## Висновки

Отримали дані по кожній з моделі, аналізуючи можемо зробити висновки все-таки залишається чималий % помилки при тестуванні моделі. Найякісніша модель отримана в KNN при k=10, частота помилки дорівнюватиме 28,6%.

Список літературних джерел.

1. Wang, Xu. Noise, vibration, and harshness. wikipedia. [В Интернете] https://en.wikipedia.org/wiki/Noise,_vibration,_and_harshness.
4. murtada. car_noise_specification. kaggle. [В Интернете] 2018 г. https://www.kaggle.com/murtio/car-noise-specification.
2. Якубич К.О., Нескородєва Т.В. Аналіз факторів впливу на рівень пожежної безпеки на прикладі Data Set «Forest Fire». Матеріали всеукраїнської науково-практичної конференції для студентів, аспірантів та молодих вчених "Прикладні інформаційні технології" (29 квітня 2020 року) - Вінниця: ДонНУ імені Василя Стуса. С.72-73.
3. Новицький М.О., Нескородєва Т. В. Аналіз даних про рівень щастя населення в країнах світу. Матеріали всеукраїнської науково-практичної конференції для студентів, аспірантів та молодих вчених "Прикладні інформаційні технології" (29 квітня 2020 року) - Вінниця: ДонНУ імені Василя Стуса. С.43-45.
5. Fatima, Shahab и Mohanty, A. An Overview of Automobile Noise and Vibration Control. Березень 2014 г.

**УДК 004.4(043.2)**

*Саган М.Я., студент4 курсу спеціальності 122 «Комп'ютерні науки»*
*Єпік М.О., к.т.н., доцент, доцент кафедри комп'ютерних наук та інформаційних технологій*

## СТВОРЕННЯ ЗАШИФРОВАНОГО КАНАЛУ ДЛЯ КЕРУВАННЯ LINUX СЕРВЕРОМ ПОЗА NAT

*Донецький національний університет імені Василя Стуса, м. Вінниця*

У теперішній час у сфері системного адміністрування доволі багато задач виконуються автоматично, за допомогою скриптів або цілих фреймворків. Використання автоматизованих сценаріїв суттєво зменшує вірогідність похибки через людський фактор. Але іноді навіть у розвинутих скриптах і фреймворках можуть виникати помилки про які розробники не знають або не потурбувалися. Для таких випадків потрібен спосіб з'єднання із сервером з боку оператора для вирішення проблеми або внесення необхідних змін. Майже для всіх серверів, які працюють на базі систем Linux таким шляхом є SSH [1, 2].

Даний протокол забезпечує безпечний канал для керування сервером, через термінал. Як і більшість систем клієнт-серверного типу, даний канал потребує виділеної (білої) IP-адреси на сервері до якої зможе підключитися

клієнт, що накладає певні обмеження на сервери, які встановлені поза мережею NAT [2].

NAT дозволяє, економити ipv4 адреси, але виникають проблеми із доступом до локальних комп'ютерів в мережі з інтернету. Webrtc дозволить вирішити дану проблему та отримати унікальні ідентифікатори навіть в мережі з NAT, щоб спілкуватися за допомогою них [2].

Оскільки метою було створити максимально зручний інтерфейс для керування сервером, то не доцільно обирати реалізації webrtc, оскільки це призведе до складностей при розгортанні збоку користувачів. Тому було обране готове рішення для середовища передачі даних, а саме надзвичайно поширений месенджер Telegram [3].

Для написання додатку використано мову програмування Python оскільки інтерпретатор Python встановлений майже на всіх linux-системах за умовчанням. У проєкті використано мінімум бібліотек, для того щоб досягти максимальної сумісності із різними типами апаратного забезпечення [4].

Серверна частина проєкту є додатком, який надає доступ до системного терміналу, використовуючи системні виклики псевдотерміналу, завдяки чому відбувається повний доступ до системи, яка є ідентичною системі що отримується при використанні ssh. Далі введення і виведення даного терміналу перенаправляється в частину, яка є сервером Telegram-бота. Підключившись до бота та прийшовши авторизацію клієнт отримує повний доступ до системної оболонки.

Для того, щоб відтворити доступ максимально, не використовуються сторонні бібліотеки для системних викликів, які могли б вплинути на виведення або швидкість роботи додатку.

На теперішній час додаток підтримує виведення як текстового, так і інтерактивного псевдографічного вмісту терміналу, як це можливо в месенджері. Бібліотека скриптів додатку містить декілька збережених скриптів, які потрібні для моніторингу стану сервера.

У додатку використовуються користувацькі сесії, є можливість обмежувати права доступу засобами самого сервера, а також шляхом створення користувача на самому сервері та підв'язки бота до нього. Таким чином сервіс зможе працювати від імені створеного користувача, що мінімізує ризики ескалації привілей.

Оскільки телеграм бот працює використовуючи rest API поверх https протоколу, це означає що з'єднання шифрується за стандартами ssl tls, що надає достатньо високий рівень безпеки [5, 6].

У подальшому планується реалізувати створення декількох сесій для одного користувача, імітуючи вкладки терміналу.

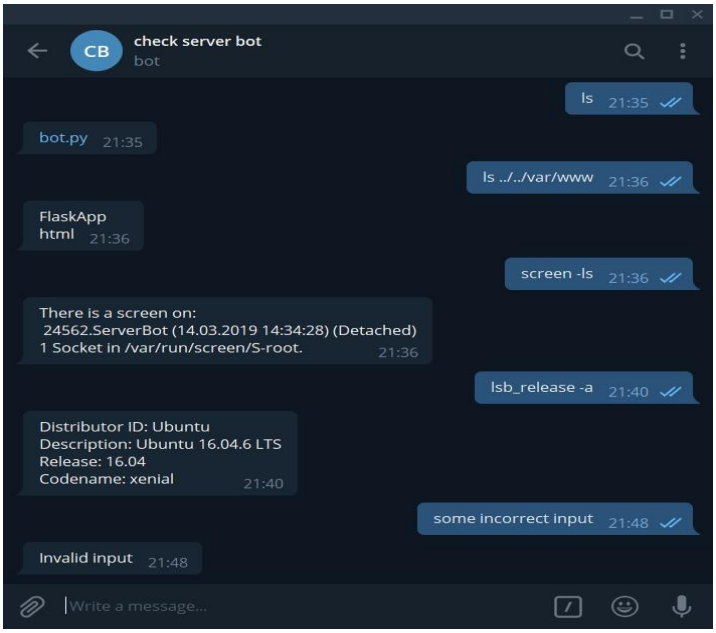


Рис 1. Приклад роботи бота

Список літературних джерел

1. webrtc.org [Електронний ресурс] – Режим доступу до ресурсу: https://webrtc.org/
2. NAT [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Network_address_translation
3. pyTelegramBotAPI [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/eternnoir/pyTelegramBotAPI
4. Python docs [Електронний ресурс] – Режим доступу до ресурсу: https://www.python.org/doc
5. xterm [Електронний ресурс] – Режим доступу до ресурсу: https://www.x.org/releases/X11R6.7.0/doc/xterm.1.html
6. os.forkpty() [Електронний ресурс] – Режим доступу до ресурсу: https://www.programcreek.com/python/example/10221/os.forkpty

**УДК 004.4(043.2)**

*Стиценко Н.Д., студентка 2 курсу*
*СО «Магістр» спеціальності*
*113 «Прикладна математика»*
*Скрипник А.В., студент 2 курсу*
*СО «Магістр» спеціальності*
*113 «Прикладна математика»*
*Вєтров О.С., старший викладач кафедри прикладної математики*

## ЗАСТОСУВАННЯ ДЕЯКИХ ПРИКЛАДНИХ КОМБІНАТОРНИХ АЛГОРИТМІВ В ЗАДАЧАХ КОЛЕКТИВНОЇ УХВАЛИ РІШЕННЯ

*Донецький національний університет імені Василя Стуса, м. Вінниця*