развертывать новые сервисы или восстанавливать существующие.
   3. Минимизировать человеческий фактор при настройке инфраструктуры.
5. **Покупка доменных имен**. Эта необходимо для удобства, чтобы к сервисам можно было обращаться не по домену и для работы с протоколом SMTP, чтобы сообщения об ошибках приходили напрямую с сервера.
6. **Использование системы PAM**. Данная система существенно повышает безопасность инфраструктуры, так как логируются все действия пользователя.

В целом этих идей уже достаточно, чтобы долгое время производить модификации нашей инфраструктуры. Несмотря на полностью рабочую среду, как говориться, нет предела совершенству.

# Заключение

Вот и все, неплохая получилась история: интересная, весёлая, порой немного грустная, а главное поучительная. Она научила быть нас смелыми и не бояться вызовов, которые готовит нам жизнь. Помогала нам добавиться поставленных целей несмотря ни на что... Но, самое главное, что у этой истории счастливый конец: итоговый проект написан, все сервера настроены и работают в штатном режиме, а за отказоустойчивость отвечает надежный инструмент мониторинга.

Теперь необходимо набираться новых знаний, чтобы поскорее улучшить данный проект, и сделать его более релевантным и экономически выгодным.