УДК 004:077:004.042
DOI https://doi.org/10.32782/IT/2024-2-22

Владислав ШЕЛІН
студент, кафедра комп'ютерної інженерії, факультет радіофізики, електроніки та комп'ютерних систем, Київський національний університет імені Тараса Шевченка, вул. Володимирська, 60, м. Київ, Україна, 01033
***ORCID:** 0009-0003-8507-9562*

Юрій БОЙКО
кандидат фізико-математичних наук, доцент, лауреат Державної премії України в галузі науки та техніки, начальник Інформаційно-обчислювального центру університету, завідувач кафедри комп'ютерної інженерії факультету радіофізики, електроніки та комп'ютерних, Київський національний університет імені Тараса Шевченка, вул. Володимирська, 60, м. Київ, Україна, 01033
***ORCID:** 0000-0003-1417-7424*
***Scopus Author ID:** 24722552300*

Віталій МАР'ЯНОВСЬКИЙ
кандидат технічних наук, асистент, кафедра комп'ютерної інженерії, факультет радіофізики, електроніки та комп'ютерних систем, Київський національний університет імені Тараса Шевченка, вул. Володимирська, 60, м. Київ, Україна, 01033
***ORCID:** 0009-0009-4057-5689*
***Scopus Author ID:** 58101062200*

Бібліографічний опис статті: Шелін, В., Бойко, Ю., Мар'яновський, В. (2024). Дослідження засобів побудови систем автоматизованого конфігурування обладнання мережі. *Information Technology: Computer Science, Software Engineering and Cyber Security,* 2, 167–175, doi: https://doi.org/10.32782/IT/2024-2-22

ДОСЛІДЖЕННЯ ЗАСОБІВ ПОБУДОВИ СИСТЕМ АВТОМАТИЗОВАНОГО КОНФІГУРУВАННЯ ОБЛАДНАННЯ МЕРЕЖІ

На сьогодні IT-компанії мають значну проблему, з якою стикаються кожен день. Питання полягає в тому, як зменшити кількість людино-годин мережевих інженерів на конфігурування мережевого обладнання в організаціях середнього та розміру. Значна частина дій мережевого інженера полягає у зміні конфігурації мережевого обладнання, що потребує задання узгодженого набору команд. Одна з особливостей ручних налаштувань є ймовірність помилок і як результат зниження показників стабільності роботи мережі.

***Метою роботи** є розробка автоматизованого підходу керування мережевим обладнання, який відповідатиме вимогам відкритої системи, матиме гнучкі та широко розповсюджені методів розробки. Запропонований підхід має бути альтернативою рішення по керування, яке пропонується виробниками мережевого обладнання.*

***Методологія** вирішення поставленої задачі полягає в формалізації підходу до конфігурування та діагностики роботи мережевого обладнання. З метою підтримки застарілого обладнання врахувати доступ на основі командного рядка. Розглянути існуючі методи розробки програмного забезпечення, інструментів автоматизації та аналізу текстової інформації з метою автоматизації формалізованого конфігурування.*

***Наукова новизна** роботи полягає у запропонованому підході формалізованої діагностики та конфігурування з використанням існуючих інструментів по автоматизації дистанційного доступу та інструментів аналізу неструктурованих текстів.*

***Висновки**. В роботі розглянуто найбільш розповсюджені системи управління конфігураціями та запропоновано власну систему для конфігурування наявного мережевого обладнання на основі таких модулів як Netmiko та TextFSM для обробки неструктурованих даних текстової відповіді. Запропоновані модулі дозволяють досягли більш високої гнучкості рішення в порівнянні з існуючими інструментами. Запропоноване рішення ґрунтуються на використанні мови Python мережевими інженерами або розробниками систем автоматизації керування мережею.*

***Ключові слова:** автоматизація мереж, керування мережею, адміністрування мереж, діагностика мережі, конфігурування мережевого обладнання, TextFSM.*

Vladyslav SHELIN
Student, Department of Computer Engineering, Faculty of Radiophysics, Electronics and Computer Systems, Taras Shevchenko National University of Kyiv, 60, Volodymyrska Str., Kyiv, Ukraine, 01033, vladsheln58@gmail.com
ORCID: *0009-0003-8507-9562*

Yurij BOYKO
Candidate of Physical and Mathematical Sciences, Associate Professor, Laureate of The State Prize of Ukraine In The Field of Science And Technology, Head of the Information and Computing Center of the University, Head of the Department of Computer Engineering, Faculty of Radio Physics, Taras Shevchenko National University of Kyiv, 60, Volodymyrska Str., Kyiv, Ukraine, 01033, yuriyboyko@knu.ua
ORCID: *0000-0003-1417-7424*
Scopus Author ID: *24722552300*

Vitalii MARIANOVSKYI
Candidate Of Technical Sciences, Assistant, Department of Computer Engineering, Faculty of Radiophysics, Electronics and Computer Systems, Taras Shevchenko National University of Kyiv, 60, Volodymyrska Str., Kyiv, Ukraine, 01033, vitalik_m@univ.kiev.ua
ORCID: *0009-0009-4057-5689*
Scopus Author ID: *58101062200*

To cite this article: Shelin, V., Boyko, Y., Marianovskyi, V. (2024). Doslidzhennia zasobiv pobudovy system avtomatyzovanoho konfihuruvannia obladnannia merezhi [Research of means of automating work with network equipment]. *Information Technology: Computer Science, Software Engineering and Cyber Security*, 2, 167–175, doi: https://doi.org/10.32782/IT/2024-2-22

RESEARCH OF MEANS OF AUTOMATING WORK WITH NETWORK EQUIPMENT

Today, IT companies face the big challenge of reducing the number of hours it takes network engineers to configure the network equipment in organizations of medium and large size. One of the most time-consuming tasks is to change the network equipment configuration, which requires the execution of a consistent set of commands. With such an approach, there is a high likelihood of error and typos, resulting in a decrease in network stability.

The purpose *of the present work is to develop an automated methodology for managing network equipment that will meet the requirements of an open system, and have flexible and widespread development methods. The proposed strategy should be an alternative to the management solution offered by network equipment vendors.*

The methodology *for solving the task is to formalize the approach to configuration and diagnosing the operation of network equipment. To support legacy hardware, mainly the command line access is considered.*

The scientific novelty *of the work is covered in the proposed approach of formalized diagnostics and configuration using existing tools for automating remote access and tools for analyzing unstructured texts.*

Conclusions*. The paper examines the most widely used configuration management systems and offers a new system for configuring existing network hardware based on modules such as Netmiko and TextFSM for text response processing. The offered modules allow the solution to achieve higher flexibility compared to the considered tools.*

Key words: *network automation, network management, network administration, network diagnostics, network equipment configuration, TextFSM.*

Системи автоматизації конфігурування мережевого обладнання. Мережеві пристрої, такі як комутатори, маршрутизатори та міжмережеві екрани, вже давно мають інтерфейси управління, особливо у професійних та корпоративних рішеннях. У цих пристроях завжди більш широко використовувався метод конфігурування на основі командного рядка. В більшості випадків, це пов'язано з тим, мережеве обладнання має досить широкий перелік можливості, які динамічно розвиваються, що суттєво ускладнює задачу забезпечення всіх цих можливостей через візуальні методи керування. Тому, при наявності дистанційного доступ до налаштувань мережевого обладнання можна впровадити системи автоматизації конфігурування цим пристроєм (Santyadiputra, 2021).

Нещодавній сплеск інструментів автоматизації мережі ознаменував зміни в тому, як адміністратори будують мережі та керують ними. Хоча інструменти автоматизації сервісів і додатків існують вже деякий час, інструменти, орієнтовані на мережу, стали популярними лише в останні кілька років.

Традиційно, ручна робота адміністраторів забезпечує всі необхідні налаштування

конфігурації та поточні зміни в мережі, але за останні роки з'явилися засоби автоматизації мережі, щоб звести до мінімуму конфігурування в ручну. Хоча засоби автоматизації мережі можуть використовувати різні підходи до автоматизації, усі вони спрямовані на скорочення часу на підтримку мережі, який витрачається на прості та повторювані процеси налаштування мережевого устаткування.

Майже кожен сучасний інструмент автоматизації мережі може автоматизувати зміни конфігурації в середовищах багатьох виробників. Системи автоматизації досягають цього шляхом обробки команд під певний синтаксис командного рядка, а потім надсилають команди на зміну параметрів на кожен пристрій, який потребує змін в конфігурації. Замість того, щоб адміністратор мережі використовував сеанс Secure Socket Shell (SSH) для кожного мережевого пристрою, щоб вручну змінювати текстові конфігурації, інструменти автоматизації створюють сценарії конфігурації, які досягають тієї ж мети за набагато менший час і з меншою кількістю помилок.

Іншим популярним методом для інструментів мережевої автоматизації для доступу та автоматизації конфігурацій мережевих пристроїв є API. Цей більш сучасний та формалізований спосіб взаємодії з мережевим обладнанням може скоротити час адміністратора, який витрачається на часті зміни мережі.

Деякі засоби автоматизації мережі виходять за рамки простої автоматизації змін конфігурації. Приклади інших функцій автоматизації мережі: резервне копіювання конфігурації, дозволяє створювати заплановані резервні копії, що шифруються та безпечно зберігаються на випадок, якщо конфігурацію мережевого устаткування потрібно повернути до попереднього стану або відновити; контроль доступу до інструментів керує доступом для тих, хто має повноваження вносити зміни в конфігурацію в певних сегментах мережі; також може створюватись журнал обліку, щоб показати повну історію змін конфігурації; моніторинг і перевірка відповідності конфігурації політикам мережі, автоматизує процес визначення, чи відповідає мережевий пристрій попередньо встановленим стандартам та політикам; оцінка вразливості, автоматично визначає параметри захищеності мікропрограми та конфігурації, яка може бути вразливою; моніторинг продуктивності, дозволяє аналізувати дані про продуктивність мережі на обладнанні, щоб надати рекомендації щодо конфігурування для подальшого підвищення продуктивності; мережева оркестровка автоматично виявляє мережеві пристрої для централізованого контролю та наскрізної координації доповнень і змін конфігурації локальної мережі.

Наразі існує досить багато систем автоматизації роботи з мережевим обладнанням і кожен інструмент має свої особливості, переваги та недоліки, розглянемо найбільш популярні інструменти мережевої автоматизації.

Можливості Python для автоматизації мереж. Впродовж останніх кількох років мережева автоматизація набула великої популярності. В результаті сучасному інженеру доступний постійно зростаючий арсенал інструментів на різних платформах (Marco Martino Rosso, 2023), які допомагають реалізувати автоматичну конфігурацію мережі та керування змінами в мережевій конфігурації (Altalebi, 2022).

Python є найбільш популярною мовою програмування для автоматизації мережі. Так, наприклад виробник Cisco віддав перевагу мові програмування Python перед будь-якою іншою мовою для цілей автоматизації. Однією з ознак широкого використання тих чи інших мов програмування для автоматизації є розробка найбільш популярних навчальних матеріалів в цьому напрямку. Одним з прикладів є компанція Cisco, яка запровадила навчальну програму з програмування для мережевих інженерів (Developing Applications and Automating Workflows using Cisco Platforms). Аналогічні тенденції спостерігаються і у інших лідерів ринку мережевого обладнання, наприклад Huawei (HCIA-Datacom. Huawei Certified ICT Associate-Datacom).

Python – це мова програмування загального призначення, для якої розроблено велику

Таблиця 1

Порівняльна таблиця інструментів мережевої автоматизації

	Chef	**Puppet**	**Ansible**
Архітектура	Client-server	Client-server	Agentless (через SSH)
Синтаксис	Ruby DSL	Puppet DSL	YAML
Мова програмування	Ruby – client Erlang – server	Ruby	Python
Підключення до вузла	SSH	Агент підключається через SSL до серверу	Ansible (через SSH)

кількість бібліотек. Ці бібліотеки корисні для написання програм, орієнтованих на роботу з мережею. Усі ці бібліотеки є достатньо розвиненими, добре перевіреними та мають підтримку спільноти. Можна також використовувати і інші мови програмування, такі як Java, C# тощо, але для поставленої задачі Python має ряд суттєвих переваг (Choi, 2021): легка у використанні та розвинена мова; легка для вивчення, використання; має широкий вибір бібліотек, що значно розширює функціонал та швидкість розробки; краще підходить для створення сценаріїв, в тому числі і автоматизації мережі; мова Python має широке використання та величезну спільноту, тому спрощує та пришвидшує процес розробки.

Використання бібліотек Paramiko та Netmiko. Paramiko – це реалізація протоколу SSHv2 мовою програмування Python. Головною перевагою використання paramiko є те, що модуль не вимагає встановлення агента на керовані хости. Особливо це актуально для пристроїв, які дозволяють працювати з ними тільки через CLI інтерфейс. Даний модуль може ефективно використовуватись для створення систем автоматизації роботи з мережевим обладнанням.

Netmiko – це Python модуль, який розширює адаптує використання Paramiko для мережевих пристроїв, netmiko це надбудова Paramiko, яка враховує особливості роботи з командним рядком мережевого обладнання. Модуль Netmiko дозволяє уніфікувати роботу з мережевим обладнання шляжою врахування особливостей різних виробників наступні дії: функцію встановлення підключення з пристроєм, пересилання команд, аналіз результату їх виконання. Вхідними параметрами для роботи модулю є набір команд та послідовність дії для автоматизації. Наприклад, Netmiko враховує, що для зміни конфігурації пристрою компанії Cisco майже всі команди, які призначені для зміни конфігурації обладнання виконується у відповідному режимі глобальної конфігурації, тому перехід у відповідний режим вразований в модулі.

Для роботи з комутаторами компанії D'Link в модуль вбудовано перед початком роботи вимикання функцію посторінкового виводу інформації (clipaging). Наведені механізми адаптації командного рядка з урахування особливостей різних пристроїв знижують складність розробки сценаріїв автоматизації.

Модуль Netmiko дозволяє виконувати підключення до обладнання та виконання команд через протоколи SSH і Telnet. Модуль має широкий список підтримуваних платформ, окрім вже описаним та популярних виробників, вузькоспеціалізованого обладнання, такого як комутаційне обладнання компанії Ubiquiti, кожен модуль містить інформацію про особливості даної серії пристроїв та особливості що враховуються при підключенні та роботі (Netmiko Supported Platform List).

Взаємодія з обладнанням на основі поточних параметрів. На даний час існує багато систем автоматизації процесу конфігурування мережевого обладнання, серед них: Puppet, Chef, Salt, Jenkins та Ansible. Один з недоліків даних систем є те, що процес виконання певних налаштувань виконується без врахування поточної конфігурації та поточних параметрів роботи мережевого обладнання. Можливість автоматизованого конфігурування з урахуванням зазначених параметрів створювати більш гнучкі системи автоматизації роботи з мережевим обладнанням, що не просто виконують певний набір команд, а проводять аналіз поточних налаштувань і тільки на їх основі виконують команди на мережевому обладнанні. Обробка отриманого текстового виводу мережевого обладнання може значно розширити можливості систем автоматизації мережевого обладнання, стає можливим робота на основі логічного аналізу виводу певних команд діагностики, сервісу, виводу поточних параметрів та налаштувань. В даному випадку є два підходи. Перший підхід передбачає отримання автоматизованою системою файлу поточної конфігурації мережевого обладнання з наступною його обробкою команда за командою, це дозволяє отримати поточні налаштування пристрою, стан певних системних параметрів та функцій, але даний метод не передбачає отримання інформації про поточні параметри роботи та стан системи в реальному часі. Другим підходом є виконання певних сервісних команд, що виводять поточну інформацію про пристрій, поточні налаштування та стан. Альтернативою запропонованого підходу є використання можливостей протоколу SNMP (Shaffi, 2013), але наявність застарылого обладнання в мережі робить більш раціональним підхід, який передбачає безпосередню роботу з CLI мережевого обладнання. Нові пристрої та платформи підтримують NETCONF, REST API, та RESTCONF з YANG моделями та структурами даних, але наявність застарылого обладнання призводить до необхідності використання SSH, Telnet та стандартного CLI. Доступ через API зазвичай дозволяє отримати вже структурований висновок, але текстовий висновок, що отримується

через CLI від мережевих пристроїв є непридатним для прямої обробки. Для отримання корисних даних традиційно використовується стандартний модуль re і регулярні вирази, але дане рішення є громіздким та вузькоспеціалізованим під конкретний випадок.

Для вирішення цієї задачі в компанії Google для роботи з власним мережевим обладнанням був створений інструмент TextFSM. TextFSM – це кінцевий автомат на основі шаблонів для розбору напів форматованого тексту в таблиці. Даний інструмент розміщений на платформі Google Code. Він легко налаштовується, оскільки працює з окремими визначеннями шаблонами, які містять змінні та правила з регулярними виразами для обробки даних. Ця бібліотека корисна для аналізу будь-якого текстового виводу, в тому числі CLI з мережевих пристроїв. Google TextFSM – це модуль Python, тому його можна використовувати в сценаріях або як окремий інструмент. Це зручний спосіб перетворити вихід інформації з CLI у структуровані дані (Zhao-yang Li, 2022).

Розглянемо декілька прикладів розбору виводу команд для комутаторів D’Link за допомогою TextFSM. Розглянемо результат виконання команди show switch на комутаторі, який потім зберігається у файл. На рис. 1 команда виводить основні дані про мережевий пристрій.

На рис. 2 сценарій мовою Python приймає текстовий вивід команди з комутатора (виконання команди та отримання виводу також можна автоматизувати за допомогою інструментів Python).

```
Command: show switch


Device Type        : DES-3028 Fast Ethernet Switch
MAC Address        : 00-21-91-92-47-B3
IP Address         : 192.168.31.235 (Manual)
VLAN Name          : default
Subnet Mask        : 255.255.255.0
Default Gateway    : 0.0.0.0
Boot PROM Version  : Build 1.00-B04
Firmware Version   : Build 2.94.B22
Hardware Version   : A1
System Name        : test
System Location    :
System Uptime      : 0 days, 0 hours, 0 minutes, 56 seconds
System Contact     :
Spanning Tree      : Disabled
GVRP               : Disabled
IGMP Snooping      : Disabled
VLAN trunk         : Disabled
802.1x             : Disabled
TELNET             : Enabled(TCP  23)
WEB                : Enabled(TCP  80)
RMON               : Disabled
SSH                : Enabled(TCP  22)
SSL                : Disabled
CLI Paging         : Disabled
Syslog Global State       : Disabled
Dual Image         : Supported
Password Encryption Status : Disabled
```

Рис. 1. Команда show version на обладнанні D’Link

```
import textfsm
from pprint import pprint
import yaml

with open('output.txt') as f:
      show_switch = f.readlines()

with open('show_switch.template') as template:
      fsm = textfsm.TextFSM(template)
      result = fsm.ParseText(show_switch)

zip_iterator = zip(fsm.header, result[0])
dictionary = dict(zip_iterator)
pprint(dictionary)

with open('show_switch.yaml', 'w') as file:
      documents = yaml.dump(dictionary, file)
```

Рис. 2. Обробка результату виконання команди

Формалізація підходу полягає в тому, що використовуючи TextFSM досягається аналіз вихідних даних. Послідовність аналізу описується в окремому файлі за допомогою визначеної структури та певної кількості правил із регулярними виразами. Послідовність операцій модуля TextFSM для отримання шаблону ускладнена для сприйняття, що є наслідком закладеної в неї гнучкості та універсальності. Розглянемо даний процес на прикладі. На рис. 3 продемонстровано структуру шаблону TextFSM для наведеної раніше команди комутатора D'Link.

Файл шаблону TextFSM структурований у два основні розділи:

1) Визначення значень, які визначають поля та типи, які слід витягти

2) Визначення стану, що визначає правила аналізу вмісту

Синтаксичний аналізатор TextFSM створить записи таблиці на основі заданого визначення шаблону під час виконання, що є методом формалізації аналізу виводу командного рядка. Якщо регулярний вираз із визначенням збігається зі значенням, то значення тримться на основі пов'язаного визначення, а потім значення додається до буфера. TextFSM продовжить аналіз порядково (за замовчуванням). Якщо значення вже призначено, вміст буде замінено.

Після виконання сценарію Python ми отримали файл show_switch.yaml. Файл містить структуровані дані, до яких можна програмно отримати доступ та обробляти їх в процесі виконання сценарію автоматизації. Можливий вміст файлу show_switch.yaml показано на рис. 4.

Висновки. У роботі розглянуті можливості мови Python для автоматизації роботи з мережевим обладнанням та можливості модулю текстової обробки відповіді TextFSM, що був адаптований для роботи з мережевим обладнанням D'Link. Відповідно до відкритої документації стандарту мережевого рядка D'Link були створені шаблони обробки текстової відповіді мережевого обладнання.

Продемонстровано переваги доступу до обладнання через SSH/Telnet з використанням платформи netmiko у випадках роботи з застарілим (legacy) обладнанням. Проведені випробування використання платформи netmiko показали поглиблену орієнтованість на мережеве обладнання та у своїй роботі враховує індивідуальні особливості відповідно до виробника, серії та моделі. Дана платформа дозволяє емулювати CLI з Python – тобто відправляти на обладнання команди та отримувати у відповідь на них, як правило, текстовий висновок того чи іншого ступеня відформатованості та передбачуваності. Новішим та популярнішим підходом до обробки таких даних є фреймворк TextFSM від Google, який виконує обробку згідно визначених шаблонів та є доволі гнучким у застосуванні.

Створена на основі розглянутих технологій система автоматизованого конфігурування

```
Value Device (\S+)
Value Mac (\w+(-\w+){5})
Value IP (\d+(\.\d+){3})
Value VLAN (\S+)
Value Subnet (\d+(\.\d+){3})
Value Gateway (\d+(\.\d+){3})
Value Boot (Build \S+)
Value Firmware (Build \S+)
Value Hardware (\S+)
Value Name (\S+)
Value Location (\S+)
Value Telnet (\S+.+)
Value SSH (\S+.+)
Value Clipaging (\S+)

Start
  ^.*Device Type.*: ${Device}
  ^.*MAC Address.*: ${Mac}
  ^.*IP Address.*: ${IP}
  ^.*VLAN Name.*: ${VLAN}
  ^.*Subnet Mask.*: ${Subnet}
  ^.*Default Gateway.*: ${Gateway}
  ^.*Boot PROM Version.*: ${Boot}
  ^.*Firmware Version.*: ${Firmware}
  ^.*Hardware Version.*: ${Hardware}
  ^.*System Name.*: ${Name}
  ^.*System Location.*: ${Location}
  ^.*TELNET.*: ${Telnet}
  ^.*SSH.*: ${SSH}
  ^.*Clipaging.*: ${Clipaging} -> Record
```

Рис. 3. Приклад структура шаблону TextFSM

```
Boot: Build 1.00-B04
Clipaging: Disabled
Device: DES-3028
Firmware: Build 2.94.B22
Gateway: 0.0.0.0
Hardware: A1
IP: 192.168.31.235
Location: ''
Mac: 00-21-91-92-47-B3
Name: test
SSH: Enabled(TCP  22)
Subnet: 255.255.255.0
Telnet: Enabled(TCP  23)
VLAN: default
```

Рис. 4. Приклад структурованих даних

мережевого обладнання була протестована для задачі автоматизації мережі, побудованої на основі обладнання D'Link. Проведене тестування з іншими аналогічними платформами, розглянутими в роботі, виявлено ряд проблем та часткову несумісність з застарілим обладнання, що є додатковим фактором вибору запропонованої в роботі платформи.

ЛІТЕРАТУРА:

1. Li Z., Zhou B., Zhou W., Xu T., Zhang X. Application Research on an Automated Batch Network Reinforcement Method Based on SSH Protocol. *Communications, Signal Processing, and Systems.* CSPS 2021. Lecture Notes in Electrical Engineering, 2022. vol 878. Springer. DOI: 10.1007/978-981-19-0390-8_13.

2. Santyadiputra G. S., Listartha I. M. E., Saskara G. A. J. The effectiveness of Automatic Network Administration (ANA) in network automation simulation at Universitas Pendidikan Ganesha. *Journal of Physics: Conference Series.* 2022. Vols 1810. Issue 117. DOI: 10.1088/1742-6596/1810/1/012028.

3. Choi B. Introduction to Python Network Automation: The First Journey. Apress Berkeley, 2022. CA. Pages 1 – 896. DOI: 10.1007/978-1-4842-6806-3.

4. Shaffi A. S. and Al-obaidy, M. Managing Network Components Using Snmp. *International Journal of Scientific Knowledge*, 2(3), 2013. pp. 11–18.

5. Rosso M., Aloisio A., Parol J., Marano G., Quaranta G. Intelligent automatic operational modal analysis. *Mechanical Systems and Signal Processing*. 2023. Volume 201. DOI: 10.1016/j.ymssp.2023.110669.

6. Altalebi O. W. J., Ibrahim A. A. Optimization of Elapsed Time of Automation for Large-Scale Traditional Networks and Proposing New Automation Scripts. *International Congress on Human-Computer Interaction, Optimization and Robotic Applications* (HORA), 2022. pp. 1-10. DOI: 10.1109/HORA55278.2022.9799873.

7. Developing Applications and Automating Workflows using Cisco Platforms. 200-901 DEVASC. URL: https://www.cisco.com/c/en/us/training-events/training-certifications/exams/current-list/devasc-200-901.html (дата звернення: 12.08.2024).

8. HCIA-Datacom. Huawei Certified ICT Associate-Datacom. Training and certificating engineers with basic datacom knowledge and skills. URL: https://e.huawei.com/en/talent/cert/#/careerCert (дата звернення: 12.08.2024).

9. Netmiko Supported Platform List. URL: https://github.com/ktbyers/netmiko/blob/develop/PLATFORMS.md (дата звернення: 12.08.2024).

10. Parsing command output with TextFSM. URL: https://pyneng.readthedocs.io/en/latest/book/21_textfsm/index.html (дата звернення: 12.08.2024).

11. DGS-3120 Series Managed Switch CLI Reference Guide. URL: https://eu.dlink.com/-/media/business_products/dgs/dgs-3120/manual/dgs3120_series_r300_cli-reference-guide.pdf (дата звернення: 12.08.2024).

12. TextFSM library official repository and documentation. URL: https://github.com/google/textfsm/wiki/TextFSM (дата звернення: 12.08.2024).

REFERENCES:

1. Li, Z., Zhou. B., Zhou, W., Xu, T. & Zhang, X. (2022). Application Research on an Automated Batch Network Reinforcement Method Based on SSH Protocol. *Communications, Signal Processing, and Systems.* CSPS 2021. Lecture Notes in Electrical Engineering, vol 878. Springer. DOI: 10.1007/978-981-19-0390-8_13.

2. Santyadiputra, G. S., Listartha, I. M. E., Saskara, G. A. J. (2022). The effectiveness of Automatic Network Administration (ANA) in network automation simulation at Universitas Pendidikan Ganesha. *Journal of Physics: Conference Series.* Vols 1810. Issue 117. DOI: 10.1088/1742-6596/1810/1/012028.

3. Choi., B. (2022). Introduction to Python Network Automation: The First Journey. Apress Berkeley, CA. Pages 1 – 896. DOI: 10.1007/978-1-4842-6806-3.

4. Shaffi, A. S. (2013). and Al-obaidy, M. Managing Network Components Using Snmp. *International Journal of Scientific Knowledge*, 2(3), pp. 11–18.

5. Rosso, M., Aloisio, A., Parol, J., Marano, G., Quaranta, G. (2023). Intelligent automatic operational modal analysis. *Mechanical Systems and Signal Processing*. Volume 201. DOI: 10.1016/j.ymssp.2023.110669.

6. Altalebi, O. W. J., Ibrahim, A. A. (2022). Optimization of Elapsed Time of Automation for Large-Scale Traditional Networks and Proposing New Automation Scripts. *International Congress on Human-Computer Interaction, Optimization and Robotic Applications* (HORA), pp. 1–10. DOI: 10.1109/HORA55278.2022.9799873.

7. Developing Applications and Automating Workflows using Cisco Platforms. 200-901 DEVASC. Retrieved from: https://www.cisco.com/c/en/us/training-events/training-certifications/exams/current-list/devasc-200-901.html (accessed date: 12.08.2024).

8. HCIA-Datacom. Huawei Certified ICT Associate-Datacom. Training and certificating engineers with basic datacom knowledge and skills. Retrieved from https://e.huawei.com/en/talent/cert/#/careerCert (accessed date: 12.08.2024).

9. Netmiko Supported Platform List. github.com. Retrieved from: https://github.com/ktbyers/netmiko/blob/develop/PLATFORMS.md (accessed date: 12.08.2024).

10. Parsing command output with TextFSM. pyneng.readthedocs.io. Retrieved from https://pyneng.readthedocs.io/en/latest/book/21_textfsm/index.html

11. DGS-3120 Series Managed Switch CLI Reference Guide. dlink.com. Retrieved from https://eu.dlink.com/-/media/business_products/dgs/dgs-3120/manual/dgs3120_series_r300_cli-reference-guide.pdf (accessed date: 12.08.2024).

12. TextFSM library official repository and documentation. github.com. Retrieved from https://github.com/google/textfsm/wiki/TextFSM (accessed date: 12.08.2024).