Міністерство науки і освіти України

Національний аерокосмічний університет ім. М.Є. Жуковського

*«Харківський авіаційний інститут»*

Факультет №5 «Радіоелектроніки, комп’ютерних систем та інфокомунікацій»

Кафедра 501

«Аерокосмічних радіоелектронних систем»

**Звіт**

**про виконання розрахунково-графічної роботи**

з дисципліни «Інформаційно-телекомунікаційні мережі»

**на тему:**

«Створення власного серверу Minecraft за допомогою сервісів GCP»

Студенти 4-го курсу групи 546

спеціальності 172 – «Телекомунікації та радіотехніка»\_

освітня програма Радіоелектронні пристрої, системи та комплекси

Китайгородський С.Ю

Мащенко О.В

(Прізвище, Ім’я, По-батькові)

Оцінка (кількість балів) **\_\_\_\_\_\_\_**

Перевірив: Перетятько Максим

м. Харків

2021 р.

2021 р.

Завдання на розрахункову роботу

1. Створити новий проект в консолі *GCP*.
2. Розгорнути віртуальну машину з ОС *Debian 10* та певним набором властивостей під необхідний нам сервер.
3. Встановити на віртуальній машині сервер *Minecraft* з найновішою версією гри – *1.16.5*.
4. Провести конфігурацію серверу та налаштування ігрового процесу.
5. Знайти добровольців для того щоб протестувати.
6. Дозволити клієнтам підключатися до нашого серверу.
7. Додати серверу можливість робити резервні копії ігрового світу раз у певний проміжок та зберігати ці копії у спеціальному сховищі.
8. Запустити сервер та вдало підключитися до нього за допомогою його IP-адреси.
9. Згорнути сервер та зайнятися його розвитком, автоматизацією процесів розгортки та згортки серверу, прописавши код з попередніх пунктів (частково) в спеціальні bash-скрипти.

Китайгородський С.Ю

Мащенко О.В

Давайте починати ?

І перше, що необхідно зробити – **створити віртуальну машину**. На трьох картинках нижче ми покажемо, як це зробити:

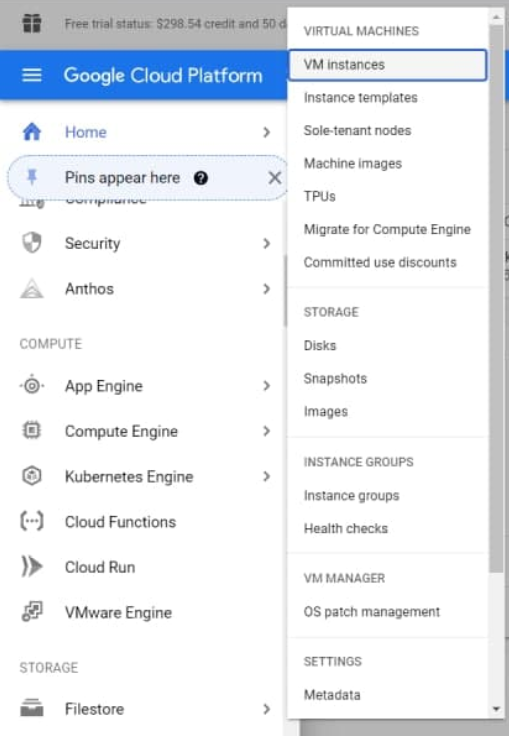


Рисунок 1 – Вікно Google Cloud Platform

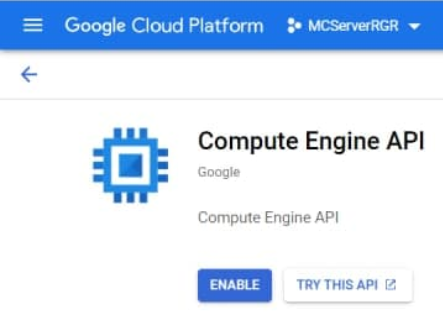


Рисунок 2 – Вікно Google Cloud Platform

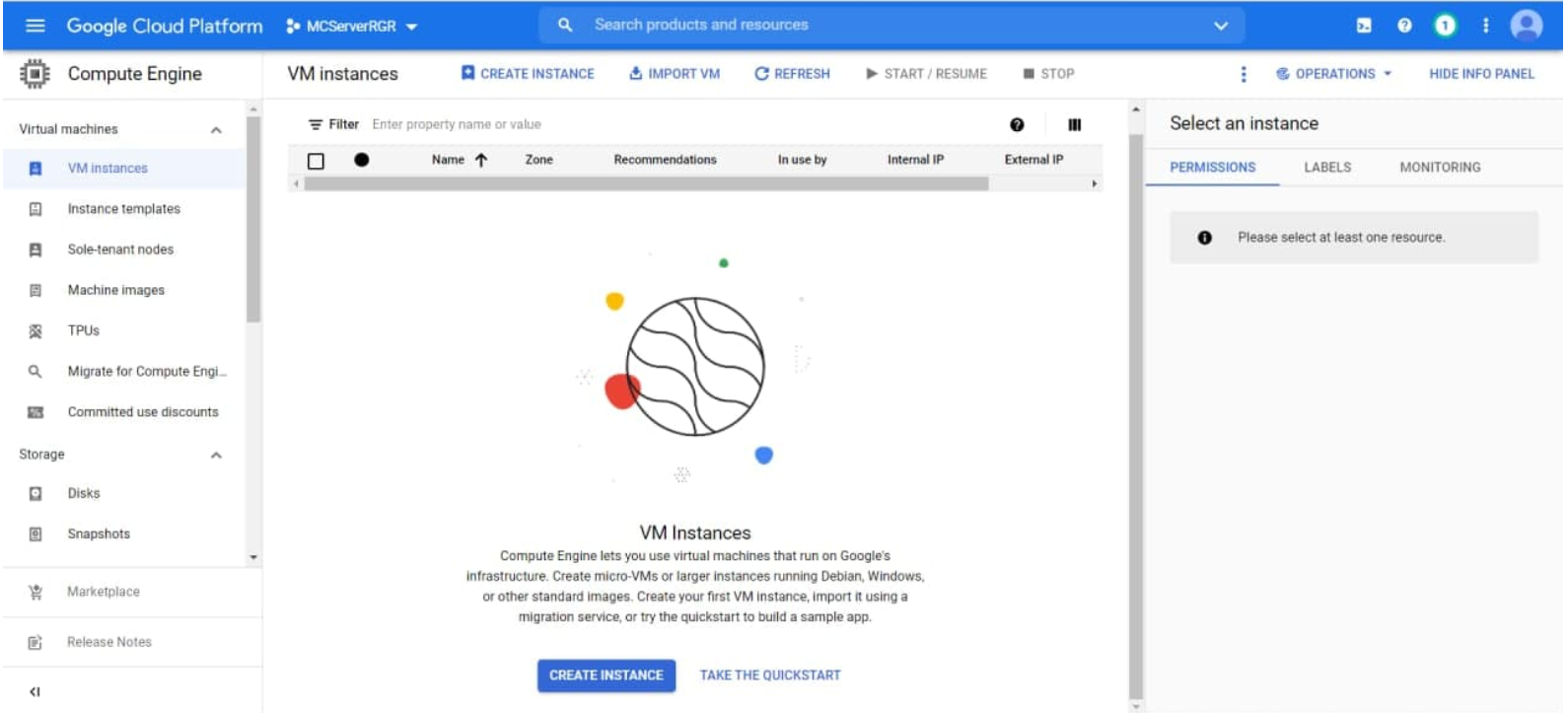


Рисунок 3 – Вікно Google Cloud Platform

тепер тиснемо *CREATE INSTANCE* і почнеться сам процес створення.

Спочатку, потрібно зробити базові налаштування :

дати назву віртуальній машині;

вказати зону її розташування;

вказати тип завантажувального диску;

обрати кількість ядер процесора та оперативної пам’яті;

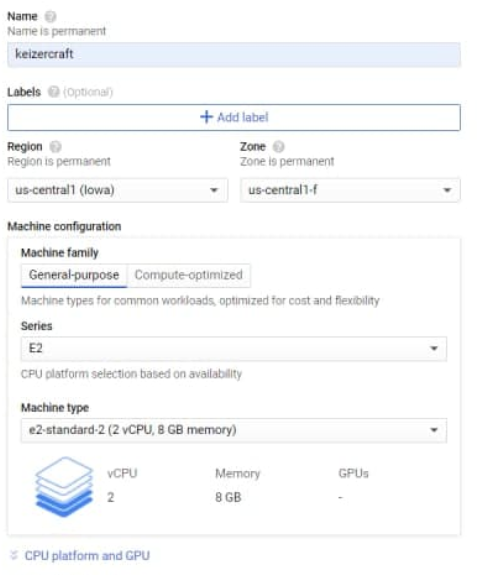


Рисунок 4 – Вікно вибору характеристик віртуальної машини

Далі йде налаштування диску з ОС *Debian 10*:

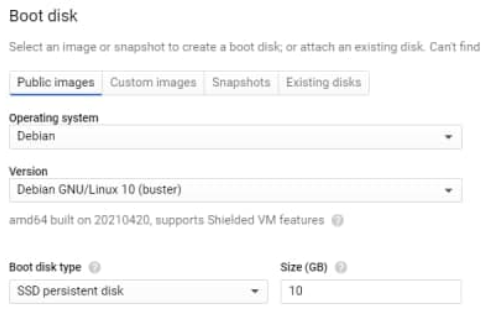


Рисунок 5 – Вікно вибору операційної системі та сховища

Тепер нам потрібно зробити трохи роботи “наперед” – зробивши так, щоб віртуальна машина могла робити резервні копій в хмарному сховищу та навпаки. Це нам ще знадобиться, коли ми далі розглядати питання резервних копій світу.

Це зробити дуже просто – треба змінити два налаштування в розділі *Identify and API access*, як показано на рисунку №6:

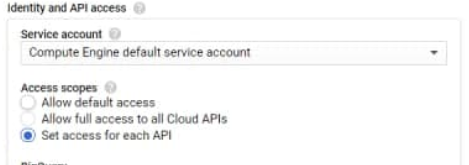




Рисунок 6 – Змінення деяких налаштувань серверу

Ось и прийшов час справжнього проектування серверу.

Почнемо з додання постійного диску до віртуальної машини (далі – ВМ).

* І навіщо нам це потрібно? – мабуть, спитаєте ви нас.

Бо завантажувальний диск, що ми створили хвилину тому, буде прив’язаний до даної конкретної ВМ. И при переході на іншого постачальника вираховувальних потужностей у нас не буде можливості забрати данні з серверу.

Скористаймося для цього постійним *SSD*-диском. Завдяки цьому рішенню, сервер буде працювати набагато стабільніше, ніж міг би, завдяки тому що цей тип диску підтримує дуже великі швидкості операцій вводу-виводу!

Спочатку знаходимо треба знайти напис як на рисунку№7



Рисунок 7 – Вибір типу диска

Необхідно розкрити його, та перейти до розділу *Disks*, та натиснути *Add new disk*.

Спочатку робимо як на цьому малюнку№8

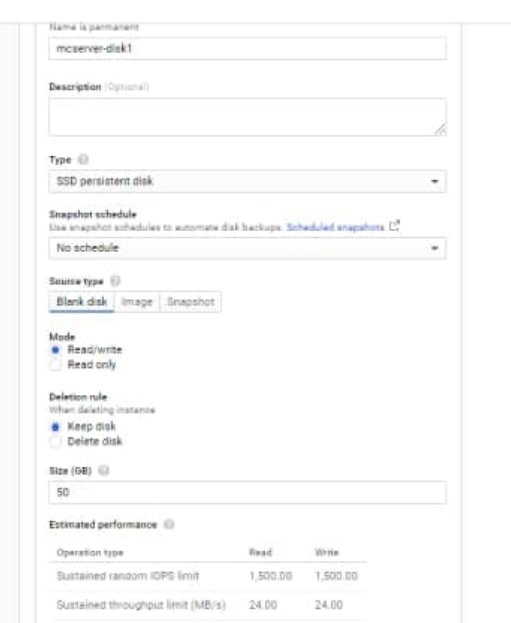


Рисунок 8 – Налаштування диску

Далі можна налаштувати шифрування даних на диску. Ми вирішили розібратися трохи, що таке *Cloud KMS*, а тепер і вам розкажемо на пальцях.

У графі *Encryption* вибираємо галочку *Customer-managed key*. Після цього, треба шифрувальний ключ створити. Тож переходимо туди, де синеньким намальовано:



Рисунок 9 – необхідне посилання

І дозволяємо цьому сервісу трохи попрацювати:

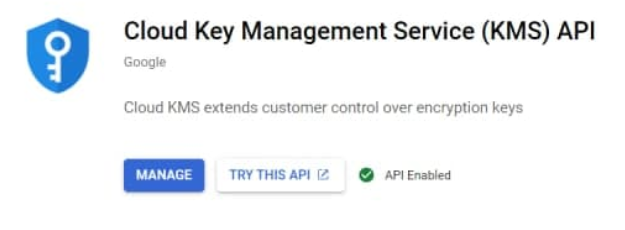


Рисунок 10 – *API* -Cloud key management service

Як уже можна було здогадатися, *Cloud KMS* – це така *API*, що може використовувати спеціальні ключі для шифрування, дешифрування та маркування даних, а також здійсняти керування цими ключами (або групами ключів) та їх організацію за заданими правилами (це зветься *key ring*).

Отже, створимо key ring та ключ у ньому для шифрування даних нашого постійного *SSD*-носія.

Почнемо з *key ring*:

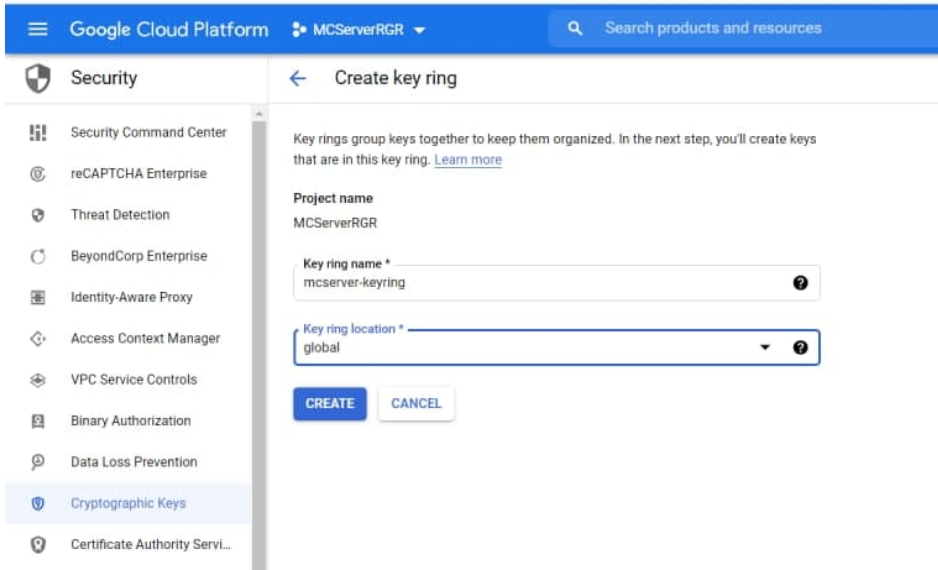


Рисунок 11 – Створення *key ring*

І далі робимо ключ. Без наворотів та зайвих конфігурацій. Просто робимо ключ.

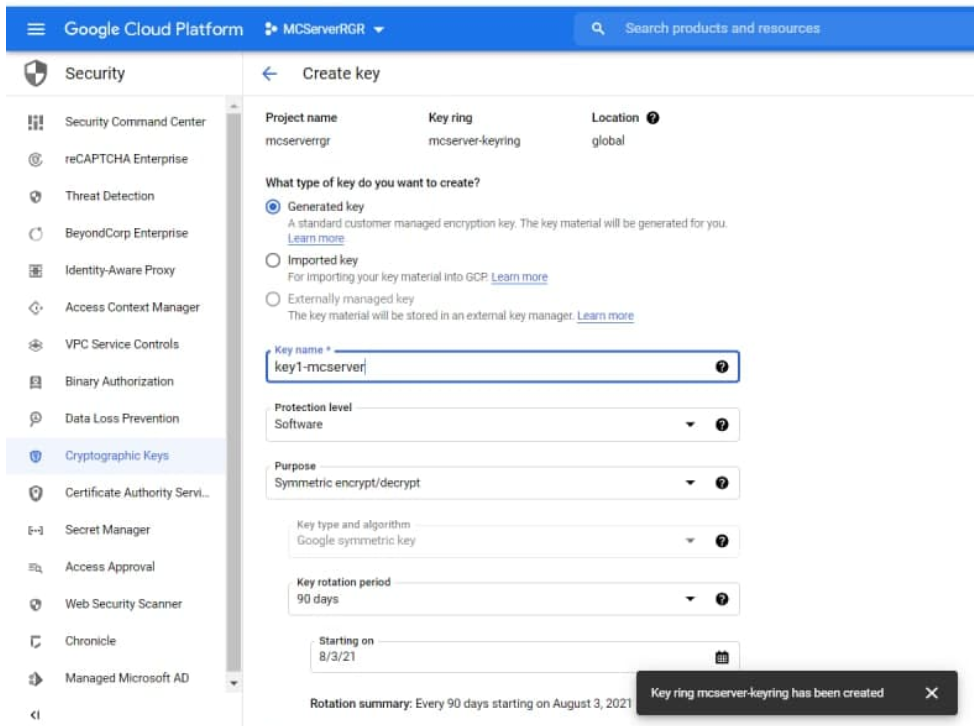


Рисунок 11 – Створення інших ключів

Готово! Вертаємося до конфігурації нашого постійного диску. Якщо оновити сторінку з конфігурацією, все, що ми робили, окрім ключа – злетить. Відновити це тепер займе у вас не більше двох хвилинок, але без цього конфігурація не побачить ключ. Додаємо його:



Рисунок 12 – додаємо ключ

Можна тиснути *Done*. Якщо ми тепер запустимо ВМ, створений диск підтягнеться автоматично.

Тепер перейдемо до розділу *Networking*. Додаємо там тег з назвою, наприклад, *minecraft-server*. Це нам ще знадобиться на етапі встановлення брандмауера, щоб можна було таки зайти на цей сервер.

Також, для надійної пересилки вхідних запитів до ВМ, їй потрібна статична *IP-адреса*. Робимо це ось таким чином:



Рисунок 13 – Додавання тегу

Ну, все! Тепер тиснемо *CREATE*, і наша ВМ готова:

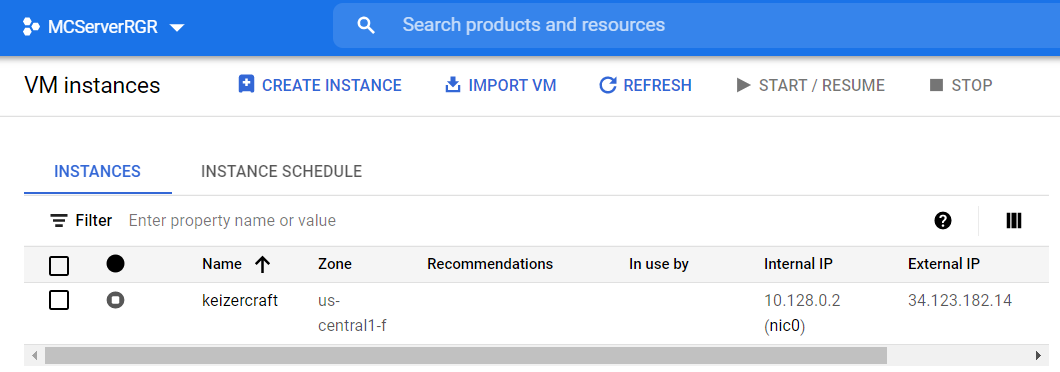


Рисунок 14 – Вікно активних ВМ

Дозволимо клієнтам підключатися до нашого майбутнього серверу. Спочатку робимо те що показано на рисунку №15

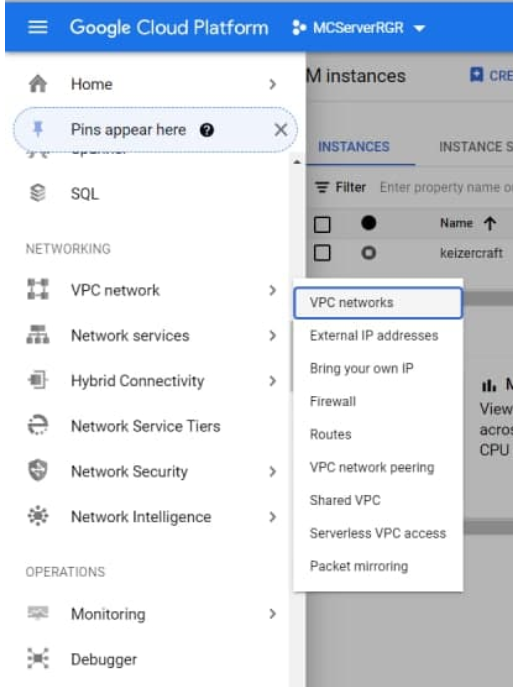


Рисунок 15 – Інструкція

В розділі *VPC Networks* тиснемо на надпис *default*. Гортаємо сторінку трохи вниз, і переходимо до розділу *FIREWALL RULES*.

І знов, робимо все як показано на рисунку №16 та №17

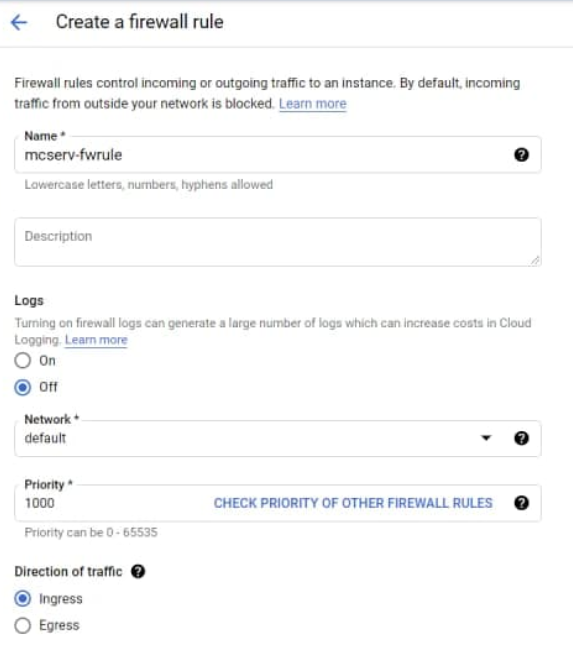


Рисунок 16 – Продовження інструкцій

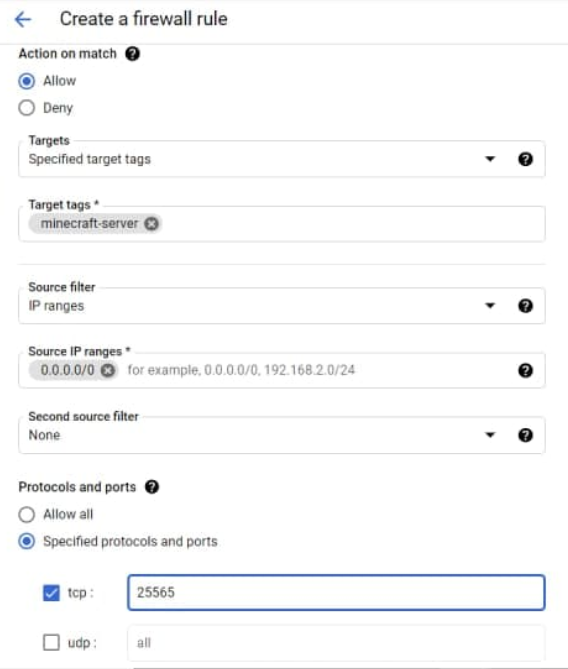


Рисунок 17 – Кінець інструкцій

Тег важливо вказати саме той, що ми зробили на етапі створювання ВМ – інакше ніхто на той сервер на зайде, і ви нічого на користувачах не заробите.

А нам це не потрібно.

По стандарту, *Minecraft* використовує порт 25565 як прослуховуючий. Тому ми і вказали саме цей номер при налаштуванні брандмауера. Це взагалі-то можна й змінити при великім бажанні (у спеціальному файлі *server.properties*, за який поговоримо дещо пізніше). Але поки що усього-на-всього тиснемо *CREATE.* Відтепер, коли ми запустимо сервер, до нашого серверу зможуть підключатися люди за певною адресою. Ця адреса виглядатиме як

*<External IP>:<tcp-port>.*

Наприклад, на прикладі зі скрінів це:

*34.123.182.14:25565.*

Добре, добре, ще тільки початок, а ми вже говоримо за користувачів та їх гроші…настав час сумлінної праці. Гайда до ВМ та командної строки *Linux*!

Спочатку, встановимо SSH-з’єднання зі створеною ВМ. Запускаємо і тиснемо на кнопку SSH у нашій ВМ:

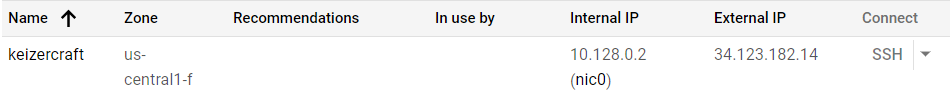


Рисунок 18 – Активна ВМ



Рисунок 19 – Командне вікно ВМ

От ми вже й усередині нашої ВМ!

І спочатку треба підключити до неї наш постійний диск.

Створюємо нову директорію під домашньою директорією ВМ:

sudo mkdir –p /home/mcreal

Вона використовуватиметься як точка зборки постійного диску.

Спочатку форматуємо диск:

sudo mkfs.ext4 –F –E lazy\_itable\_init=0,lazy\_journal\_init=0,discard /dev/disk/by-id/google-<disk name>

В нашому випадку <disk name> - це mcserver-disk1.

І далі збираємо диск:

sudo mount –o discard,defaults

/dev/disk/by-id/google-<disk name> /home/mcreal

Диск офіційно встановлено. Саме тепер ви будете робити те, для чого читали усю цю важливу інформацію!

Сервер *Minecraft* працює на ВМ *Java* (або *JVM*). А значить, він потребує і спеціальне оточення *Java* (що зветься у народі *JRE*). Адже наш сервер не потребує графічний інтерфейс, ми використовуватимемо без заголовкову версію *JRE.* Це рішення допоможе *Minecraft* використовувати більше власних ресурсів для можливого підвищення продуктивності серверу, замість зайвого використання ресурсів *JRE*.

Отже, спочатку оновимо репозиторії нашої ОС за допомогою стандартної команди:

sudo apt-get update

Після цього, встановлюємо беззаголовкову *JRE*:

sudo apt-get install –y default-jre-headless

Не втомилися? Настав час піти зробити собі міцної кави, і при бажанні запалити цибарочку…попереду найцікавіше!

Допили, докурили? Тоді бігом в створену під *Minecraft* директорію:

cd /home/mcreal

Оскільки тепер ця директорія містить зібраний постійний диск, для подальшої роботи потрібен спеціальний доступ, відомий в народі як *root user access.* Отримується він завдяки простій команді:

sudo su

Тепер завантажуємо власне сервер *Minecraft*, використовуючи команду:

curl <https://launcher.mojang.com/v1/objects/1b557e7b033b583cd9f66746b7a9ab1ec1673ced/server.jar>

Ну що, стартуємо сервер вперше!

java –Xms1G –Xmx7G –jar minecraft\_server.1.16.5.jar nogui

І…нічого не відбувається☹ Сервер на чинному етапі просто стартує, робить декілька досить швидких операцій і згортається сам собою. Внаслідок чого це коїться? Зараз дізнаєтесь:

ls –l

Помітили серед невеликої купи елементів файл *eula.txt* ? Це так званий файл ліцензійної згоди кінцевого користувача, яку потрібно прийняти. Заходимо у нього, його потрібно дещо відредагувати:

nano eula.txt

Вже побачили рядок білого кольору? Його зміст потрібно змінити наступним чином:



Рисунок 20 – Приклад необхідних змін

Для зберігання та застосування змін натисніть *Ctrl+O* та далі *Enter*. Для виходу з редактору натисніть *Ctrl+X*.

Далі, налаштуємо конфігурацію нашого ігрового світу. Окрім файлу *eula.txt*, можна ще побачити файл *server.properties*. Перейдемо до його редагування!

nano server.properties

Поки що він є зовсім пустим. За цим посиланням знаходимо базові рекомендації з контенту цього файла та копіюємо їх у редактор: [*https://minecraft.fandom.com/wiki/Server.properties*](https://minecraft.fandom.com/wiki/Server.properties)

Також, за цим посиланням, можна знайти детальний опис властивостей, поданих у цьому файлі. Можете налаштовувати їх як завгодно, але головне – дуже бажано виставити цей флаг:

online-mode=false

Якщо цей флаг не буде виставлений саме таким чином, гравці без платної ліцензії *Minecraft* просто не зможуть зайти на сервер! А у ~~психічно здорових~~  великої більшості людей (особливо серед цільової аудіторії – ~~школоти~~  дуже творчих особистостей) нема тої ліцензії. І в нас її нема, і у вас її теж 100% нема, так що виставляйте той флаг, та йдемо далі.

Отже, якщо ви піднімете сервер просто зараз, він не житиме без чинної активної *SSH*-сесії…тобто, якщо закрити термінал, сервер згорнеться. А нам хочеться, щоб він робив постійно! Рішення цієї проблеми, авжеж, також вже придумали – ми зараз говоримо про таку гарну річ, як *screen*. Ця штука дозволяє вам створити ще один віртуальний термінал, яким можна керувати с точки зору перетворення процесів, що там відбуваються, з фонових на основні та навпаки за допомогою відповідних команд. І якщо наш термінал працює у фоновому режимі, він буде працювати незалежно від того, чи йде просто зараз *SSH*-сесія!

Як і будь-який інструмент, що покращує наше життя, його спочатку треба завантажити:

apt-get install screen

А от тепер, нарешті, урочисто, під ~~гімн СРСР~~ улюблену музику стартуємо сервер у фоновому режимі (терміналі на ім’я mcs):

screen -S mcs java -Xmx8000M -Xms1024M -jar /home/mcreal/minecraft\_server.1.16.5.jar nogui

Приблизно так має виглядати успішний запуск серверу:

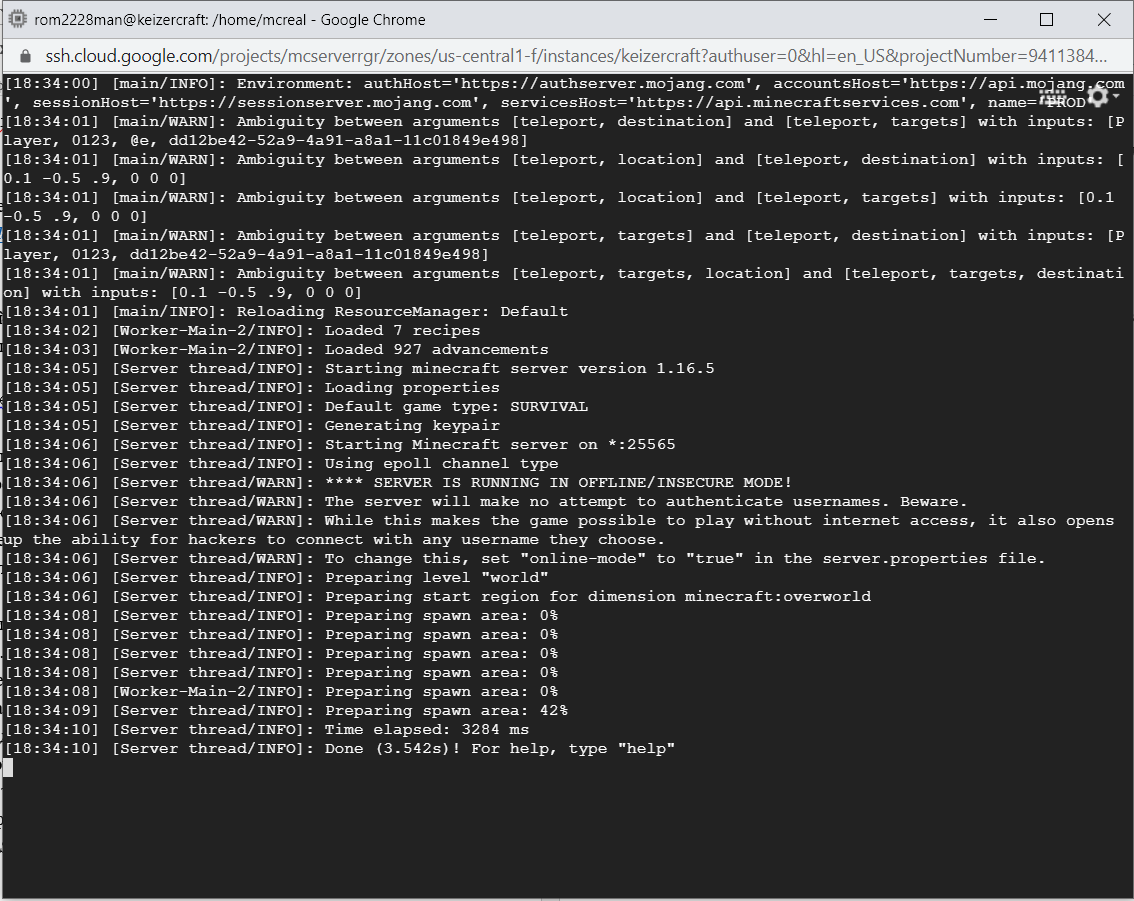


Рисунок 21 – Приклад діалогового вікна при успішному запуску

Щоб залишити цей термінал у фоновому режимі і повернутися до власне командної строки, треба використати комбінацію Ctrl+A, і далі натиснути D.

Щоб повернутися назад до терміналу сервера, треба використати ~~ще одну бісову~~ наступну зручну команду:

screen –r mcs

Тепер у нас є приватний сервер *Minecraft*! Але треба зробити ще дещо дуже важливе – резервне копіювання нашого ігрового світу з певним часовим інтервалом.

За потреби, спочатку треба знов прописати команду для отримання *root access*:

sudo su

І тепер створюємо новий бакет *Google Storage*:

gsutil mb gs://<project-id>-minecraft-backup

ID проекту можна встановити у вікні проектів:

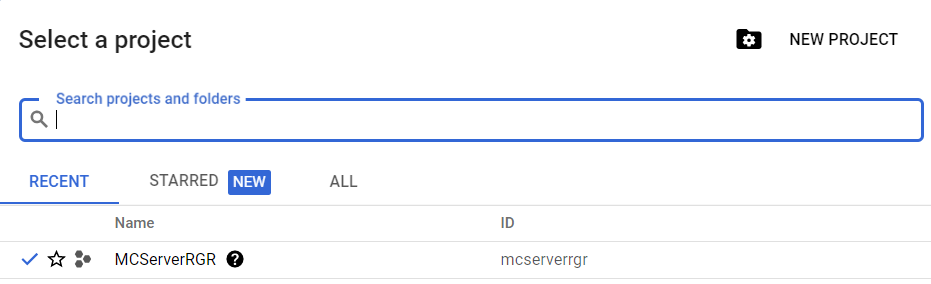


Рисунок 22 – Встановлення ID

Тобто, у нашому випадку, <project-id> - це mcserverrgr.

Далі, у вашій *minecraft*-директорії, створіть наступний *bash*-скрипт:

nano /home/mcreal/backup.sh

У вас має автоматично відкритися редактор щойно створеного файлу.

І от туди з дуже розумним обличчям (тупо та копіпастом) вписуємо ось такий код зображений на рисунку№22

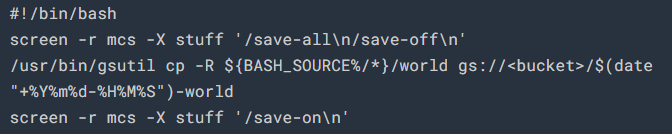


Рисунок 23 – Код який необхідно вставити

Тільки не забудьте замість <bucket> вписати ім’я саме вашого бакету. Будьте обережні зі скриптами. В нашому випадку, наприклад:

<bucket>=mcserverrgr-minecraft-backup

Далі стандартна процедура…*Ctrl+O, Enter, Ctrl+X*.

А як же саме працює цей скрипт? Дещо пояснимо!

Починає він свою роботу з того, що зберігає чинний стан вашого ігрового світу, при цьому призупиняючи можливість цього світу виконувати авто зберігання. Далі він робить резервну копію директорії з даними світу, та зберігає їх у спеціальну директорію у вашому бакеті з маркуванням часу. Нарешті, після виконання резервного копіювання, скрипт знову дозволяє ігровому світові виконувати автозберігання.

Тепер поступово напишіть ці дві команди:

chmod 755 /home/mcreal/backup.sh

/home/mcreal/backup.sh

Ці команди власне запускають щойно написаний скрипт *backup.sh*.

Після того, як скрипт завершить роботу, треба відвідати сторінку:

[*https://console.cloud.google.com/storage/browser*](https://console.cloud.google.com/storage/browser)

Там тепер має з’явитися резервна копія вашого світу. Виглядати буде як щось схоже на те що зображено на рисунку №24

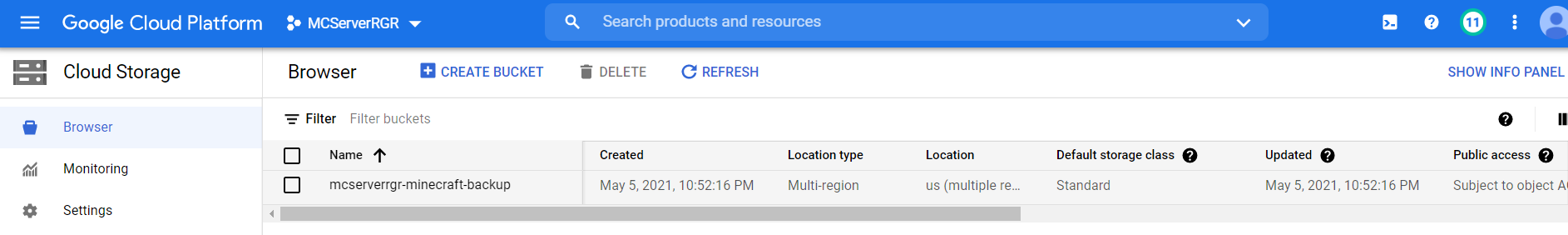


Рисунок 24 – Вдале зберігання світу

Щоб не починати процес резервного копіювання кожного разу власноруч, можна скористатися такою штукою, як *cron job*. Детальніше за посиланням:

[*https://ru.wikipedia.org/wiki/Cron*](https://ru.wikipedia.org/wiki/Cron)

Послідовне виконання наступних дій допоможе вам жити прекрасно, без труднощів та в достатку та зробити так, щоб резервна копія ігрового світу виконувалася, наприклад, раз на 4 години.

Спочатку виконуємо команду:

crontab –e

Гортаємо файл, що відкрився, у самий низ, і вписуємо туди ось таку команду:

0 \*/4 \* \* \* /home/mcreal/backup.sh

І знову *Ctrl+O*, і знову *Enter*, і знову *Ctrl+X*.

Тепер резервне копіювання відбуватиметься автоматично!

Врешті-решт, навчимо вас і тому, як згортати сервер. Це робиться командою:

sudo screen –r –X stuff ‘/stop\n’

Після цього, можно безпечно згортати і саму ВМ.

Ось декілька знімків роботи нашого серверу! Якщо ви ніде не помилилися, у вас має бути готовий до бою сервер!

Ось так виглядає термінал, що працює зазвичай у фоновому режимі

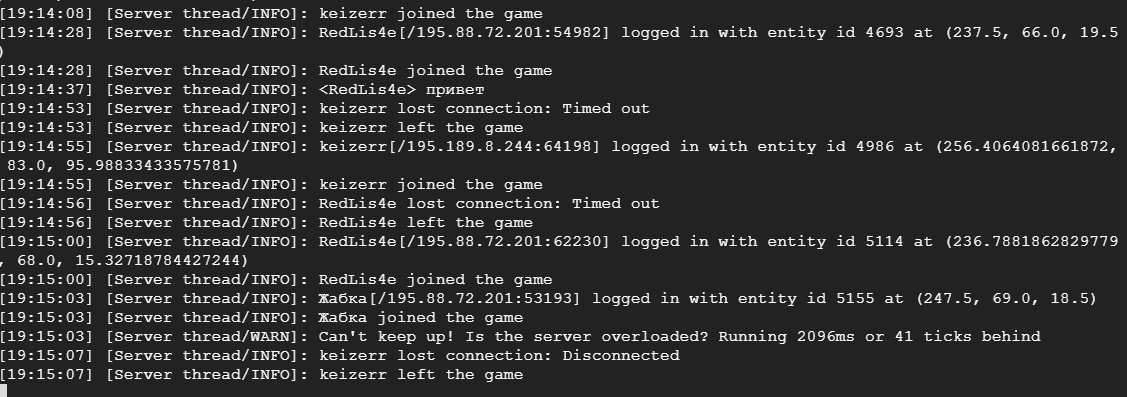


Рисунок 25 –Робота терміналу у фоновому режимі

Ось так виглядає гра біля місця спауну:



Рисунок 26 – Вигляд гри біля місця спавну

Як можна бачити, сервер дійсно працює як сервер – на поданому знімку екрана можна спостерігати присутність трьох гравців!



Рисунок 27 – Гравці онлайн

Тобто, все працює!

А тепер, щодо покращень.

Взагалі-то, якщо ви живете в Україні, і зробили все абсолютно як показано тут, на виході ви отримаєте сервер, на якому грати буде досить складно. Найуважніші з вас могли помітити, що ще на самому початку ми розмістили нашу ВМ…у США! При з’єднанні з таким сервером може статися так, що великий пінг не дозволить вам нормально пограти в гру. Насправді, дуже важливо аналізувати цей момент ще до створення ВМ. Наприклад, у випадку України, частіше за все оптімальним рішенням є розташування ВМ на території Західної Європи.

Далі…якщо постійно згортати/розгортати сервер/ВМ, деякі операції треба буде виконувати кожного разу. Це до питання автоматизації – частково, описані нами команди можна записати в окремі *bash*-скрипти, які далі треба буде виконувати кожного разу при розгортанні/згортанні серверу.

Нарешті, гра стане набагато цікавішою, якщо добавити моди! Наприклад, це може бути наш улюблений *Industrial Craft 2*, або *ThaumCraft*…

Але тут є достатньо підводних каменів. Не всі моди є на найновіші версії *Minecraft.* Той же *Industrial Craft 2* не працює на версіях, новіших од *1.12.2.*

Також, треба встановлювати ядро *Forge* для модів – додаткова морока.

А якщо треба встановити старішу версію гри – дещо складнішим стане процес налаштування *JRE*, адже старі версії гри можуть не подружитися з новою *Java*.

Але всі ці речі того насправді варті! І зараз ми наведемо докази цього.

Авжеж, сервіси GCP – це далеко не єдиний спосіб пограти одночасно з багатьма людьми у *Minecraft*.

Якщо у вас усе ж-таки є ліцензія *Minecraft,* , ви можете придбати таку річ, як *Realm*. Завдяки цьому, компанія-розробник гри (*Mojang*) виступатиме як хостінг вашого серверу та надають вам базовий сет можливостей для сервера (наприклад, міні-ігри). Мінус цього рішення полягає в тому, що на користувачах заробити у такий спосіб нереально багато людей на ваш сервер ніколи не зайде (технологія розрахована на гру від 2 до 10 гравців), да й в цілому ви будете сильно обмежені в своїх діях як володар серверу.

Є також багато сторонніх хостінгів, які (авжеж, за гроші) можуть надати послуги для вашої задумки по ігровим серверам. Це, наприклад, hostinger.ru, myhost.su і т.п. Всі вони також відрізняються вартістю та запропонованими можливостями.

Якщо є фізична можливість зібрати бажану кількість людей в одному місці – тоді і взагалі немає сенсу займатися серверами. Адже що може бути краще, ніж гра по локальній мережі?:)

Є також один сервіс, який є абсолютно безкоштовним і дозволяє зробити сервер на будь-якій версії *Minecraft,* з модами або без, не потребує наявності ліцензії, і навіть також може робити резервні копії! Також він має приємний для користувача інтерфейс на своєму сайті, що дозволяє дуже швидко і зручно змінювати конфігурацію сервера. Насправді, простору для змін там не так і багато (декілька необов’язкових *json*-файлів, вже відомий вам файл *server.properties*), вибір версії безпосередньо гри, додаткових ядер, а також можливість додавати моди. І це все, повторюємося, взагалі без коду – просто натискаючи кнопки на сайті! Фух.

Це ми про *Aternos*! Справді гарна штука, але теж має недоліки.

Перший - на сервері не пограє багато людей зі схожої причини, що була у нас, коли ми вибрали розташування ВМ в Америці. *Aternos* – німецький хостінг, і проблем з ним менше – але він теж не ідеальний. Але у кінцевого користувача немає вибору – за нього все робить хостінг! Тому на практиці, коли ми тестували наш сервер – утрьох гра йшла круто, але коли нас на сервері було четверо – лагало так, що ми подумали, що догралися. Не круто. Тобто, круто, якщо стоїть задача пограти с друзями, але для великих проектів *Aternos* не годиться.

Другий – при запуску сервера іноді приходиться багато чекати, поки прийде твоя черга. Бажаючих зробити безкоштовний простий сервер чимало, і всіх за раз хостінг обслужити не зможе. Іноді чекаєш хвилину, а іноді й 15-20. Кажуть, буває і більше, але ми цього не помічали.

Третій (власноруч не перевірений) – якщо на сервері упродовж великого часу не буде онлайну, він буде автоматично видалений. Не хочеться перевіряти, чесно. Ви зрозумієте, чому не хочеться, побачивши деякі картинки дещо пізніше.

Отже, покажемо вам трохи того *Aternos*, і що ми створювали на нашому сервері ночами за допомогою моду *Industrial Craft 2.*

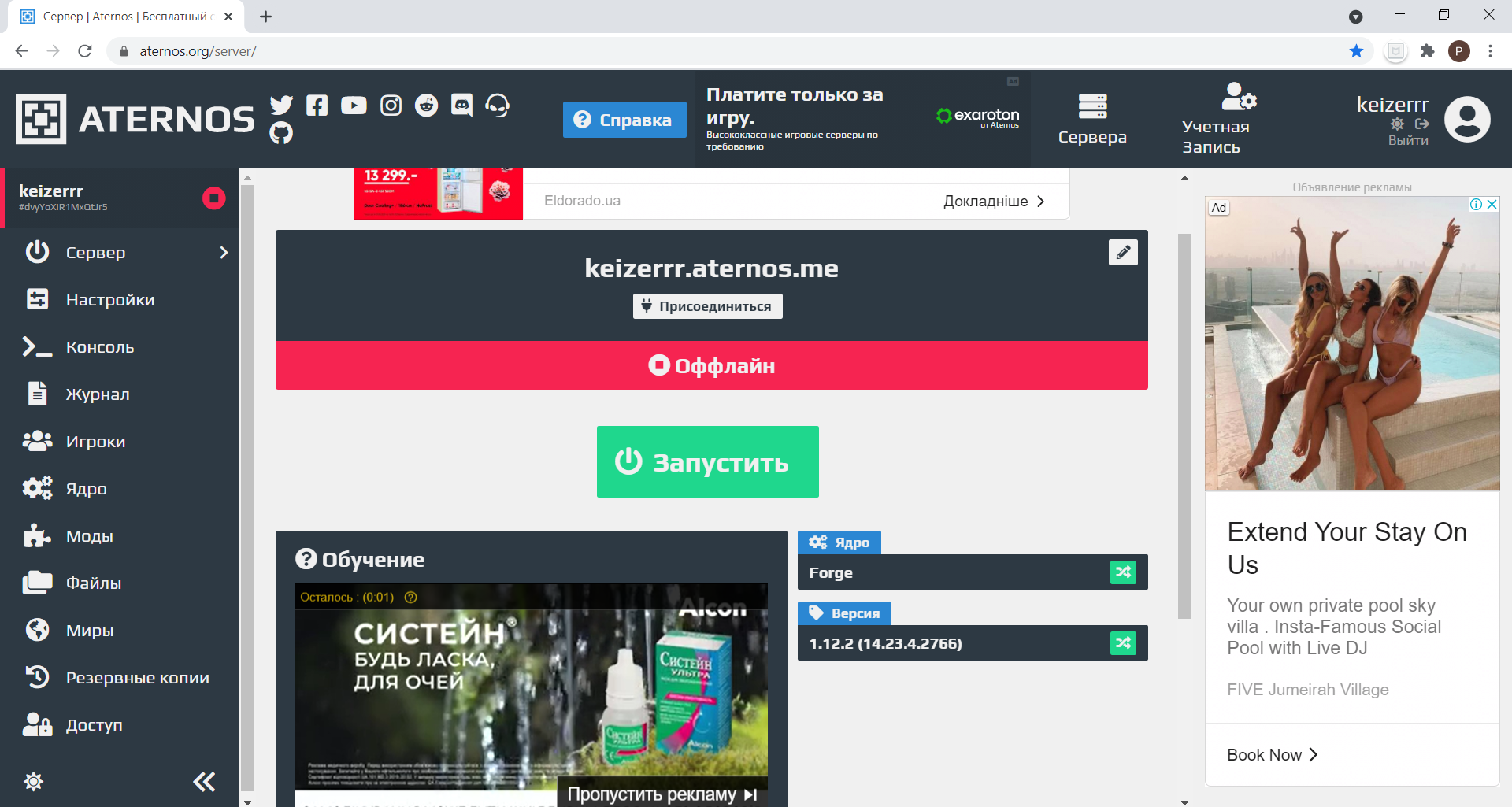


Рисунок 28 – Головне вікно *Aternos*

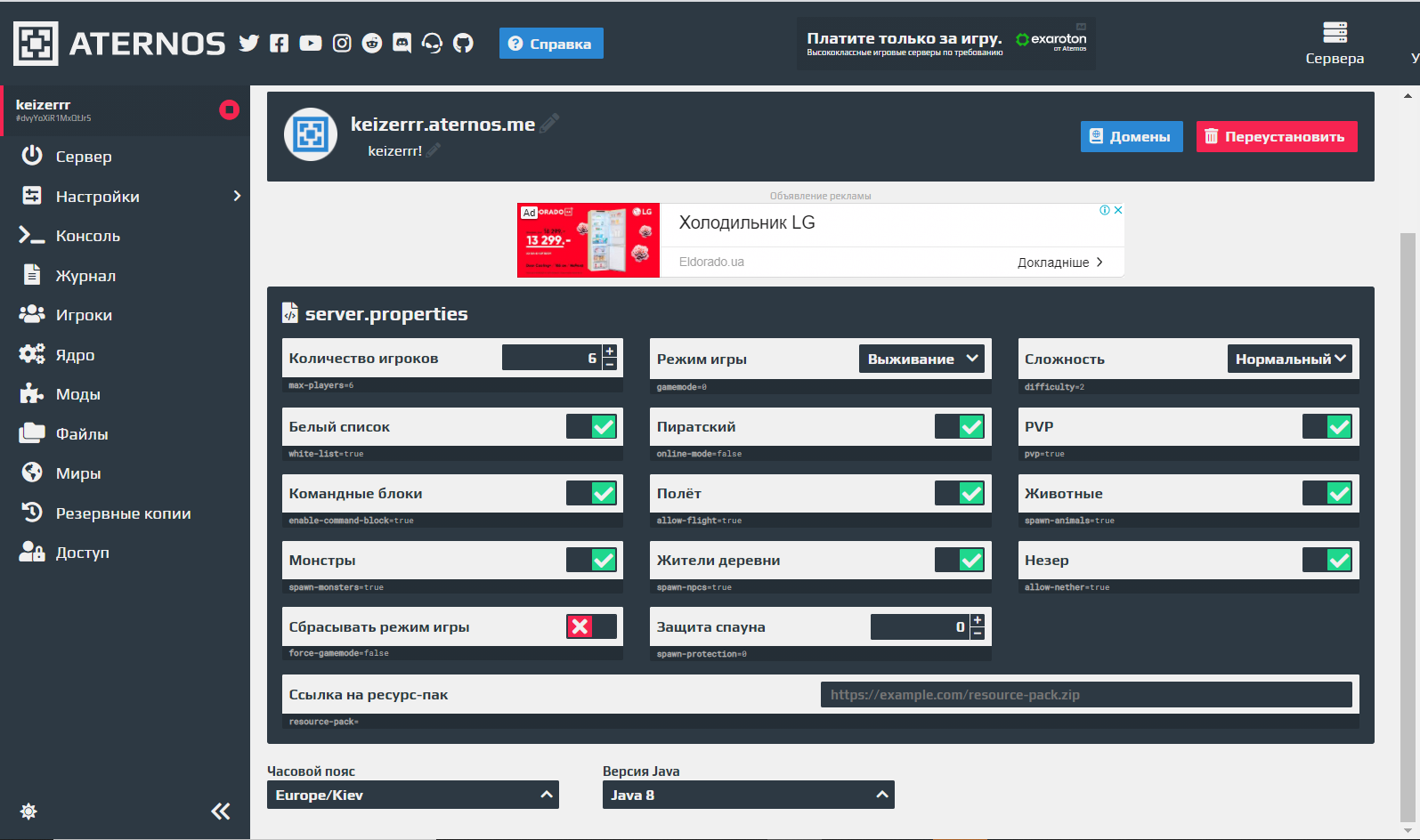


Рисунок 29 – Вікноналаштувань *Aternos*

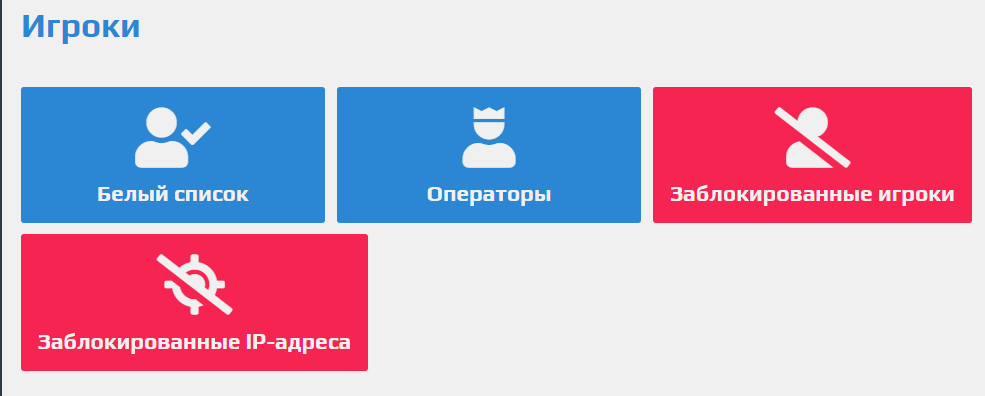




Рисунок 30 – ще одне вікноналаштувань *Aternos*

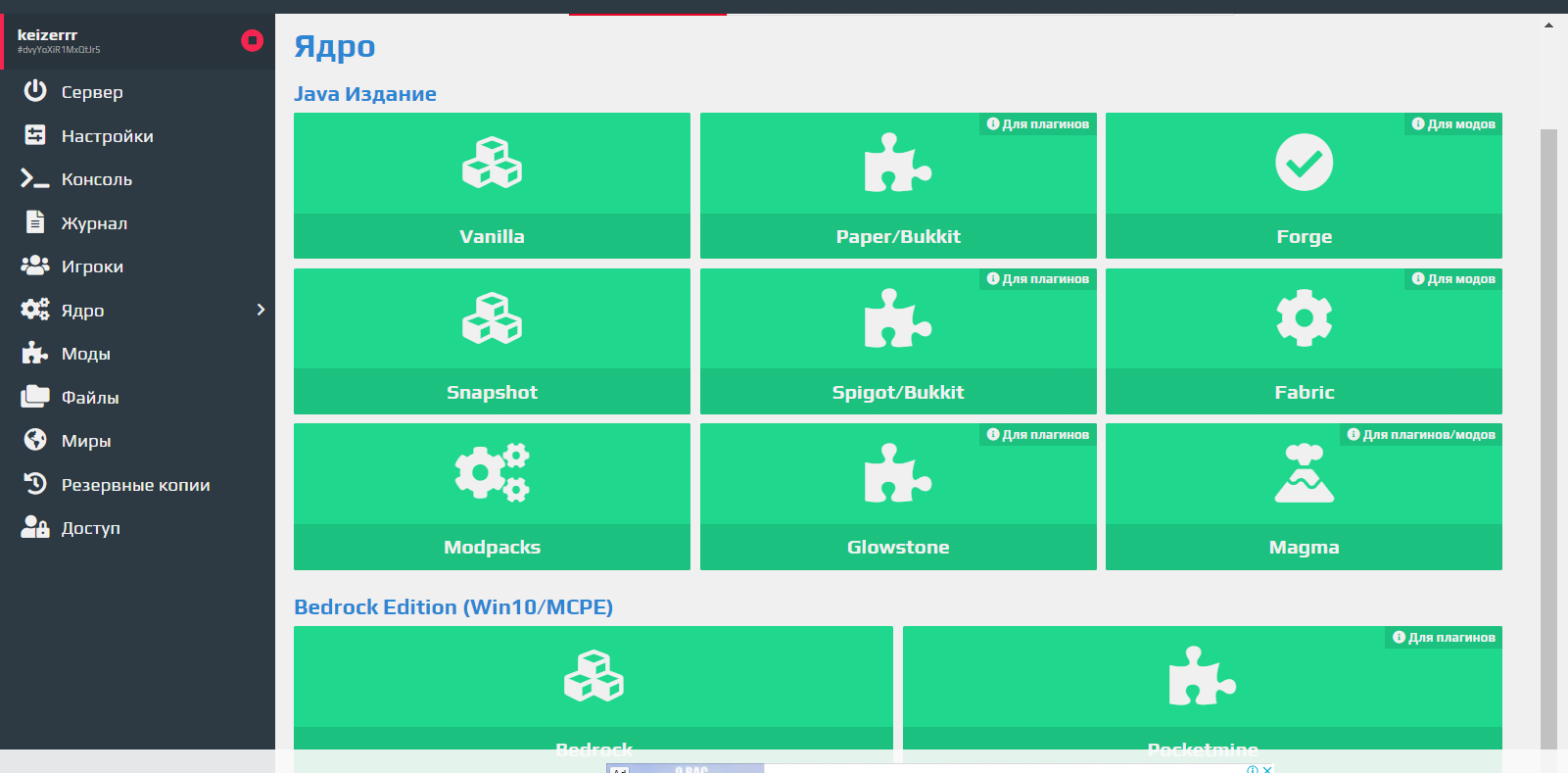


Рисунок 31 – майже останнє вікноналаштувань *Aternos*

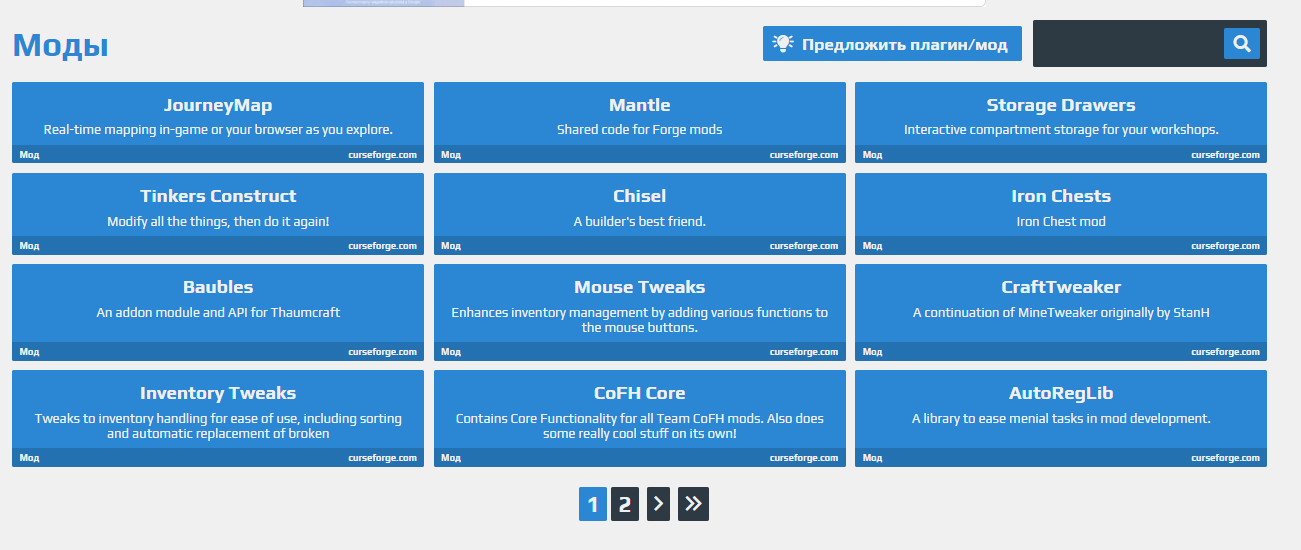


Рисунок 32 – досі не останне вікноналаштувань *Aternos*



Рисунок 33 – перед останнє вікноналаштувань *Aternos*



Рисунок 34 – ось ви і дочекались останнє вікноналаштувань *Aternos*

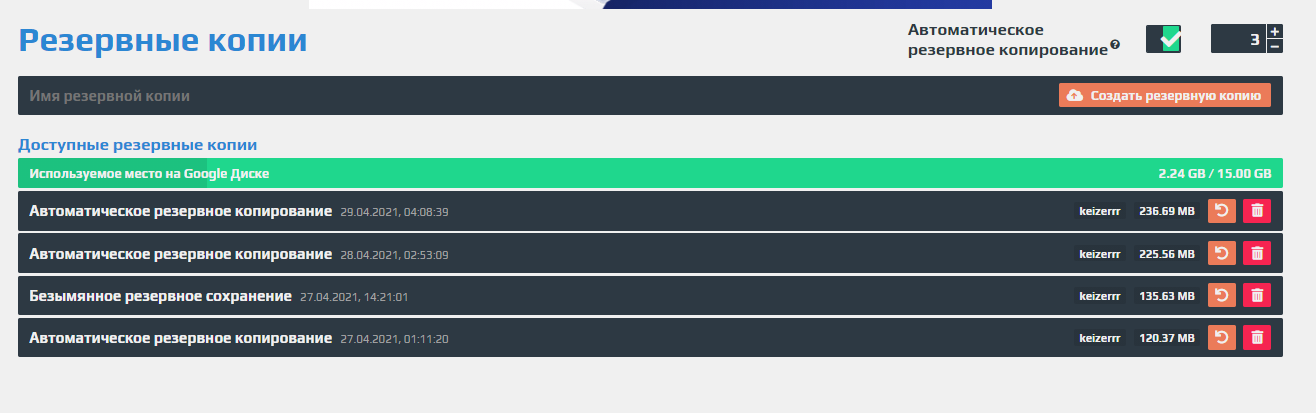


Рисунок 35 – Вікноналаштувань *Aternos*

Ну і ось були подані основні розділи хостінгу. Для встановлення модів потрібно тільки також додати пакети модів у спеціалізовану папку лаунчера – інакше на сервер ви або на зайдете, або зайдете, але не відчуєте усіх його красок.

У *Tlauncher* це робиться так: переходимо до наступної директорії:

*C:\Users\User\AppData\Roaming\.minecraft\mods*

І далі прямо туди додаємо завантажені з відповідних сайтів .jar-файли модів.

Можна грати! Стартуємо сервер, *Aternos* нам самостійно видає статичну IP-адресу. А саме:



Ось так. Це коли тобі 21 рік, ти буцімто на серйозній роботі, все ж-таки доросла людина, а іноді хочеться в куби погратися. З іншого боку, ніхто не заважає комбінувати це з мережевою інженерією…цікаво та корисно!



Рисунок 36 – Результат гри деякий час в найкращу гру століття

Telegram: *@irbars* – пишіть, якщо будуть питання з налаштування серверів, або якщо захочеться пограти – наша команда майже завжди ваша☺

А саме:

*keizerr* – головний ідеолог та автор проекту;

*Никитос* – помічник головного ідеолога;

*RedLis4e* – головний спеціаліст по грі та модах, тестувальник;

*Жабка* – головний з атмосфери гри, тестувальник.

Бажаємо творчих та життєвих успіхів!

ВИСНОВКИ

У роботі були детально розглянуті два способи створення ігрового серверу *Minecraft* – за допомогою сервісів *GCP* та хостінгу *Aternos* – від початку і до готового продукта. Були розглянуті декілька аналогічних рішень даної задачі. Була зазначена велика кількість нюансів при розробці ігрових серверів, та декілька пропозицій з покращення їх роботи. А також була доведена їх працездатність і продемонстровані успіхи в грі – запрошуємо долучитися до процесу!

І наприкінці зазначимо – ми на 100% певні, що ХАІ такої РГР ще не бачив. Час настав!