

# Linux Server Recovery and Port Validation

The screenshot shows the Hetzner Console interface. The top navigation bar includes the Hetzner logo, 'Console', a project selector, a search bar, and user controls. A banner for 'Important status messages' is visible. The left sidebar contains a navigation menu with categories like Dashboard, Pinned, Cloud, and Networking. The main content area displays details for server 'CPX22' (MUM), including its IP addresses (91.99.163.16 and 2a01:4f8:1c1a:e81c::/64) and a 'PUBLIC NETWORK' section. This section contains a table with network details:

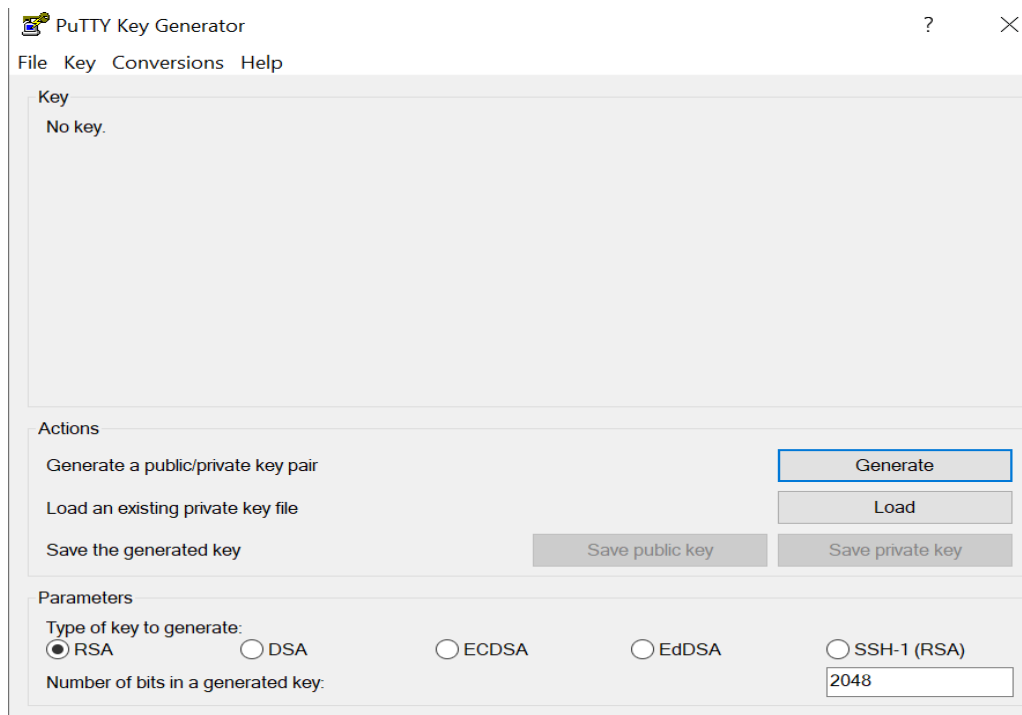
PRIMARY IP	REVERSE DNS
91.99.163.16	static.16.163.99.91.clients.your-server.de
2a01:4f8:1c1a:e81c::/64	0 Entries

Below the table is a 'Disable public network' button.

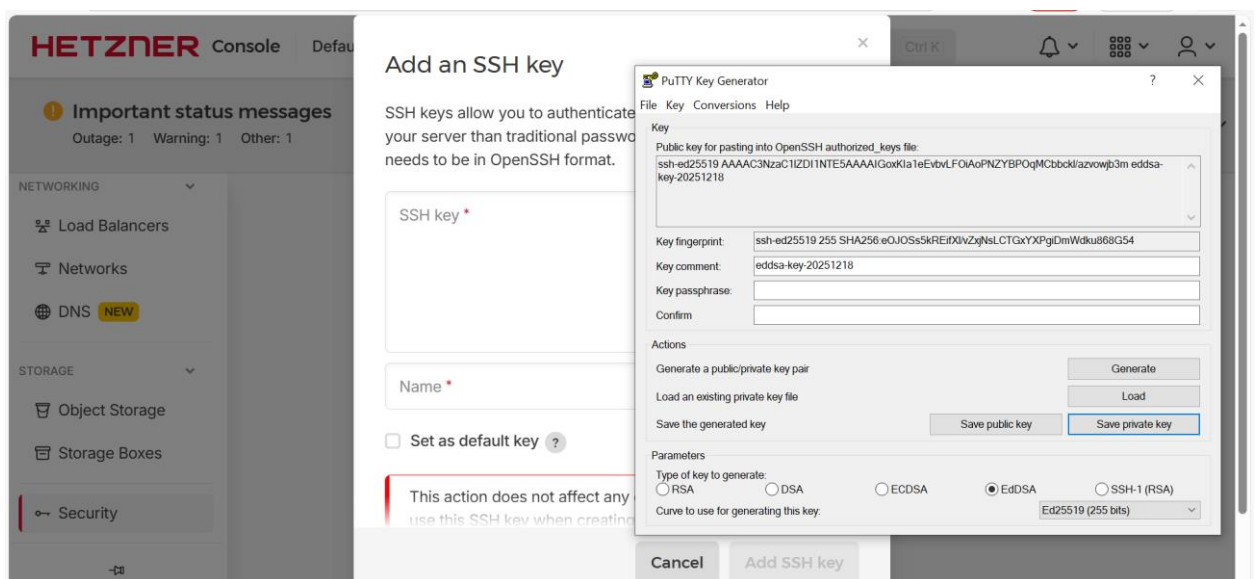
В панели Hetzner открываем вкладку Networking и смотрим сетевые параметры сервера (Public Network). Мы видим публичный IPv4 сервера **91.99.163.16**, а также IPv6 и Reverse DNS — это адрес, по которому будем подключаться по SSH и проверять доступность портов и сервисов.

The screenshot shows the 'Security - SSH keys' page in the Hetzner Console. The URL bar indicates the path: `https://console.hetzner.com/projects/12693650/security/sshkeys`. The left sidebar shows the 'Security' section selected. The main content area features a large key icon and the message: 'You haven't added an SSH key yet.' Below this, explanatory text states: 'SSH keys are a method to authenticate yourself to a server in a way that is more convenient and secure than traditional password authentication. Adding an SSH key has no impact on existing resources.' A prominent red button labeled 'Add SSH key' is at the bottom.

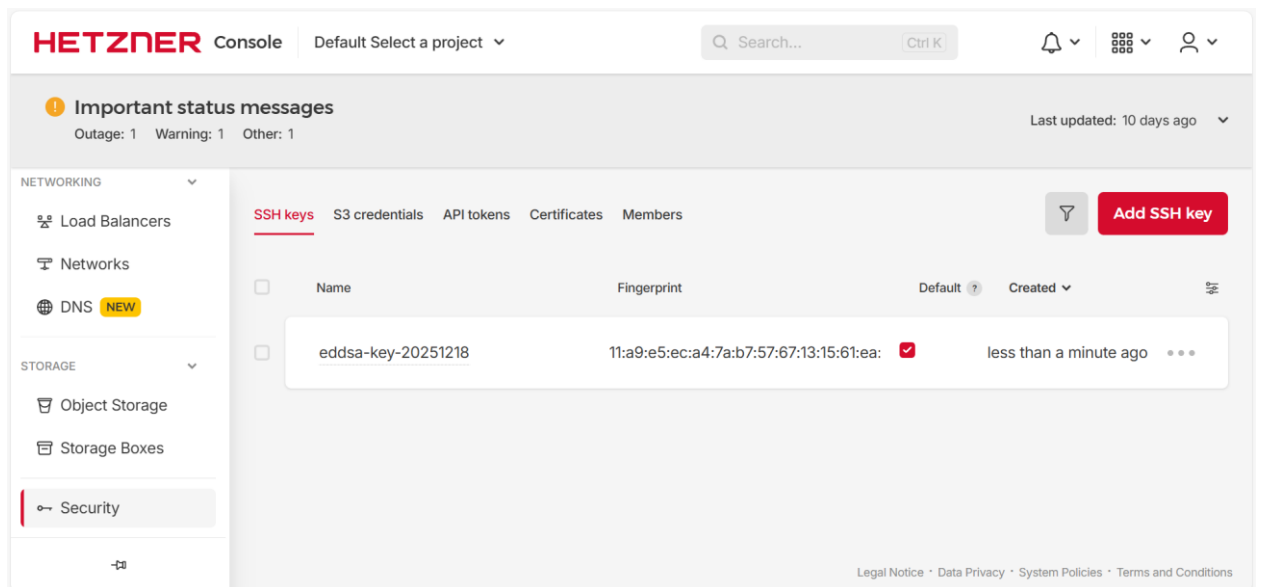
В панели Hetzner переходим в раздел **Security - SSH keys**. Мы видим, что SSH-ключей ещё нет, поэтому сервер пока не настроен для входа по ключу. Это означает, что нам необходимо создать и добавить SSH-ключ, чтобы обеспечить безопасную аутентификацию.



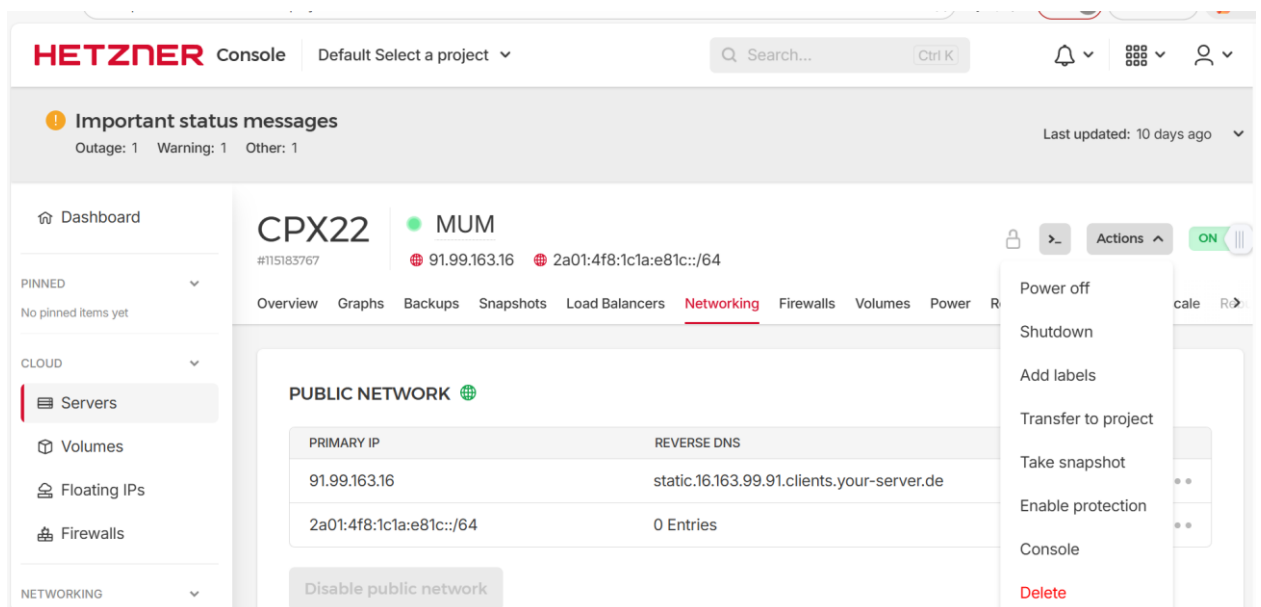
Открываем программу **PuTTY Key Generator (PuTTYgen)** для создания SSH-ключа. На этом этапе мы генерируем пару ключей — публичный и приватный. Публичный ключ будем добавлять в панель Hetzner, а приватный останется у нас для безопасного подключения к серверу по SSH.



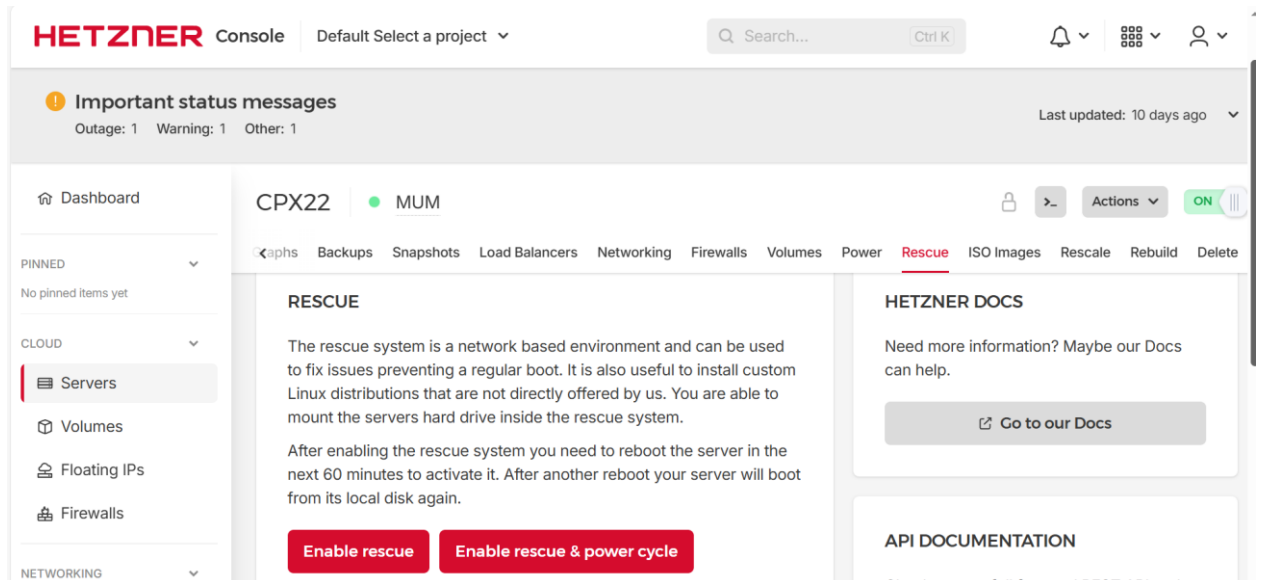
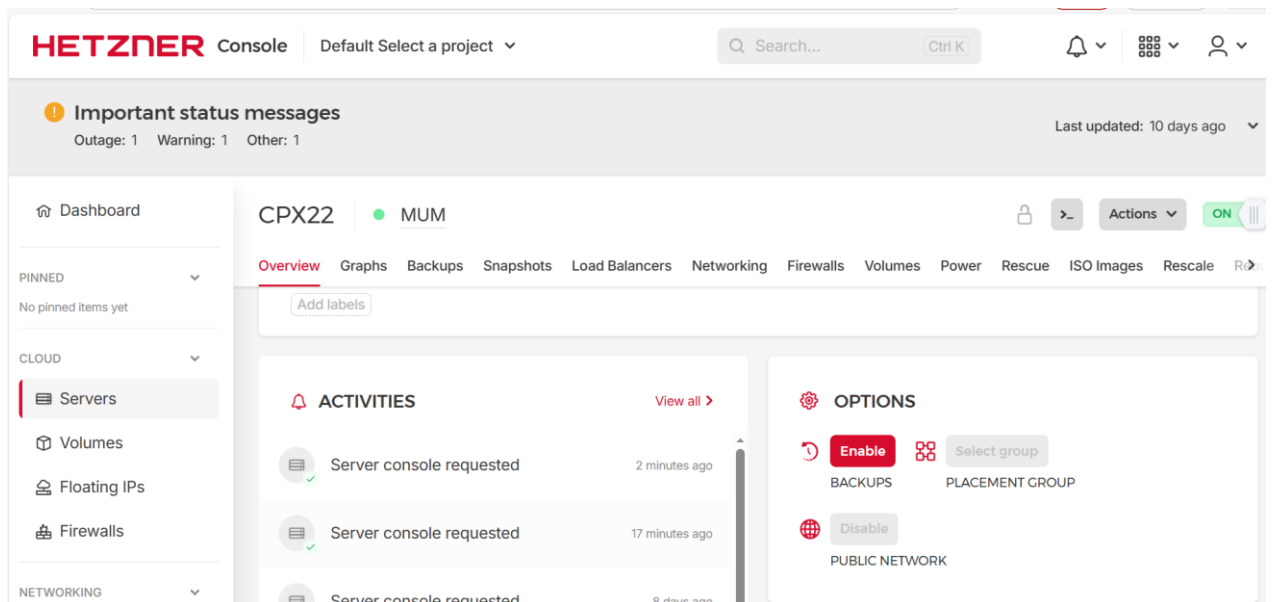
После генерации ключа в PuTTYgen копируем **публичную часть SSH-ключа** и вставляем её в форму добавления ключа в панели Hetzner (Add SSH key). Таким образом мы регистрируем наш публичный ключ в облачной панели, чтобы в дальнейшем использовать его для безопасного доступа к серверу.



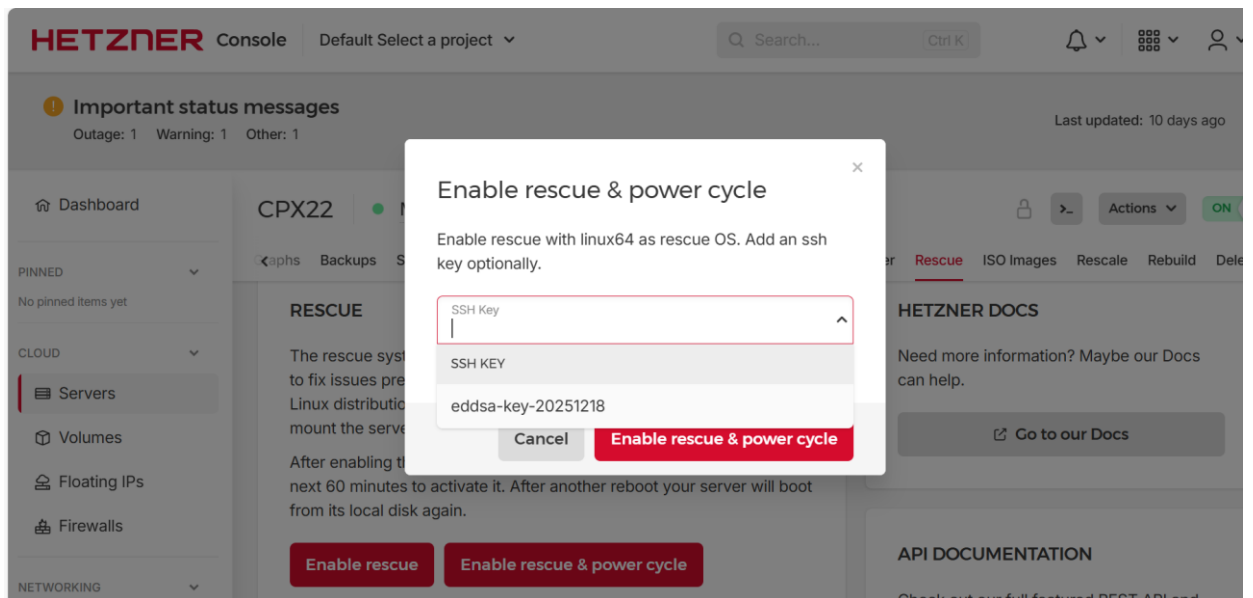
После добавления ключа возвращаемся в раздел **SSH keys** и видим, что новый ключ появился в списке. Это означает, что публичный SSH-ключ успешно сохранён в панели Hetzner и теперь может использоваться для доступа к серверу или для загрузки в режиме Rescue.



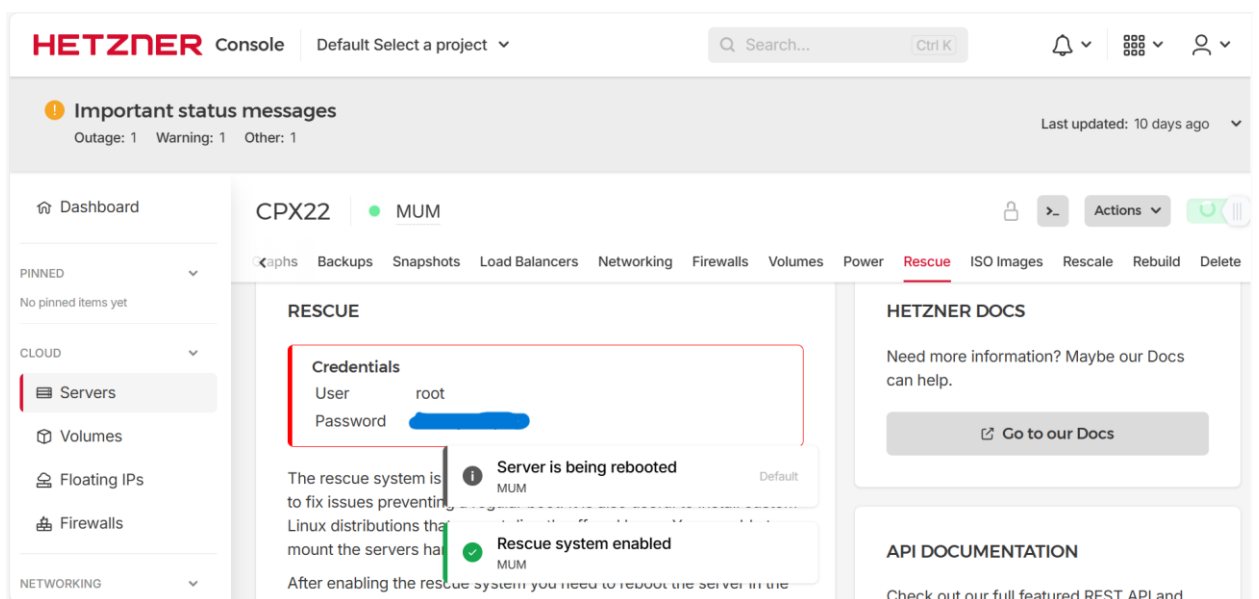
В панели управления сервером открываем раздел **Actions**, где доступны операции управления сервером (перезагрузка, выключение, консоль, Rescue и другие действия). Здесь мы готовимся перейти в режим Rescue, чтобы получить доступ к системе для восстановления или настройки SSH-доступа.



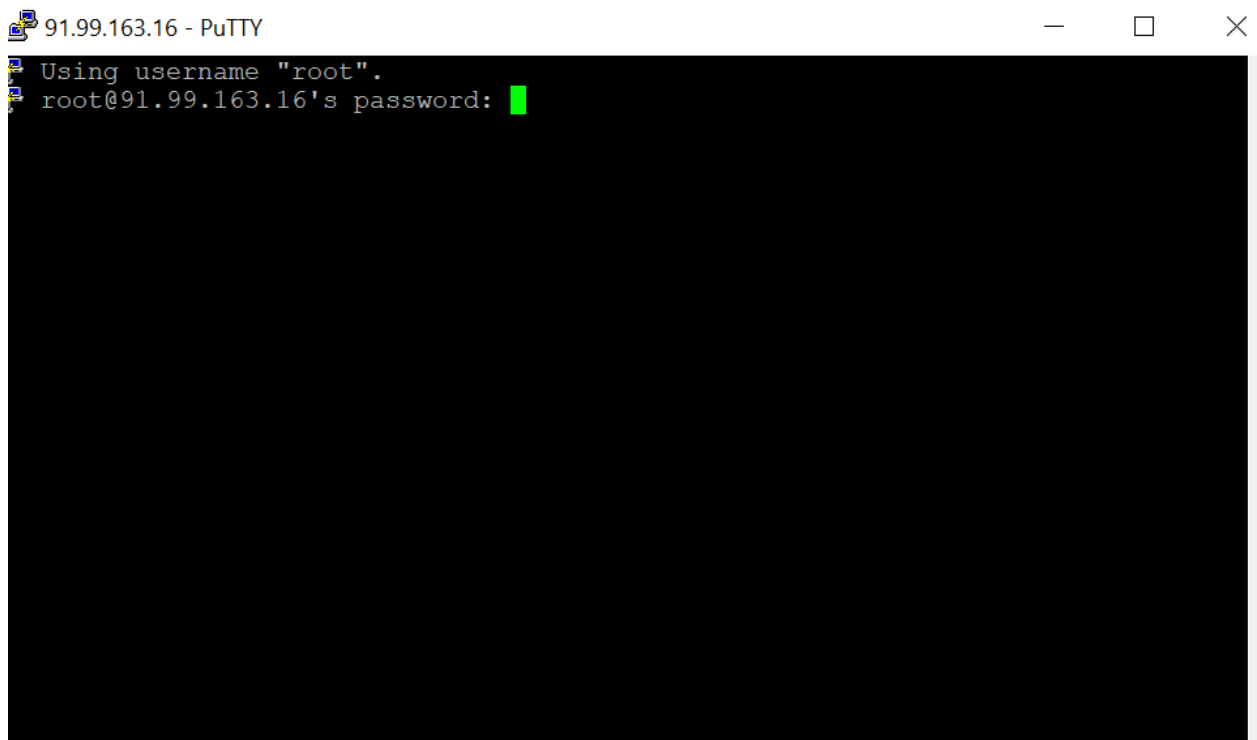
Переходим во вкладку **Rescue** в панели Hetzner. Здесь можно включить режим аварийной загрузки сервера. Rescue-система — это отдельная временная операционная система, которая загружается вместо основной и позволяет получить доступ к дискам сервера для восстановления системы или изменения конфигурации.



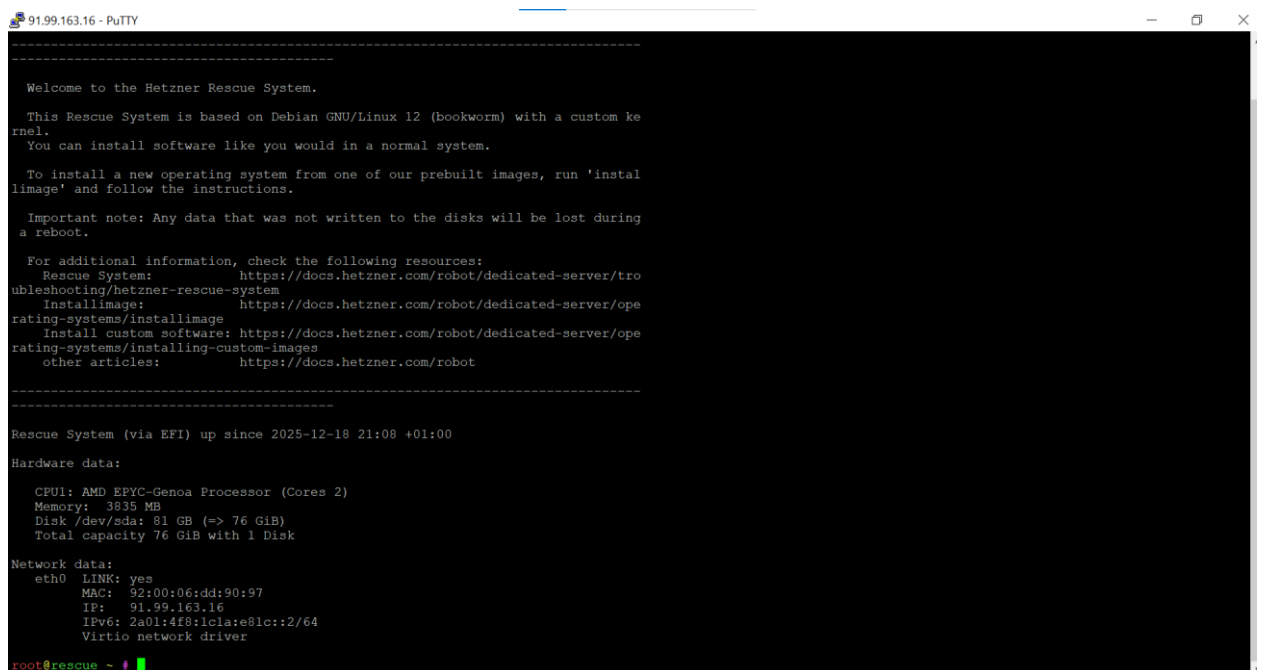
Во вкладке **Rescue** включаем режим аварийной загрузки и выбираем ранее добавленный SSH-ключ. После этого выполняем **Power cycle**, чтобы сервер перезагрузился и загрузился уже в Rescue-системе. Таким образом мы получаем возможность подключиться к серверу в специальном режиме для восстановления доступа.



После активации Rescue-системы панель Hetzner показывает подтверждение включения режима и временные данные для входа (root и пароль). Это означает, что сервер теперь загружен в Rescue-окружение, и мы можем подключиться к нему по SSH для выполнения восстановительных действий.



Подключаемся к серверу по SSH через PuTTY, указывая публичный IP-адрес сервера и пользователя root. Система запрашивает пароль, выданный в режиме Rescue. Это шаг аутентификации, после которого мы получим доступ к серверу в аварийном режиме.



После успешной аутентификации мы видим приветственное сообщение **Hetzner Rescue System**. Это подтверждает, что сервер загружен не в основную операционную систему, а в аварийное Rescue-окружение.

```

91.99.163.16 - PuTTY
Important note: Any data that was not written to the disks will be lost during
a reboot.

For additional information, check the following resources:
  Rescue System:      https://docs.hetzner.com/robot/dedicated-server/tro
ubleshooting/hetzner-rescue-system
  Installimage:       https://docs.hetzner.com/robot/dedicated-server/ope
rating-systems/installimage
  Install custom software: https://docs.hetzner.com/robot/dedicated-server/ope
rating-systems/installing-custom-images
  other articles:      https://docs.hetzner.com/robot

-----
Rescue System (via EFI) up since 2025-12-18 21:08 +01:00

Hardware data:

CPU1: AMD EPYC-Genoa Processor (Cores 2)
Memory: 3835 MB
Disk /dev/sda: 81 GB (=> 76 GiB)
Total capacity 76 GiB with 1 Disk

Network data:
eth0  LINK: yes
      MAC: 92:00:06:dd:90:97
      IP:  91.99.163.16
      IPv6: 2a01:4f8:1c1a:e81c::2/64
      Virtio network driver

root@rescue ~ # lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0       7:0      0   3.4G 1 loop
sda         8:0      0   76.3G 0 disk
├─sda1      8:1      0    76G 0 part
├─sda14     8:14     0     1M 0 part
└─sda15     8:15     0   256M 0 part
sr0         11:0     1 1024M 0 rom

root@rescue ~ # mount /dev/sda1 /mnt
root@rescue ~ # ls /mnt
bin  cdrom  etc  lib  lib64  lost+found  mnt  proc  run  snap  sys  usr
boot dev  home  lib32  libx32  media  opt  root  sbin  srv  var
root@rescue ~ #

```

Rescue-системе выполняем команду `lsblk`, чтобы посмотреть список дисков и разделов сервера. Затем монтируем основной раздел системы, `/dev/sda1`, в каталог `/mnt` с помощью команды `mount`. После этого проверяем содержимое каталога `/mnt` и убеждаемся, что видим стандартную структуру Linux (`bin`, `etc`, `home` и другие директории). Это означает, что мы успешно подключили файловую систему основной операционной системы и можем вносить в неё изменения.

```

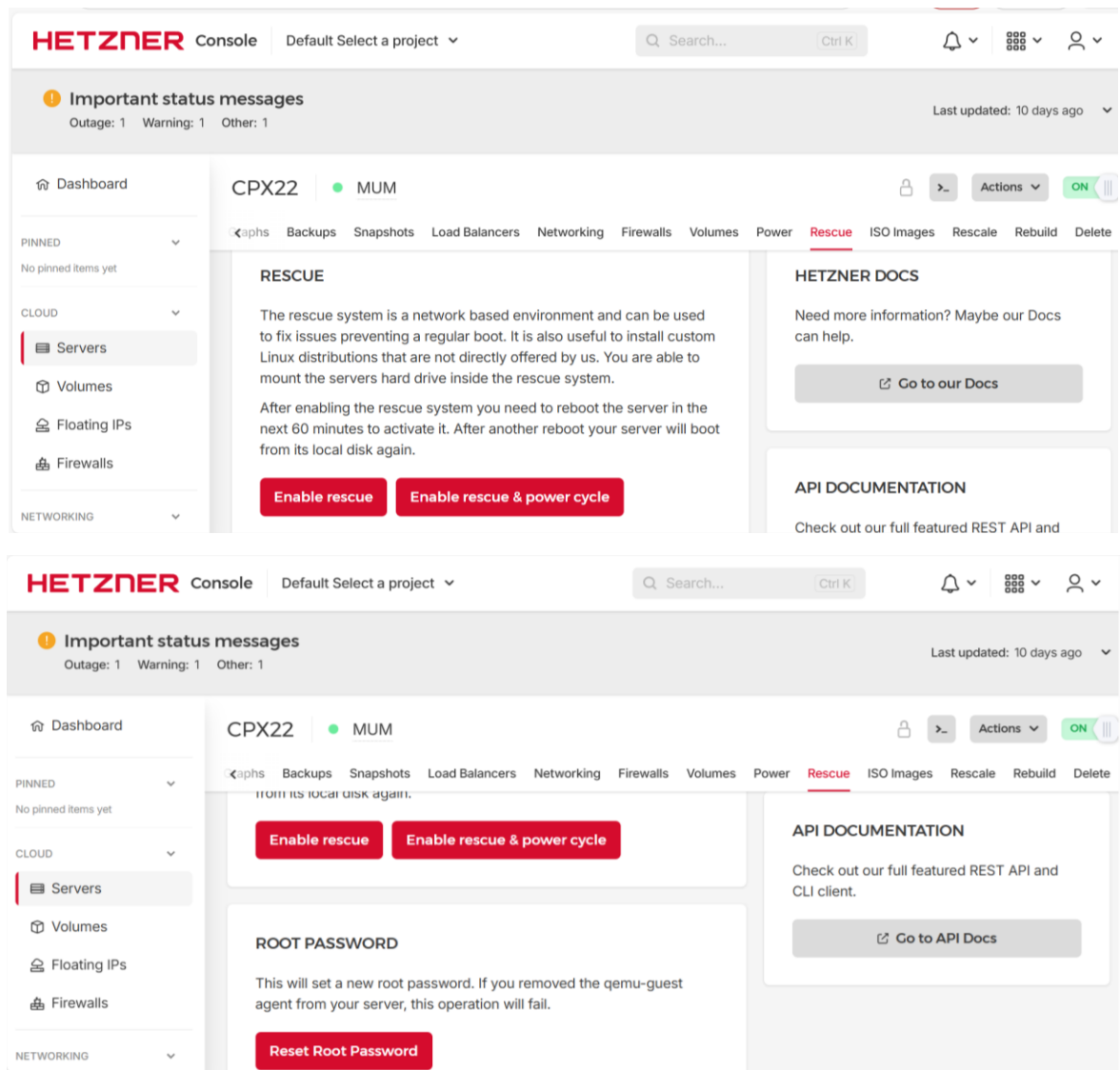
root@rescue ~ # ls /mnt
bin  cdrom  etc  lib  lib64  lost+found  mnt  proc  run  snap  sys  usr
boot dev  home  lib32  libx32  media  opt  root  sbin  srv  var
root@rescue ~ # ls -l /mnt/root
total 4.0K
drwx----- 3 root root 4.0K Nov  2 05:23 snap
root@rescue ~ # mkdir -p /mnt/root/.ssh
root@rescue ~ # ls -la /mnt/root
total 40K
drwx----- 7 root root 4.0K Dec 18 21:23 .
drwxr-xr-x 20 root root 4.0K Dec 10 18:56 ..
-rw----- 1 root root 1.8K Dec 10 21:08 .bash_history
-rw-r--r-- 1 root root 3.1K Oct 15  2021 .bashrc
drwx----- 2 root root 4.0K Nov  2 05:23 .cache
-rw-r--r-- 1 root root  0 Nov  2 05:27 .cloud-locale-test.skip
drwxr-xr-x 3 root root 4.0K Dec 10 19:10 .local
-rw-r--r-- 1 root root 161 Jul  9  2019 .profile
drwxr-xr-x 2 root root 4.0K Dec 18 21:23 .ssh
drwx----- 3 root root 4.0K Nov  2 05:23 snap
drwx----- 2 root root 4.0K Nov  2 05:23 .ssh

root@rescue ~ # touch /mnt/root/.ssh/authorized_keys
root@rescue ~ # ls /mnt/root/.ssh
authorized_keys
root@rescue ~ # nano /mnt/root/.ssh/authorized_keys
root@rescue ~ # cat /mnt/root/.ssh/authorized_keys
root@rescue ~ # cat /mnt/root/.ssh/authorized_keys
root@rescue ~ # nano /mnt/root/.ssh/authorized_keys
root@rescue ~ # cat /mnt/root/.ssh/authorized_keys
root@rescue ~ # chmod 600 /mnt/root/.ssh/authorized_keys
root@rescue ~ # nano /mnt/root/.ssh/authorized_keys
root@rescue ~ # cat /mnt/root/.ssh/authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIGoxK1aleEvvbLF01AoFNZYBPOqMCbbck1/azvowjb3m eddsa-key-20251218
root@rescue ~ # chmod 600 /mnt/root/.ssh/authorized_keys
root@rescue ~ # reboot

```

В Rescue-системе создаём каталог `/mnt/root/.ssh`, если он отсутствует, затем создаём файл `authorized_keys` с помощью команды `touch`. После этого открываем файл и вставляем в него наш публичный SSH-ключ. Далее устанавливаем корректные права доступа: `chmod 600` для файла `authorized_keys`, чтобы обеспечить безопасность SSH-аутентификации. В конце выполняем `reboot`, чтобы сервер перезагрузился и применил изменения.

Таким образом, мы вручную добавляем SSH-ключ в основную систему, чтобы в дальнейшем входить на сервер без использования временного rescue-пароля.



После перезагрузки сервер выходит из Rescue-режима и загружается в основную операционную систему. Мы снова подключаемся по SSH к серверу, и теперь вход выполняется с использованием ранее добавленного публичного ключа. Это подтверждает, что SSH-доступ успешно восстановлен и сервер работает в штатном режиме.



```
root@MUM: ~  
login as: root  
Authenticating with public key "eddsa-key-20251218"  
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-164-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/pro  
  
System information as of Thu Dec 18 10:43:08 PM UTC 2025  
  
System load:  0.01      Processes:      128  
Usage of /:   2.6% of 74.79GB    Users logged in: 0  
Memory usage: 5%          IPv4 address for eth0: 91.99.163.16  
Swap usage:   0%          IPv6 address for eth0: 2a01:4f8:1c1a:e81c::1  
  
* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s  
  just raised the bar for easy, resilient and secure K8s cluster deployment.  
  https://ubuntu.com/engage/secure-kubernetes-at-the-edge  
  
Expanded Security Maintenance for Applications is not enabled.  
  
5 updates can be applied immediately.  
To see these additional updates run: apt list --upgradable  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
New release '24.04.3 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Thu Dec 18 18:56:25 2025 from 178.223.74.164  
root@MUM:~#
```

```
root@MUM:~#  
^C  
root@MUM:~# systemctl status xray  
● xray.service - Xray Service  
   Loaded: loaded (/etc/systemd/system/xray.service; enabled; vendor preset:   
   Drop-In: /etc/systemd/system/xray.service.d  
            └─10-donot_touch_single_conf.conf  
   Active: active (running) since Thu 2025-12-18 22:39:20 UTC; 27min ago  
     Docs: https://github.com/xtls  
  Main PID: 686 (xray)  
    Tasks: 9 (limit: 4523)  
   Memory: 36.6M  
      CPU: 202ms  
   CGroup: /system.slice/xray.service  
            └─686 /usr/local/bin/xray run -config /usr/local/etc/xray/config.j  
  
Dec 18 22:39:20 MUM systemd[1]: Started Xray Service.  
Dec 18 22:39:20 MUM xray[686]: Xray 25.12.8 (Xray, Penetrates Everything.) 81f8  
Dec 18 22:39:20 MUM xray[686]: A unified platform for anti-censorship.  
Dec 18 22:39:20 MUM xray[686]: 2025/12/18 22:39:20.251960 [Info] infra/conf/ser  
Dec 18 22:39:20 MUM xray[686]: 2025/12/18 22:39:20.274391 [Warning] core: Xray  
lines 1-18/18 [END]...skipping...  
● xray.service - Xray Service  
   Loaded: loaded (/etc/systemd/system/xray.service; enabled; vendor preset: enabled)  
   Drop-In: /etc/systemd/system/xray.service.d  
            └─10-donot_touch_single_conf.conf  
   Active: active (running) since Thu 2025-12-18 22:39:20 UTC; 27min ago  
     Docs: https://github.com/xtls  
  Main PID: 686 (xray)  
    Tasks: 9 (limit: 4523)  
   Memory: 36.6M  
      CPU: 202ms  
   CGroup: /system.slice/xray.service  
            └─686 /usr/local/bin/xray run -config /usr/local/etc/xray/config.json  
  
Dec 18 22:39:20 MUM systemd[1]: Started Xray Service.  
Dec 18 22:39:20 MUM xray[686]: Xray 25.12.8 (Xray, Penetrates Everything.) 81f8f39 (gol.25.5 linux/amd64)  
Dec 18 22:39:20 MUM xray[686]: A unified platform for anti-censorship.  
Dec 18 22:39:20 MUM xray[686]: 2025/12/18 22:39:20.251960 [Info] infra/conf/serial: Reading config: &{Name=/usr/local/etc/xray/config.json Format=json}  
Dec 18 22:39:20 MUM xray[686]: 2025/12/18 22:39:20.274391 [Warning] core: Xray 25.12.8 started
```

После успешного входа в основную систему Ubuntu мы проверяем состояние сервиса **xray** с помощью команды `systemctl status xray`. В выводе видно, что сервис загружен (loaded), включён для автозапуска (enabled) и находится в состоянии active (running). Это подтверждает, что служба запущена корректно и работает без критических ошибок.

```
root@MUM:~# ss -tulpn | grep xray  
tcp    LISTEN 0      4096      *:443      *:~        users:(("xra  
y",pid=686,fd=3))  
root@MUM:~# ss -tulpn | grep 443  
tcp    LISTEN 0      4096      *:443      *:~        users:(("xra  
y",pid=686,fd=3))  
root@MUM:~# nano /usr/local/etc/xray/config.json
```

Проверяем, прослушивает ли сервис нужный порт, с помощью команды `ss -tulpn | grep 443`. В выводе видно, что процесс **xray** слушает порт **443** на всех интерфейсах. Это означает, что сервис корректно привязан к HTTPS-порту и готов принимать входящие подключения из сети.

```
root@MUM: ~  
GNU nano 6.2 /usr/local/etc/xray/config.json  
"log": {  
  "loglevel": "warning"  
},  
"inbounds": [  
  {  
    "listen": "0.0.0.0",  
    "port": 443,  
    "protocol": "vless",  
    "settings": {  
      "clients": [  
        {  
          "id": "dbf79f8a-ac88-4ce2-b0b4-340a62e63498",  
          "flow": "xtls-rprx-vision"  
        }  
      ],  
      "decryption": "none"  
    },  
    "streamSettings": {  
      "network": "tcp",  
      "security": "reality",  
      "realitySettings": {  
        "show": false,  
        "dest": "www.cloudflare.com:443",  
        "xver": 0,  
        "serverNames": [  
          "www.cloudflare.com"  
        ],  
        "privateKey": "www.cloudflare.com:443",  
        "shortIds": [  
          "d0ed093792455f46"  
        ]  
      }  
    }  
  ],  
  "outbounds": [  
    {  
      "protocol": "freedom",  
      "settings": {}  
    }  
  ]  
}
```

Открываем конфигурационный файл сервиса Xray с помощью команды `nano /usr/local/etc/xray/config.json`. В файле проверяем параметры входящего подключения: порт 443, сетевые настройки и параметры безопасности. Это позволяет убедиться, что сервис настроен корректно и использует нужные значения для обработки входящих соединений.

```
root@MUM: ~  
y",pid=686,fd=3))  
root@MUM:~# nano /usr/local/etc/xray/config.json  
root@MUM:~# /usr/local/bin/xray x25519  
PrivateKey: [REDACTED]  
Password: [REDACTED]  
Hash32: LGS11Iuh6QsmtgnpUSq3XCbVCoTRO--hg0f4w2PpKKA  
root@MUM:~# journalctl -u xray -f  
Dec 10 18:53:33 MUM systemd[1]: Started Xray Service.  
Dec 10 18:53:33 MUM xray[7720]: Xray 25.12.8 (Xray, Penetrates Everything.) 81f8f39 (gol.25.5 linux/amd64)  
Dec 10 18:53:33 MUM xray[7720]: A unified platform for anti-censorship.  
Dec 10 18:53:33 MUM xray[7720]: 2025/12/10 18:53:33.996925 [Info] infra/conf/serial: Reading config: &{Name:/usr/local/etc/xray/config.json Format:json}  
Dec 10 18:53:34 MUM xray[7720]: 2025/12/10 18:53:34.000774 [Warning] core: Xray 25.12.8 started  
-- Boot 984f07ed489a4aa09f2a41a86af98778 --  
Dec 18 22:39:20 MUM systemd[1]: Started Xray Service.  
Dec 18 22:39:20 MUM xray[686]: Xray 25.12.8 (Xray, Penetrates Everything.) 81f8f39 (gol.25.5 linux/amd64)  
Dec 18 22:39:20 MUM xray[686]: A unified platform for anti-censorship.  
Dec 18 22:39:20 MUM xray[686]: 2025/12/18 22:39:20.251960 [Info] infra/conf/serial: Reading config: &{Name:/usr/local/etc/xray/config.json Format:json}  
Dec 18 22:39:20 MUM xray[686]: 2025/12/18 22:39:20.274391 [Warning] core: Xray 25.12.8 started
```

Проверяем логи сервиса с помощью команды `journalctl -u xray -f`. В режиме реального времени наблюдаем сообщения о запуске и работе службы. Отсутствие критических ошибок в логах подтверждает, что сервис функционирует корректно и обрабатывает подключения без сбоев.

```
root@MUM:~# /usr/local/bin/xray x25519 -i [REDACTED]  
PrivateKey: [REDACTED]  
Password: [REDACTED]  
Hash32: RHdOhrASut5TWygmnoNjcmegR1lHXppAmnOrvvy1qFA  
root@MUM:~# grep -n "privateKey" /usr/local/etc/xray/config.json  
39:   "privateKey": [REDACTED],  
root@MUM:~# grep -n "shortIds" /usr/local/etc/xray/config.json  
40:   "shortIds": [  
root@MUM:~# /usr/local/bin/xray x25519  
PrivateKey: [REDACTED]  
Password: [REDACTED]  
Hash32: r-la--CeDvMlyURYj5NAPYRU769Vpo2TK23Bzh7Sfo4  
root@MUM:~# /usr/local/bin/xray x25519 -i [REDACTED]  
PrivateKey: [REDACTED]  
Password: [REDACTED]  
Hash32: z-mpAlsWquHDiYEqrUfaJ9YsW-wp9co-Qzov07V7M  
root@MUM:~#
```

Проверяем работу сервиса и криптографические параметры, используя встроенные команды Xray для генерации ключей и просмотра конфигурации. Это позволяет убедиться, что необходимые параметры безопасности (ключи и идентификаторы) корректно заданы в конфигурационном файле.

```
Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Test-NetConnection - 91.99.163.16:443
    Attempting TCP connect
    Waiting for response

Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/pscore6)

PS C:\Users\Борис> Test-NetConnection 91.99.163.16 -Port 443

ComputerName      : 91.99.163.16
RemoteAddress     : 91.99.163.16
RemotePort        : 443
InterfaceAlias    : Беспроводная сеть
SourceAddress     : 192.168.1.73
TcpTestSucceeded  : True

PS C:\Users\Борис>
```

С локального компьютера выполняем проверку доступности сервера с помощью команды PowerShell `Test-NetConnection 91.99.163.16 -Port 443`. В выводе отображается `TcpTestSucceeded : True`, что означает успешное установление TCP-соединения с сервером на порту 443. Это подтверждает, что порт открыт, сервер доступен из интернета и сервис корректно принимает входящие подключения.

Таким образом, мы подтверждаем, что сервис запущен, слушает порт 443 и настроен с корректными криптографическими параметрами для безопасной работы.