МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЧЕРНІГІВСЬКА ПОЛІТЕХНІКА»
WIN-WIN EDIH



Co-funded by
the European Union



Співфінансується
Європейським Союзом



ВІЛЬНЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ В ОСВІТІ, НАУЦІ ТА БІЗНЕСІ

FREE/OPEN SOURCE SOFTWARE IN EDUCATION, SCIENCE & BUSINESS (FOSS 2026)

Матеріали II Міжнародної науково-практичної конференції
01–02 жовтня 2026 р.
м. Чернігів

Чернігів 2026

УДК 004.49:[37+001+334.722]
В41

Рекомендовано до друку вченою радою Національного університету «Чернігівська політехніка» (протокол № 12 від 02.11.2026 р.)

Вільне програмне забезпечення в освіті, науці та бізнесі : матеріали ІІ Міжнар. В41 наук.-практ. конф. (м. Чернігів, 01–02 жовтня 2026 р.) / відп. за вип. Д. Б. Мехед. Чернігів : НУ «Чернігівська політехніка», 2026. 156 с.

ISBN 978-617-8584-09-2

Збірник містить матеріали ІІ Міжнародної науково-практичної конференції «Вільне програмне забезпечення в освіті, науці та бізнесі» (FOSS 2026), що відбулася онлайн 01–02 жовтня 2026 року в Національному університеті «Чернігівська політехніка». Розглянуто питання використання вільного та відкритого програмного забезпечення в освіті, науці, бізнесі, ІТ-інфраструктурі, кібербезпеці, аналізі даних і штучному інтелекті. Матеріали подано українською та англійською мовами.

Для науковців, викладачів, здобувачів освіти, фахівців ІТ-галузі та всіх, хто цікавиться вільним програмним забезпеченням.

Матеріали подано в авторській редакції. Відповідальність за зміст, достовірність фактів, цитат і посилань несуть автори.

Конференцію проведено в межах проєкту WIN-WIN EDIH (грантова угода № 101191647), співфінансованого Європейським Союзом у межах програми Digital Europe. Висловлені погляди та думки належать виключно автору(ам) і не обов'язково відображають позицію Європейського Союзу чи Європейської Комісії.

Відповідальний за випуск: Мехед Дмитро Борисович

ISBN 978-617-8584-09-2

© НУ «Чернігівська політехніка», 2026

Чопорова О. В., Чопоров С. В. ВИКОРИСТАННЯ 3D-МОДЕЛЮВАННЯ ТА АЛГОРИТМІВ НЕЙРОННИХ МЕРЕЖ В НАВЧАЛЬНИХ ПРОЄКТАХ ІНЖЕНЕРНО-ТЕХНІЧНОГО СПРЯМУВАННЯ / USE OF 3D MODELING AND NEURAL NETWORK ALGORITHMS IN EDUCATIONAL PROJECTS OF ENGINEERING AND TECHNICAL DIRECTION 82

Чуб М. В. ВИЯВЛЕННЯ ТА ПРОТИДІЯ АДАПТИВНИМ ВЕБ-АТАКАМ ЗА ДОПОМОГОЮ МОДЕЛЕЙ ГЛИБОКОГО НАВЧАННЯ 86

Шевченко Ю. О. ПОРІВНЯЛЬНЕ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ POSTGRESQL ТА INFLUXDB ДЛЯ ЗБЕРІГАННЯ ДАНИХ ФІЗИЧНОЇ АКТИВНОСТІ / COMPARATIVE STUDY OF POSTGRESQL AND INFLUXDB PERFORMANCE FOR STORING PHYSICAL ACTIVITY DATA 89

Al-Akwaa K. H. M. A. OPEN-SOURCE DATA SCIENCE FOR SUSTAINABLE DEVELOPMENT: A REPRODUCIBLE ANALYTICAL FRAMEWORK 93

Borysenko O. A. AN ARCHITECTURAL APPROACH TO LLM-BASED ROOT CAUSE ANALYSIS FOR CONTAINER ORCHESTRATION INCIDENTS 100

Burmaka I. LOCAL, OPEN, AND ENERGY-EFFICIENT: HARNESSING X86 NPUS FOR EVERYDAY AI TASKS 102

Byvoino T. P., Byvoino P. G. SIMULATION MODELING AND STATISTICAL ANALYSIS OF THE EFFICIENCY OF AN EMERGENCY MEDICAL SERVICE STATION 106

Didusenko O. A., Bula Y. I., Shymchuk G. V. REPRODUCIBLE FOSS FRAMEWORK FOR RESILIENCE EVALUATION OF CONTAINERIZED DISTRIBUTED SYSTEMS 109

Didusenko O. A., Bula Y. I., Shymchuk G. V. REPRODUCIBLE FOSS TESTBED FOR FAULT-TOLERANT MULTIPATH ROUTING IN IP NETWORKS 112

Ebirim V., Jadama A. F. R. RESILIENT DIGITAL EDUCATION IN DISRUPTED ENVIRONMENTS: THE CCDS APPROACH 115

Fraya DIGITAL TRANSFORMATION OF BUSINESS THROUGH FREE AND OPEN SOURCE SOFTWARE: OPPORTUNITIES, BARRIERS, AND STRATEGIC PATHWAYS 119

Kala P. RUN MONGODB YOUR WAY: A PRACTICAL OPEN-SOURCE STACK FOR SELF-MANAGED DEPLOYMENTS 121

Lukin O. A MODERN C++ KMIP CLIENT BASED ON A REUSABLE PROTOCOL LIBRARY 124

Mekhed D. B. DEVELOPING STUDENTS' CYBER HYGIENE SKILLS THROUGH THE USE OF FREE SPECIALIZED SOFTWARE 127

Nahornyi B. I., Lysenok D. V., Savenko A. O. AN OPEN BROWSER-BASED CO-SIMULATOR OF STM32 FIRMWARE AND ELECTRICAL CIRCUITS AS A FREE ALTERNATIVE TO PROTEUS 130

Romanenko M. V. FOSS APPROACH IN MOBILE DEVELOPMENT: CREATING AN OPEN-SOURCE ANDROID APPLICATION WITH THE INTEGRATION OF READY-MADE LIBRARIES FOR STYLISTIC RECOMMENDATIONS 134

Shevchenko N. A., Kibitkin D. S., Shymchuk G. V. REPRODUCIBLE OPEN-SOURCE BENCHMARK OF CONCEPT-DRIFT DETECTORS FOR STREAMING REGRESSION 137

Shevchenko N. A., Kibitkin D. S., Shymchuk G. V. INTERPRETABLE OPEN-SOURCE MACHINE LEARNING FOR NATURAL GAS CONSUMPTION FORECASTING WITH WEATHER AND LAG FEATURES 140

Shivashankar C. ARCHITECTING DATABASE DEPLOYMENTS ON KUBERNETES: PATTERNS, TRADE-OFFS AND EXPERIENCES 143

Shymchuk G. V., Yatskiv P. O. DETECTION OF ANOMALOUS STATES IN IP NETWORKS USING MACHINE LEARNING AND OPEN-SOURCE SOFTWARE TOOLS 146

Shymchuk G. V., Yatskiv P. O. OPEN-SOURCE SOFTWARE ENVIRONMENT FOR MODELING FAULT-TOLERANT MULTIPATH ROUTING IN IP NETWORKS 149

Silva F. HOSTING SOMEONE ELSE'S APP: IFRAME CHOREOGRAPHY IN PMM 152

Šupić A. ZERO-DOWNTIME MIGRATION WITH PERCONA CLUSTERSYNC FOR MONGODB 154

Shymchuk G. V.1, Yatskiv P. O.2
1 – Ternopil Ivan Puluj National Technical University, Department of Computer Science, Ternopil, Ukraine
2 – Ternopil Ivan Puluj National Technical University, Department of Computer-Integrated Technologies, Ternopil, Ukraine
Corresponding author: Grigorii Shymchuk, e-mail: gorych@gmail.com, ORCID: 0000-0003-2362-7386

OPEN-SOURCE SOFTWARE ENVIRONMENT FOR MODELING FAULT-TOLERANT MULTIPATH ROUTING IN IP NETWORKS

Abstract. An open-source software environment for modeling and investigating fault-tolerant multipath routing in IP networks is proposed. The environment combines Mininet, Open vSwitch, FRRouting, iperf3, and Python and supports reproducible creation of topologies with multiple equal-cost paths, programmatic link failures, and route-state monitoring. A baseline topology with two equal-cost paths between end hosts is defined together with a three-stage sequence: normal operation, failure, and recovery. Functional verification demonstrates that connectivity is preserved after one path fails because the alternative path remains available. The proposed environment can be used for subsequent quantitative comparison of ECMP and modified multipath-routing methods without specialized commercial networking hardware.

Keywords: open-source software; IP networks; multipath routing; fault tolerance; ECMP; Mininet; FRRouting.

Introduction

Modern IP networks provide the transport foundation for distributed systems, cloud platforms, and critical digital services. A failure of a single link or intermediate node should not cause a prolonged loss of availability. Multipath routing uses several alternative paths between a source and a destination, improving resilience, reducing dependence on one route, and enabling more flexible load distribution [1, 2].

Current multipath-routing methods combine path selection with network-state estimation, traffic prediction, and adaptive flow distribution [3, 4]. This strengthens interest in route control that accounts for resource availability. However, validation on a physical network requires multiple devices, complicates precise repetition of identical failures, and increases experimental cost.

FOSS tools transfer these experiments into a software environment. Mininet is widely used to emulate SDN topologies and test networking algorithms [5], while Open vSwitch provides programmable switching in virtualized environments [7]. Topology, addressing, link parameters, failure timing, and metrics can be fixed in scripts and reproduced on another system.

The aim of this work is to build a reproducible open-source environment for modeling fault-tolerant multipath routing and verifying IP-network behavior during normal operation, failure of one path, and recovery.

Materials and Methods

The experimental environment is defined on Ubuntu 22.04 LTS and includes Mininet 2.3.0, Open vSwitch 4.0.0, FRRouting 10.7.1, iperf3 3.21, and Python 3.10. FRRouting 10.7.1 and Open vSwitch 4.0.0 were released in 2026 [6, 7], while iperf3 3.21 is used to generate TCP/UDP traffic and later measure throughput [8]. All components are open source or free software.

The baseline topology contains hosts H1, H2 and routers R1–R4. Two equal-cost paths are defined between R1 and R4: P1 = R1–R2–R4 and P2 = R1–R3–R4. Identical internal-link metrics give a total cost of 2 for both paths. Link R2–R3 is retained for extended reconfiguration scenarios. The topology is shown in figure 1.

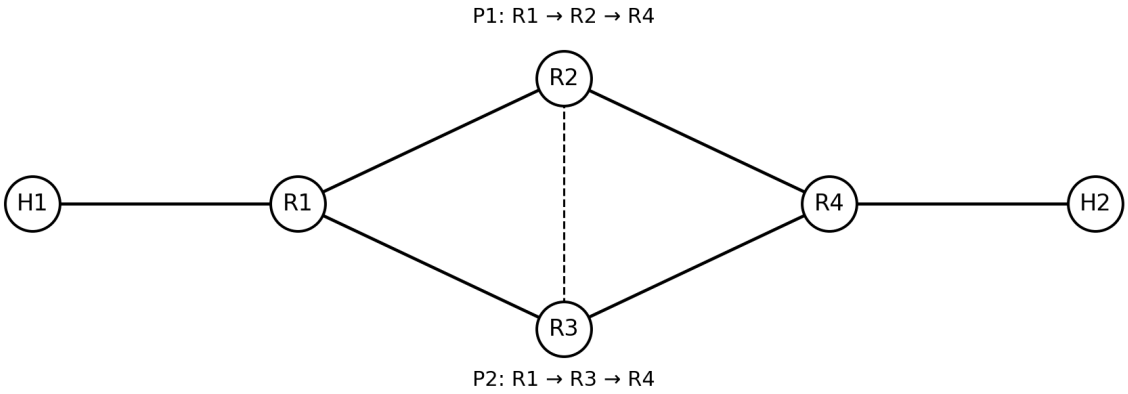

Figure 1 – Experimental topology with two equal-cost paths

For reproducibility, topology parameters are fixed in a Python scenario: node names, IP addressing, metrics, bandwidth, delay, and the link-state sequence. The baseline scenario has three intervals: 0-20 s, normal operation, 20-40 s, R1–R2 disabled; 40-60 s, link restored. This provides an unambiguous relation between telemetry and failure/recovery events.

Table 1 – Baseline configuration of the reproducible experiment

Parameter	**Value**
Operating system	Ubuntu 22.04 LTS
Network emulator	Mininet 2.3.0
Routing	FRRouting 10.7.1; ECMP/OSPF or static multipath routes
Software switching	Open vSwitch 4.0.0
Traffic generator	iperf3 3.21; TCP/UDP
Topology	2 end hosts, 4 routers, 2 equal-cost paths
Scenario	0–20 s: normal; 20–40 s: R1–R2 down; 40–60 s: restore

The routing state is monitored through Linux/FRRouting tables and traffic is generated between H1 and H2. The quantitative stage is intended to measure recovery time $T_r = t_r - t_f$, where t_f is the failure instant and t_r is restored delivery, packet-loss ratio $L = N_{loss} / N_{send} \cdot 100\%$, mean RTT, throughput, and the number of active next hops. The present work evaluates the testbed structure and route-set transitions, so empirical RTT and throughput are not replaced by synthetic values.

Results

Functional verification confirms two equal-cost paths between R1 and R4 during normal operation. With identical metrics, P1 and P2 have the same total cost and can be installed as two next hops for the same prefix. This satisfies the basic ECMP conditions and supports experiments on redundancy and flow distribution.

When R1–R2 is disabled, P1 becomes unavailable while P2 = R1–R3–R4 remains operational. H1–H2 connectivity is preserved and the number of available paths decreases from two to one. After R1–R2 is restored, both P1 and P2 are available again. Thus, the testbed reproduces the sequence «2 paths → 1 path → 2 paths» required for fault-tolerant routing experiments.

Table 2 – Functional verification of route-state transitions

State	**Available paths**	**Next hops**	**H1–H2 connectivity**
Normal	P1, P2	2	preserved
R1–R2 failure	P2	1	preserved
Recovery	P1, P2	2	preserved

This result is not a convergence-speed measurement for a specific protocol. Reaction time depends on OSPF/BFD timers, FRRouting configuration, Linux-kernel behavior, and load. The proposed scheme separates functional validation from the measurement stage: first the route set is verified, then T_r, packet loss, RTT, and throughput are measured on the same topology.

The practical output is a reproducible FOSS testbed that can be stored in a Git repository together with the Python topology script, FRRouting configurations, and iperf3 commands. It

provides a common basis for comparing standard ECMP with a modified multipath-routing method under identical conditions.

Discussion

The proposed approach combines accessible FOSS tools with controlled experimentation. Unlike a physical testbed, a software scenario can reproduce the same failure sequence repeatedly. This simplifies statistical comparison and makes it possible to vary only one factor at a time, such as convergence timers, link metric, or path-selection logic.

Software emulation also has limitations: Mininet virtual nodes share host resources, so absolute throughput and latency values should not be transferred directly to backbone networks. The testbed is most appropriate for comparative experiments on the same hardware platform. Future work will automate telemetry collection and repeated failures, estimate confidence intervals for T_r and packet loss, and compare ECMP with a modified algorithm that accounts for link state.

Conclusions

An open-source environment for modeling fault-tolerant multipath routing has been formed using Mininet, Open vSwitch, FRRouting, iperf3, and Python. The topology contains two equal-cost routes and reproduces three controlled states: normal operation, failure of one path, and recovery.

Functional verification confirms preservation of the alternative route after a link failure and restoration of the initial route set after recovery. The scientific and practical value of the approach is a reproducible basis for quantitative comparison of standard and modified multipath-routing methods without specialized hardware. The next stage is measurement of T_r, packet loss, RTT, and throughput followed by statistical comparison under identical failure scenarios.

References

1. Zhang W., Geng H. Improving Network Availability through Optimized Multipath Routing and Incremental Deployment Strategies. Computers, Materials & Continua, 2024, 80(1), 427–448. https://doi.org/10.32604/cmc.2024.051871.
2. Barkalov A., Lemeshko O., Persikov A., Yeremenko O., Titarenko L. Evaluation of Traffic Engineering Routing Models Based on Type of Service in Communication Networks. Electronics, 2024, 13(18), 3638. https://doi.org/10.3390/electronics13183638.
3. Xing Z., Qi H., Di X., Liu J., Xu R., Chen J., Cong L. A Multipath Routing Algorithm for Satellite Networks Based on Service Demand and Traffic Awareness. Frontiers of Information Technology & Electronic Engineering, 2023, 24, 844–858. https://doi.org/10.1631/FITEE.2200507.
4. Luo Y., Ning Q., Chen B., Zhou X., Huang L. Software Defined Network-Based Multipath State-Aware Routing with Traffic Prediction in Satellite Network. Transactions on Emerging Telecommunications Technologies, 2023, 34(9), e4819. https://doi.org/10.1002/ett.4819.
5. Sheikh M.N.A., Hwang I.-S., Raza M.S., Ab-Rahman M.S. A Qualitative and Comparative Performance Assessment of Logically Centralized SDN Controllers via Mininet Emulator. Computers, 2024, 13(4), 85. https://doi.org/10.3390/computers13040085.
6. FRRouting Project. FRRouting Releases: FRRouting 10.7.1, 25 August 2026. https://frrouting.org/release/ (accessed 12.09.2026).
7. Open vSwitch Project. Open vSwitch 4.0.0 Release, 17 August 2026. https://www.openvswitch.org/download/ (accessed 12.09.2026).
8. ESnet. iperf3 Release Notes: iperf 3.21, 9 April 2026. https://github.com/esnet/iperf/blob/master/RELNOTES.md (accessed 12.09.2026).