УДК 004.056.3
DOI https://doi.org/10.32689/maup.it.2023.3.2

Микола ВАСИЛЕНКО
доктор фізико-математичних наук, доктор юридичних наук, професор, професор кафедри кібербезпеки, Національний університет «Одеська юридична академія», вул. Рішельєвська, 28, Одеса, Україна, індекс 65011 (vasylenko.it@journals.maup.kiev.ua)

ORCID: 0000-0002-8555-5712

Валерія СЛАТВІНСЬКА
викладач кафедри кібербезпеки, Національний університет «Одеська юридична академія», вул. Рішельєвська, 28, Одеса, Україна, індекс 65011, Україна, індекс 65000 (slatvinskaya_valeriya@ukr.net)

ORCID: 0000-0002-6082-981X

Світлана СИСОЄНКО
кандидат технічних наук, доцент, доцент кафедри інформаційної безпеки та комп'ютерної інженерії, Черкаський державний технологічний університет, бульв. Шевченка, 460, Черкаси, Україна, індекс 18006 (s.sysoienko@chdtu.edu.ua)

ORCID: 0000-0002-0009-337X

Nikolai VASILENKO
Doctor of Physical and Mathematical Sciences, Doctor of Law, Professor, Professor at the Department of cybersecurity, National University "Odessa Law Academy", 28 Richelevskaya str., Odessa, Ukraine, postal code 65011 (vasylenko.it@journals.maup.kiev.ua)

Valeriia SLATVINSKA
Assistant Professor at the Department of Cybersecurity, National University "Odessa Law Academy", 28 Richelevskaya str., Odessa, Ukraine, postal code 65011 (slatvinskaya_valeriya@ukr.net)

Svitlana SYSOIENKO
Candidate of Technical Sciences, Associate Professor, Associate Professor at the Department of Computer Information Systems and Technologies, Cherkasy State Technological University Shevchenko blvd., 460, Cherkasy, Ukraine, postal code 18006, (s.sysoienko@chdtu.edu.ua)

Бібліографічний опис статті: Василенко, М., Слатвінська, В., Сисоєнко, С. (2023). Особливості створення сценарію для оболонки bash при створенні резервних копій. *Інформаційні технології та суспільство*, 3, 15–22. DOI: https://doi.org/10.32689/maup.it.2023.3.2

Bibliographic description of the article: Vasilenko M., Slatvinska V. & Sysoienko S. (2023). Osoblyvosti stvorennia stsenariiu dlia obolonky bash pry stvorenni rezervnykh kopii [Features of creating a script for the bash shell when creating backups]. *Informatsiini tekhnolohii ta suspilstvo – Information technology and society*, 3, 15–22. DOI: https://doi.org/10.32689/maup.it.2023.3.2

ОСОБЛИВОСТІ СТВОРЕННЯ СЦЕНАРІЮ ДЛЯ ОБОЛОНКИ BASH ПРИ СТВОРЕННІ РЕЗЕРВНИХ КОПІЙ

Анотація. *У даній статті пропонується сценарій на мові bash для створення резервних копій файлів і каталогів. Взаємодія користувача з операційною системою здійснюється через оболонку. В Linux існує ряд різних оболонок, найпопулярнішої з яких є саме bash. Створення резервної копії даних надає можливість виконати відновлення інформації при втраті оригіналу, з якого було створено резервну копію. При цьому під втратою треба розуміти настання події, що призвела до зміни даних, після чого вони втратили цінність або були видалені з носія. Сценарій має наступні функції: Сценарій приймає параметри командного рядка, що є іменами каталогів, які треба додати до резервної копії. Командний рядок може стати ідеальним інструментом для забезпечення кібербезпеки. Неймовірна гнучкість і абсолютна доступність перетворюють стандартний інтерфейс командного рядка на фундаментальне рішення, яке надає швидко створювати та моделювати складні функції за допомогою лише одного рядка конвеєрних команд. Якщо параметри не задано, створюється резервна копія домашнього каталогу. Перевіряється наявність каталогу "archives" у домашньому каталозі користувача. Якщо його немає, він створюється. Отримується поточна дата і час для використання у назві архіву. Якщо параметри командного рядка задано, додавання вказаних каталогів до резервної копії. Перевіряється наявність каталогу перед додаванням. Навички та уміння ефективно працювати з командним рядком – найважливіша навичка для фахівців з безпеки та адміністрування. Створюється резервна копія кожного каталогу, включаючи його підкаталоги, за допомогою команди tar. Виводиться повідомлення про створену резервну копію разом зі шляхом, за яким вона зберігається. Стаття також надає інструкції щодо перевірки*

працездатності скрипта та налаштування періодичного запуску за допомогою cron і ефективності використання командного рядка для поліпшення наявного функціоналу.

Ключові слова: *скрипт, копіювання, командний рядок, резервна копія.*

FEATURES OF CREATING A SCRIPT FOR THE BASH SHELL WHEN CREATING BACKUPS

Abstract. *This article provides a bash script for creating backup copies of files and directories. The user interacts with the operating system through a shell. In Linux, there are a number of different shells, the most popular of which is bash itself. Creating a backup copy of data provides an opportunity to restore information in case of loss of the original from which the backup copy was created. At the same time, loss should be understood as the occurrence of an event that led to a change in data, after which they lost their value or were deleted from the medium. The script has the following functions: the script accepts command line parameters, which are the names of the directories to be added to the backup. The command line can be an ideal tool for ensuring cybersecurity. Unbelievable flexibility and absolute accessibility transform the standard command-line interface into a fundamental solution that allows you to quickly create and model complex functions using only one line of pipelined commands. If no parameters are specified, a backup copy of the home directory is created. If no parameters are specified, a backup copy of the home directory is created. If it does not exist, it is created. The current date and time are obtained to be used in the archive name. If command line parameters are specified, the specified directories are added to the backup. The existence of the directory is checked before adding it. The ability to work effectively with the command line is an essential skill for security and administration professionals. Each directory, including its subdirectories, is backed up using the tar command. A message about the created backup copy is displayed along with the path in which it is saved. The article also provides instructions on how to test the functionality of the script and how to set up a periodic run using cron and how to use the command line effectively to improve existing functionality.*

Key words: *script, copy, command line, backup copy.*

Актуальність проблеми. Розвиток інформатизації у суспільстві останніх років висунув на одне з перших місць проблему захисту величезної кількості інформації, що формується, оброблюється і передається в комп'ютерних системах, а також створення копії даних з носіїв за допомогою резервного копіювання або (бекап) та призначений для відновлення цих даних у разі їх пошкодження або видалення щодо різних дестабілізуючих факторів.

Основні причини створення резервних копій включають:

Захист від втрати даних: Резервні копії дозволяють відновити важливі файли та дані у випадку їх втрати, помилкового видалення або пошкодження. Це особливо важливо для бізнесу та користувачів, які зберігають цінні дані.

Відновлення системи: Резервні копії допомагають відновити систему у разі непередбачених ситуацій, таких як аварії жорсткого диска, атаки зловмисників або вірусів.

Міграція та переміщення даних: Резервні копії дозволяють легко переміщувати дані між різними середовищами або системами. Вони також корисні для міграції на нову апаратну або програмну інфраструктуру.

Відновлення попередніх версій: Резервні копії дозволяють повернутися до попередніх версій файлів або каталогів, які були змінені або видалені. Це може бути корисним у випадку помилкових змін або потреби відновити стару версію.

Забезпечення безпеки даних: Резервні копії є важливим елементом стратегії захисту даних. Вони дозволяють уникнути втрати важливих даних через випадкові або навмисні дії.

Одним із способів полегшити цю проблему з технічної точки зору є корисним інструментом вивчення мови bash. Без цього системні адміністратори та фанати терміналів опинилися б на терміналі годинами. Замість цього ми можемо писати сценарії bash для автоматизації Linux. Навчившись використовувати можливості Bash, ми можемо писати всі ці складні операції та швидко запускати їх за допомогою сценарію.

Оболонка чудова, досліджуючи її, можемо запропонувати шляхи автоматизації та покращення внутрішньої роботи системи Linux.

Аналіз останніх досліджень і публікацій. Для стабільної роботи компаній які працюють з інформацією є процес резервного копіювання даних. Не залежно від того, чи це банк, чи сайт по продажам господарських товарів, у разі несистемного збою та пошкодження основних серверів або баз даних –в найкоротший термін, системна помилка повинна бути виправлена а резервні дані відновлені. *Враховуючи активний розвиток шкідливого ПЗ, що випереджає розвиток антивірусів, найбільш раціонально будувати IT-безпеку навколо системи резервного збереження інформації. Замість того, щоб фокусуватися виключно на запобіганні атакам та боротьбі з вірусами, набагато простіше, дешевше і легше підняти систему з резервних копій* [1]. Авторами роботи [2] спроектовано інтелектуальну систему дедублікації та розподілу даних у хмарних сховищах, яка при доробці систем перевір-

ки цілісності резервних копій та оптимізації дискового простору після видалення старих резервних копій може використовуватись на практиці. У сфері IT оболонка – це програмне забезпечення, яке з'єднує користувача з комп'ютером. Оболонка дозволяє користувачеві дізнатися про систему або файли в ній або керувати системою. Оболонка зазвичай є частиною операційної системи. Існує два типи оболонок.

Командно-орієнтована оболонка, наприклад, OS X термінал.

Оболонка з графічним інтерфейсом користувача, наприклад шукач для OS X або всім знайома Windows.

Термінал є командно-орієнтованою оболонкою, тому що тут ми вводимо команди безпосередньо, а не натискаємо кнопки миші або вводимо інформацію в поля. Коли ми знаходимося на терміналі, у нас працює процес Bash, який надає нам оболонку Bash. Якщо ми починаємо виконувати сценарій, він фактично не виконується в цьому процесі, а замість цього запускає новий процес для виконання всередині. Командно-орієнтовані оболонки також мають безліч різновидів – оболонка Bourne, Bash, оболонка Z тощо. Як стандарт на комп'ютерах Mac, термінал відкриває оболонку Bash. Bash є дуже потужним, оскільки він може спростити певні операції, які важко ефективно виконати за допомогою графічного інтерфейсу користувача. Користуючись підтримкою численних прихильників програмного забезпечення з відкритим кодом, Linux набула великої популярності, а разом з тим – і значного технологічного удосконалення [3]. В Linux існує ряд різних оболонок, найпопулярнішої з яких є bash. Оболонка bash стала ідеальним інструментом для операцій із забезпечення безпеки, оскільки вона має міжплатформні методи і сценарії. Поширеність bash також дає певну перевагу фахівцям, які тестують на стійкість до атак і вторгнень, оскільки в багатьох випадках їм не потрібно встановлювати в системі додаткову інфраструктуру підтримки або інтерпретатор [4 – 5].

У [6-7] автори діляться деякими корисними інструментами та ресурсами, які можуть допомогти у вивченні командного інтерпретатора. Написання сценаріїв для оболонки – це потужна та універсальна навичка, яка може допомогти автоматизувати завдання, маніпулювати даними та налаштовувати інтерфейс командного рядка (CLI). Незалежно від того, чи використовується Bash, Zsh або інша оболонка.

Робота [8] охоплює програмування в командному рядку з акцентом на Linux в командному рядку Bash; що забезпечує достовірну, реальну актуальність, а також надає гнучкі інструменти для негайного початку роботи та вирішення різноманітних реальних завдань. Робота [9] присвячена багатьом поширеним програмам, що використовуються у командному рядку, а також більш складним темам. З огляду на коло досліджень, знаня загального синтаксису bash та вміня писати на ньому – критичний скіл для розробника. Дані дослідження обов'язково станутьв нагоді як початківцям, так і досвідченим розробникам, які могли забути деякі особливості цієї надпотужної командної оболонки.

Метою статті є створення сценарію для оболонки Bash, який автоматизує процес створення резервних копій файлів і каталогів. Основна мета такого сценарію полягає в забезпеченні збереження важливих даних шляхом регулярного створення резервних копій.

Виклад основного матеріалу. Напишемо сценарій для оболонки bash для створення резервних копій згідно таких вимог:

Сценарій приймає довільну кількість параметрів командного рядка. Якщо жодного з параметрів не задано, повинна створюватись резервна копія (tar.gz або tar.bz2) домашнього каталогу користувача. Якщо параметри задані, вони розглядаються як імена каталогів, які треба додати до резервної копії (при цьому домашній каталог у цілому за умовчанням не додається).

Якщо імена каталогів задані параметрами командного рядка, слід додати перевірку їх наявності. Сценарій повинен повідомляти про відсутність певних каталогів, але продовжувати роботу з архівації інших каталогів із списку.

Архів повинен створюватись у підкаталозі archives домашнього каталогу. Сценарій повинен перевірити наявність цього каталогу, і якщо його немає, створити його. Ім’я створюваного архіву повинно містити поточну дату у форматі YYYYMMDDHHMM.

Сценарій повинен коректно відпрацьовувати помилки, такі як некоректні імена каталогів і помилки доступу (зокрема, відсутність права доступу до певних файлів). При цьому сценарій повинен видавати діагностику помилок [3, с. 50]. Перша частина сценарію– це шебанг (#!). Ця послідовність дозволяє скрипту повідомляти інтерпретатору, що він повинен використовувати для розуміння коду. Далі напишемо коментар. Це дозволить кожному, хто використовує сценарій, зрозуміти, для чого призначений код. Коментарі можна додати до скрипту, поставивши символ # [10].

Результати досліджень. Для вирішення завдання напишемо скрипт на мові bash:

```
#!/bin/bash

# Перевірка наявності каталогу "archives" у домашньому каталозі
archive_dir="$HOME/archives"
if [[ ! -d $archive_dir ]]; then
  mkdir $archive_dir
fi

# Отримання поточної дати та часу
current_date=$(date +"%Y%m%d%H%M")

# Якщо параметри командного рядка не задано, створення резервної копії домашнього каталогу
if [[ $# -eq 0 ]]; then
  backup_filename="$archive_dir/home_backup_$current_date.tar.gz"
  tar -zcf $backup_filename -C $HOME .
  echo "Створено резервну копію домашнього каталогу: $backup_filename"
  exit 0
fi

# Якщо параметри командного рядка задано, додавання вказаних каталогів до резервної копії
for dir in "$@"; do
  # Перевірка наявності каталогу
  if [[ ! -d $dir ]]; then
    echo "Помилка: Каталог '$dir' не існує. Пропускаю..."
    continue
  fi

  # Отримання імені каталогу
  dir_name=$(basename "$dir")

  # Створення резервної копії каталогу
  backup_filename="$archive_dir/$dir_name"_"$current_date.tar.gz"
  tar -zcf $backup_filename -C $dir .
  echo "Створено резервну копію каталогу '$dir': $backup_filename"
done
```

Цей скрипт отримує параметри командного рядка, які є іменами каталогів, які потрібно додати до резервної копії. Якщо параметри не задано, створюється резервна копія домашнього каталогу користувача. Сценарій перевіряє наявність каталогів, повідомляє про відсутність деяких каталогів, але продовжує створювати резервну копію інших каталогів зі списку. Резервні копії зберігаються у каталозі "archives" у домашньому каталозі користувача з іменем, що містить поточну дату та час.

Для налаштування періодичного запуску скрипта за допомогою cron, ви можете відкрити cron-редактор командою **crontab -e** і додати наступний рядок:

0 0 * * * /шлях/до/скрипта.sh

Цей рядок вказує, що скрипт має бути запущений щодня о 00:00. Ви можете змінити ці значення відповідно до своїх потреб. Замість **/шлях/до/скрипта.sh** вкажіть повний шлях до файлу скрипта. Після збереження змін у cron-редакторі, cron почне виконувати скрипт зазначеним розкладом.

Щоб перевірити, чи працює скрипт резервного копіювання, було виконано наступні кроки:

1. Збережіть скрипт у файл з розширенням**.sh** (наприклад, **backup_script.sh**) і надайте йому права на виконання, використовуючи команду **chmod +x backup_script.sh**.

2. Запустіть скрипт, вказавши параметри командного рядка (якщо необхідно) або без них. Наприклад:

./backup_script.sh # Без параметрів – створення резервної копії домашнього каталогу;

./backup_script.sh dir1 dir2 # 3 параметрами – додавання вказаних каталогів до резервної копії.

На рисунку 1 наведено фрагмент виконання цих кроків в Git Bash на Windows і в результаті маємо наступне:

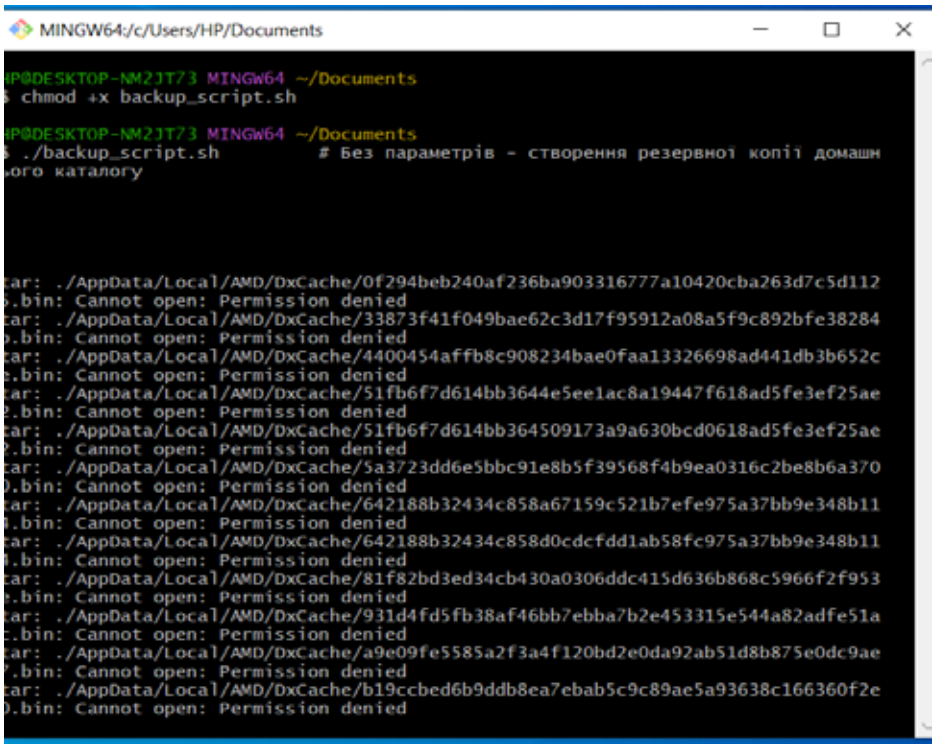

Рис. 1. Фрагмент виконання скрипту резервного копіювання

Після запуску скрипту з'явилося повідомлення про створення резервної копії та шлях, за яким зберігається архів (рисунок 2).

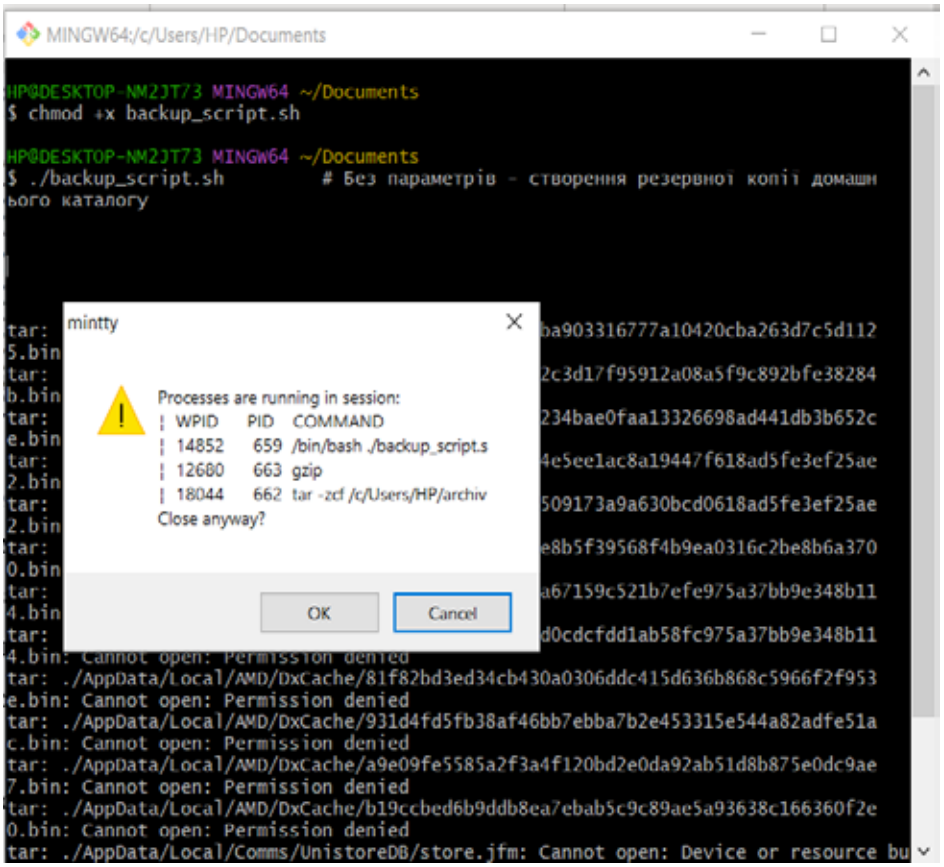

Рис. 2. Фрагмент створення та зберігання резервної копії

Потім було перевірено каталог "archives" у власному домашньому каталозі. Там з'явились нові архіви, які створив скрипт.

Архів містить необхідні файли та каталоги (рисунок 3), і їх можна відновити при необхідності.

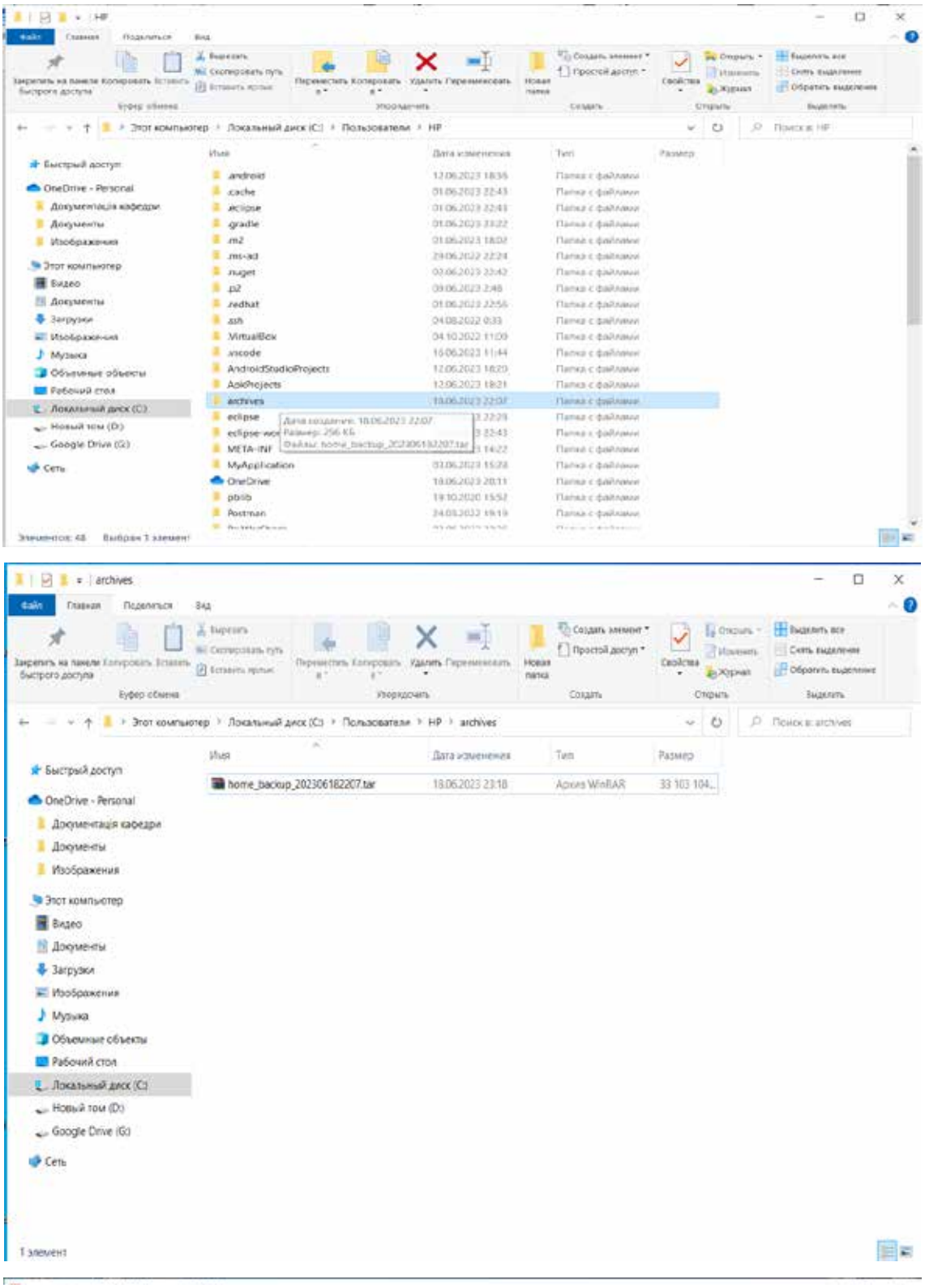

Рис. 3. Архів файлів та каталогів створених скриптом

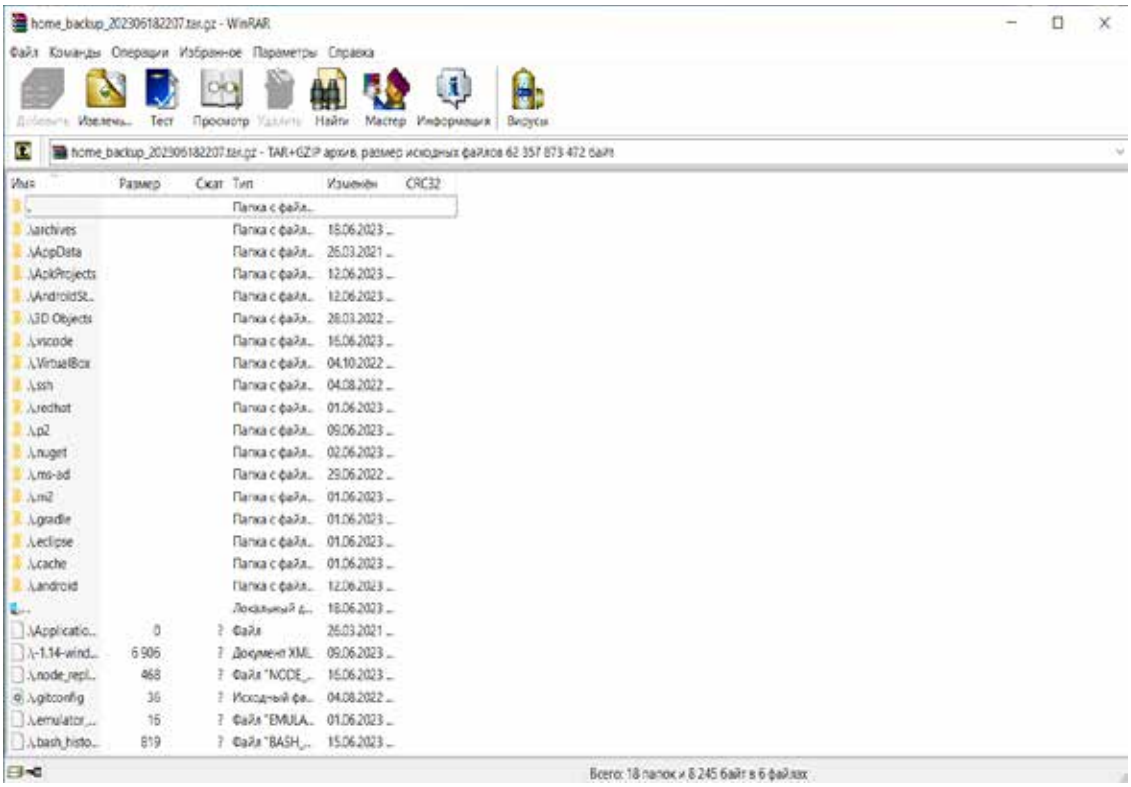

Рис. 3 (закінчення)

Обговорення результатів. Отже, сценарій для оболонки Bash для створення резервних копій має на меті автоматизувати процес створення резервних копій файлів та каталогів. Основна мета такого сценарію – забезпечити збереження важливих даних шляхом регулярного створення резервних копій.

Основні причини використання такого сценарію:

1. Захист від втрати даних: Резервні копії дозволяють відновити важливі файли та дані у випадку їх втрати, помилкового видалення або пошкодження. Це особливо важливо для бізнесу та користувачів, які зберігають цінні дані.

2. Відновлення системи: Резервні копії допомагають відновити систему у разі непередбачених ситуацій, таких як аварії жорсткого диска, атаки зловмисників або вірусів.

3. Міграція та переміщення даних: Резервні копії дозволяють легко переміщувати дані між різними середовищами або системами. Вони також корисні для міграції на нову апаратну або програмну інфраструктуру.

4. Відновлення попередніх версій: Резервні копії дозволяють повернутися до попередніх версій файлів або каталогів, які були змінені або видалені. Це може бути корисним у випадку помилкових змін або потреби відновити стару версію.

5. Забезпечення безпеки даних: Резервні копії є важливим елементом стратегії захисту даних. Вони дозволяють уникнути втрати важливих даних через випадкові або навмисні дії.

Висновки та перспективи подальших досліджень. Практична значущість отриманих результатів полягає у створенні сценарію для оболонки Bash, що допомагає автоматизувати процес створення резервних копій, забезпечуючи надійний та ефективний спосіб збереження даних. Командний рядок і пов'язані з ним можливості та інструменти створення сценаріїв є безцінним ресурсом для фахівця з кібербезпеки. Це можна порівняти з мультиінструментом із нескінченнозмінюваною конфігурацією. Поєднуючи продуману послідовність команд у конвеєр, можна створити однорядковий сценарій, що виконує надзвичайно складні функції. Для збільшення функціональності можна створювати багаторядкові сценарії.

Хоча у bash є свої переваги, такі як простий виклик інших програм або з'єднання послідовностей інших програм. У нього також є свої недоліки: bash не може виконувати операції над числами з плаваючою крапкою і не підтримує (навіть частково) складні структури даних. Але на сьогоднішній день величезна кількість програмістів працюють над постійним удосконаленням Linux: пишуть різні програми, що працюють в ній, розробляють різновиди і нові версії цієї ОС. Мета цієї статті полягає в тому,

щоб розглянути основні особливості мови BASH в ракурсі її використання для написання сценаріїв в операційній системі Linux. Linux на сьогоднішній день є єдиною альтернативою операційній системі Windows від розробників Microsoft. Подальші дослідженнях плануються в напрямку написання сценаріїв для оболонки, яка може допомогти автоматизувати завдання, маніпулювати даними та налаштовувати інтерфейс командного рядка (CLI) при використанні інших оболонок.

Список використаних джерел:

1. Види резервного копіювання: повний, інкрементальний та диференціальний бекап. URL: https://www.sim-networks.com/ukr/blog/backup-full-increment-differential

2. Русин Б. П., Погрелюк Л. В., Висоцька В. А., Осипов М. М., Варецький Я. Ю., і Капшій О. В. Архітектура системи дедублікації та розподілу даних у хмарних сховищах під час резервного копіювання. Інформаційні технології та комп'ютерна інженерія. 2019. Т. 45, Вип. 2. С. 40–63.

3. Операційні системи: Методичні вказівки до комп'ютерного практикуму: навч. посіб. для студ. спец. 113 «Прикладна математика», 125 «Кібербезпека» / КПІ ім. Ігоря Сікорського ; уклад.: М. В. Грайворонський, В. В. Демчинський. Електронні текстові дані (1 файл: 1,44 Мбайт). Київ : КПІ ім. Ігоря Сікорського, 2021. 74.

4. Paul Troncone, Carl Albing. Cybersecurity Ops with bash: Attack, Defend, and Analyze from the Command Line 1st Edition. *O'Reilly Media 1st edition* (April 2, 2019). 504 p. ISBN 978-1492041313.

5. Jason Cannon. Shell Scripting: How to Automate Command Line Tasks Using Bash Scripting and Shell Programming. *CreateSpace Independent Publishing Platform* (September 14, 2015). 99 p. ISBN 151738043X.

6. What are some useful tools or resources for learning and improving your shell scripting skills? URL: https://www.linkedin.com/advice/3/what-some-useful-tools-resources-learning-improving-your-shell

7. What is a Bash Script? URL: https://ryanstutorials.net/

8. Steve Parker. Shell Scripting: Expert Recipes for Linux, Bash, and more 1st Edition. *Wrox; 1st edition* (August 30, 2011), 608 p. ISBN-101118024486.

9. Shotts, W. E., Book, A. L. (2009). The Linux command line. Lulu. Com, 2009. 555 p.

10. Гайд для початківців: як писати Shell скрипти URL: https://blog.iteducenter.ua/guides/shell-scripting-for-beginners/

References:

1. Vydy rezervnoho kopiiuvannia: povnyi, inkrementalnyi ta dyferentsialnyi bekap. [Types of backups: full, incremental, and differential backups]. Retrievet from: https://www.sim-networks.com/ukr/blog/backup-full-increment-differential [in Ukrainian].

2. Rusyn, B.P., Pohreliuk, L.V., Vysotska, V.A., Osypov, M.M., Varetskyi, Ya. Yu. & Kapshii, O. V. (2019). Arkhitektura systemy dedublikatsii ta rozpodilu danykh u khmarnykh skhovyshchakh pid chas rezervnoho kopiiuvannia, [Architecture of the system for deduplication and distribution of data in cloud storage during backup]. *Informatsiini tekhnolohii ta kompiuterna inzheneriia – Information technology and society*, 45 (2), 40–63. [in Ukrainian].

3. M. V. Hraivoronskyi, V. V. Demchynskyi. (2021). Operatsiini systemy: Metodychni vkazivky do kompiuternoho praktykumu, [Operating systems: Methodological instructions for a computer workshop]. navch. posib. dlia stud. spets. 113 «Prykladna matematyka», 125 «Kiberbezpeka» / KPI im. Ihoria Sikorskoho ; uklad.: – Elektronni tekstovi dani (1 fail: 1,44 Mbait). – Kyiv : KPI im. Ihoria Sikorskoho, 74. [in Ukrainian].

4. Paul Troncone, Carl Albing. Cybersecurity Ops with bash: Attack, Defend, and Analyze from the Command Line 1st Edition. *O'Reilly Media 1st edition*, 504 p. ISBN 978-1492041313.

5. Jason Cannon. Shell Scripting: How to Automate Command Line Tasks Using Bash Scripting and Shell Programming. *CreateSpace Independent Publishing Platform*, 99 p. ISBN 151738043X.

6. What are some useful tools or resources for learning and improving your shell scripting skills? Retrievet from: https://www.linkedin.com/advice/3/what-some-useful-tools-resources-learning-improving-your-shell.

7. What is a Bash Script? – Retrievet from: https://ryanstutorials.net/.

8. Steve Parker. Shell Scripting: Expert Recipes for Linux, Bash, and more 1st Edition. *Wrox; 1st edition*, 608 p. ISBN-101118024486.

9. Shotts, W. E., & org Book, A. L. (2009). The Linux command line. Lulu. Com, 555 p.

10. Haid dlia pochatkivtsiv: yak pysaty Shell skrypty. Retrievet from: https://blog.iteducenter.ua/guides/shell-scripting-for beginners/ [in Ukrainian].