A Comparison of Discrete Event Simulator and Real-Time Emulator for Mobile Ad Hoc Network

Jingzhi Wang, Penghui Hu, Yixin Zhang(✉), and Jian Wang(✉)

School of Electronic Science and Engineering, Nanjing University, Nanjing 210023, China
{zyixin,wangjnju}@nju.edu.cn

Abstract. Network research requires testing tools to evaluate new protocols and algorithms. Since implementing and verifying these protocols and algorithms on real network devices is time-consuming and costly, widely-used tools often run in virtual environments to reproduce the behavior of real networks. These tools can be divided into simulators and emulators. Unlike simulator, which is mature and widely used in network research, emulators are relatively new and still developing, especially those based on the latest virtualization technologies. The most common application scenario of an emulator is Software Defined Network (SDN). Still, it has a great development prospect in Mobile Ad hoc Network (MANET) research. In an ad hoc network, nodes are equivalent; each node acts as a router. Large-scale MANET simulations with a simulator often take a long time, while real-time emulators do not. However, although the strong expansion capability of the newly proposed emulator is suitable for MANET, it lacks the mobility and scenario model necessary. In this paper, the discrete event simulator and real-time emulator are thoroughly compared, the parameters and performance of the classical simulator and emulator are compared, the development trend of the emulator is summarized, and open issues are proposed.

Keywords: Network Simulator · Emulator · MANET · SDN · Docker

1 Introduction

As network technology evolves, researchers pursue simulation tools to evaluate and analyze newly proposed architectures, algorithms, and protocols. The simulation tools can verify the throughput, end-to-end delay and other performance parameters of a protocol or algorithm without implementing a complex and large-scale network prototype system. Network simulation tools typically support network modeling with different topologies by defining the behavior of communication channels and network nodes [1].

There are many simulation tools on the market, and they can be classified differently. They can be divided into discrete event simulators and real-time emulators from the implementation principle. Also, they can be divided into general tools and special tools from the applicable field, and they can be divided into open source tools and commercial

X. Jiang (Ed.): MLICOM 2022, LNICST 481, pp. 63–74, 2023.
https://doi.org/10.1007/978-3-031-30237-4_6

tools. This paper mainly discusses the differences between discrete event simulator and real-time emulators and their capacities for MANET research.

Mobile Ad hoc Network (MANET) is a wireless Network in which nodes can communicate with each other through radio. The nodes are equivalent to each other and have mobility. MANET does not rely on pre-existing infrastructure communications and can be applied to tactical communications, emergency medical rescue, rescue, and disaster relief [2]. Each node of a MANET is equivalent to a router in a traditional network, and each node in the network needs to perform forwarding and run routing protocols. Existing research on MANET covers the whole protocol stack from top to bottom, including the physical layer, MAC layer, network layer and node mobility model. It is usual to study the signal model in the mobile scenario and the fading resistant physical layer protocol for the physical layer. And for the MAC layer, the access problem under unstable communication links and mobile scenarios needs to be studied. The routing decision problem in dynamic changing topology in the network layer is a research hotspot. Classical MANET routing protocols include active, reactive and hybrid routing protocols. Active routing protocols include Optimized Link State Routing Protocol (OLSR), the Destination Sequenced Distance Vector (DSDV), etc. Reactive routing protocols include the Temporally Ordered Routing Algorithm (TORA), Ad Hoc on-demand distance vector routing (AODV), etc. Hybrid routing protocols include border gateway protocol 1 (BGP) and zone routing protocol (ZRP) [3]. When studying the above MANET protocols at each layer, the network simulators and the emulators are adopted to verify their behavior and analyze their performance.

The remaining part of this paper is organized as follows: In Sect. 2, we compare the simulator and emulator, analyzing their implementation principles and characteristics. Section 3 surveys some classical network simulators, and Sect. 4 introduce two well-known emulators. Section 5 comprehensively compares the above tools from the aspects of running speed, scalability, protocol support, etc. Section 6 discusses the recent development direction of network emulators and proposes several problems to be solved in MANET emulation. In the last section, we summarize the whole paper and propose some suggestions for MANET emulator development.

2 Fundamentals of Discrete Event Simulation and Real-Time Emulation

The discrete-event simulation tool uses software programs to simulate the operation of real devices and their interactions, such as NS-3 [4], OPNET [5], OMNET++ [6], etc. Compared to running real operating systems and applications on real devices, running simulators on PC is low-cost, flexible, controllable, and scalable. They run in virtual tick time instead of real-time. However, if the model used in the simulator is not correct enough, the results will be biased from the actual experimental results. Real-time online emulators, such as Mininet [7] and EstiNet [8], are generally based on virtualized nodes and run actual applications. Most classical network simulation tools are based on the discrete event simulation paradigm.

A discrete event simulator simulates the random events of nodes in the network, as shown in Fig. 1. For example, when a packet starts to send, the simulator puts it into the

event queue according to the time. And then, the event in the queue will be processed sequentially, according to the actual network behavior. When an event in the queue is finished, the global variable is changed to trigger the next event.

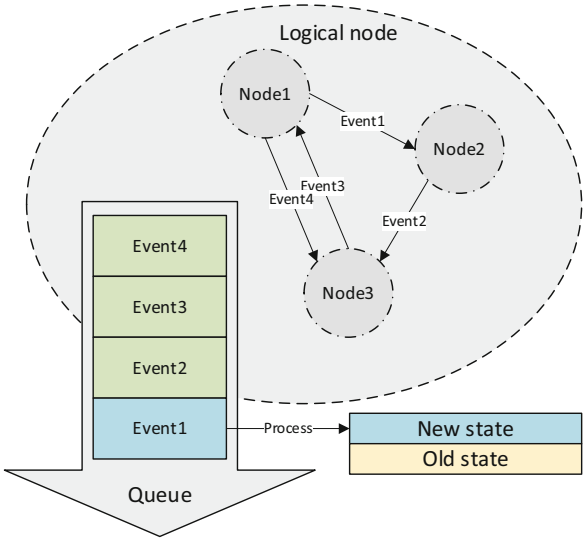

Fig. 1. Discrete event simulation

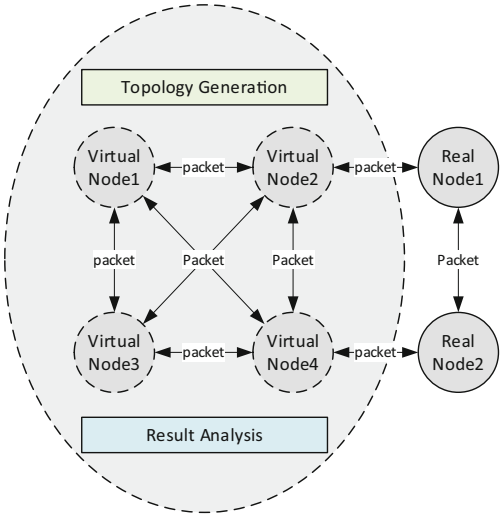

Fig. 2. Realtime emulation based on virtualization

Real-time emulation tools are usually based on virtualization technology and resource isolation technology. Real applications can flow in virtual nodes to send and

receive data packets to each other in real time, mimicking the behavior of a real network, as shown in Fig. 2. Emulators are most commonly used in SDN simulation. It can easily build network topology, control the behavior of each virtual node through a centralized controller, and conduct real traffic tests. Most of the above tools can be run on one or more computers and use algorithmic models to simulate the wireless channel. The accuracy depends on the algorithm defined in the software.

In the existing research, discrete event simulators are generally adopted for MANET research. These simulation tools provide the modules of each layer protocol. As the number of nodes increases, because events are processed one by one in the discrete event simulator, a large number of nodes will bring an exponentially growing event queue, and the processing time required by the simulator will increase significantly, which may be unacceptable. However, the emulator reproduces the behavior in the network in real time. With the network scale increasing, it will occupy more RAM and CPU resources, but the running time will not increase significantly. Therefore, the emulator has more time advantages in large-scale MANET simulation. At the same time, with the rise of SDN, SD-MANET is also proposed to design the cross-layer MANET protocol for improving the performance of MANET [9]. Traditional Simulator has limited support for these aspects. Considering the above, applying emulators in MANET research may be the future development trend.

3 Discrete Event Simulators

The large number of network simulation tools brings difficulties to the selection of researchers and network engineers. There have been some works on the comparative study of simulation software. But no protocols and mobile models are mentioned in [10]. Paper [11] investigates the common simulators and discusses the protocols and mobility models they support. Some simulation tools mentioned above have developed new features and provided more MANET support in recent years.

This section introduces several well-known simulators. Simulators for other specific scenarios, such as SensorSim, and GrooveNet, are not included in the description. Some older simulators that have not been updated for a long time or are inconvenient to use are also out of the picture, such as JiST/SWANS, GlomoSim, and QualNet.

3.1 Ns-2&NS-3

The Network Simulator project originates from the US military's Real Simulator Project [12]. NS-2 [13], the second version of the NS project, is an open-source object-oriented discrete event Simulator. NS-3 is the third version of Network Simulation, started in 2006. NS-3 is not an upgraded version of NS-2 but a re-development of core functional modules based on C++. Furthermore, the API and framework of NS-3 are not completely consistent with NS-2. NS-3 also provides an optional Python extension interface, which greatly reduces the programming complexity for users [14].

The simulation layers of NS-3 are similar to the OSI model. It encapsulates data in the device, network, transport, and application layers sequence. The process of the packet transmission is simulated according to the protocol of the classes in C++. NS-3

is an event-driven simulator. Events are triggered by network nodes, the core module extracts and processes events in the queue, and the interaction of pointers completes the packet forwarding process, so it has a fast running speed and less memory consumption. The tracing system of NS-3 is also convenient for users to analyze the output using software such as Wireshark [4].

Currently, NS-2 has stopped updating. NS-3 is still developing rapidly. Although the model and protocol library are not as perfect as NS-2, it provides a better user interface and higher performance. The disadvantages of NS-3 include: (1) limited GUI support, which requires the development of additional plug-ins; (2) inadequate model libraries and protocol libraries, which require user customization.

3.2 OMNET++

OMNET++ is also an object-oriented discrete event network simulator, and the latest version was released in 2022. It is not only designed to simulate a network but can theoretically be used to simulate any system that transmits information to each other.

OMNET++ has been applied in several problem areas, such as communication network modeling, protocol simulation, network queueing, modeling of multiprocessors and other distributed hardware systems, hardware architecture verification, and performance evaluation of complex software systems [6].

OMNET++ has good GUI support, which makes it easy to modify models and observe results under the graphical interface. OMNET++ uses C++ programming, of which the architecture can realize the parallel simulation and is scalable. OMNET++ has the following disadvantages: (1) it does not organize simulation levels according to OSI standards; (2) it has limited support for communication protocols; (3) it has poor support for mobility modeling; and (4) it is not easy to analyze performance metrics.

3.3 NetSim

NetSim [11] is used for network design, planning, research, and development, supporting the simulation of many networks, such as TCP, IP, WIMAX, WLAN, wireless sensor networks, and MANET. And it also supports the simulation of 802.11, LTE, 5G and other protocols. NetSim is also a discrete event simulator. TECOS developed it with Indian Institute of Science and enabled end-to-end, full-stack, packet-level event simulation. An advanced feature of NetSim is the ability to debug custom code at various levels, including each packet interval, which avoids a time-consuming programming process. NetSim has an excellent GUI, friendly interface, programmability, and built-in data analysis framework. As commercial software, it has detailed documentation and user tutorials. The downside of NetSim is that com-mercial software costs money, and the code is not open-source.

4 Real-Time Emulators

Real-time emulators are often called SDN emulators. In fact, both of emulators and simulators can be used to conduct MANET research. This section presents two widely used emulators.

4.1 Mininet

Mininet [15] is a virtual network simulation platform developed by Stanford University. It started in 2010 and has been widely used in SDN emulation. Mininet can be developed to validate various communication protocols. Because the code on the Miniet node is a Linux executable, its protocols can be better ported to hardware.

Mininet creates virtual hosts using the process-based virtualization method and network namespace mechanism to separate network interfaces, routing tables and ARP tables of different virtual hosts. Namespace and group have been supported since Linux version 2.2.26, and these features also support container development. The virtual switch in Mininet is a software OpenFlow switch named "Open vSwitch". The link between the virtual host and the virtual switch is achieved by using the virtual Ethernet pair provided by the Linux kernel. The Linux kernel protocol stack parses data sent from one virtual host to another [16]. Mininet uses the Python API to set up network topologies.

Mininet has good scalability. Its lightweight structure makes it easy to simulate a network with thousands of nodes on a single host and verify and test a complete network system, including hosts, links, switches, etc. Mininet can also be used to verify the routing algorithm of the MANET network. However, Mininet has the following disadvantages: (1) limited protocol support and dependence on Linux kernel protocol stack, (2) the node lacks mobility modeling support, (3) CPU can not accurately schedule the sequence of the virtual host, virtual switch and controller, and its simulation results are not accurate and difficult to reproduce.

4.2 EstiNet

EstiNet is a novel commercial simulator. EstiNet originated from NCTUns and became commercial software in 2011, which can run network emulation at high speed without losing fidelity. EstiNet's network simulation layer includes the physical, MAC, network, transport, and application layers, which correspond to the actual network structure [17].

Unlike the virtual nodes in Mininet that completely depend on the Linux protocol stack, EstiNet uses a method called "CR" kernel re-entry. EstiNet captures the IP packets sent by the Linux kernel to the lower layer through the tunnel network interface and sends them to EstiNet's simulation engine. Each virtual node has its simulation protocol stack in the simulation engine, including the MAC and physical layers, which can simulate parameters such as link delay and bandwidth. As a commercial software, EstiNet is much better than Mininet in terms of accuracy and repeatability [18]. At the same time, EstiNet also has good scalability and superior performance in large-scale network simulation.

EstiNet is mainly used in SDN to emulate the interaction between the Openflow switch and the Openflow controller. However, this architecture is also very suitable for MANET emulation. EstiNet also provides examples of MANET protocol emulation. The disadvantages of EstiNet include: (1) commercial software with no open source and (2) limited support for MANET and node mobility.

5 Comparison and Analysis

This paper investigates the five well-known simulation and emulation tools, each with its characteristics, applicable fields, and shortcomings. Table 1 summarizes the basic characteristics of these tools.

Table 1. Comparison List of Basic Characteristics

Name	NS-3	OMNET++	NetSim	Mininet	EstiNet
Type	simulator	simulator	simulator	emulator	emulator
License	Open Source	Open Source	Commercial	Open Source	Commercial
OS	windows, linux, macOS	windows, macOS	windows	linux	linux
Program Language	C++, python	C++	C, JAVA	python	–
Released	2008	1997	2002	2010	2011
Latest	2022	2022	2021	2021	2021

Table 2. Comparison List of Performance and Supported Network

Name	NS-3	OMNET++	NetSim	Mininet	EstiNet
Memory Usage	excellent	poor	good	good	excellent
Velocity	Good	average	average	good	good
scalability	excellent	good	good	excellent	good
User Guide	Good	good	excellent	good	average
Supported Network Scenario	Ethernet, 4G, 5G, Ad hoc, mesh, WIFI, VANET, MANET	5G, Ad hoc, WiFi, Ethernet, VANET, MANET, LTE	5G, WiFi, Ad hoc, mesh, MANET, VANET, IOT/WSN, SDN	SDN, Ad hoc, MANET	SDN, WiFi, Ad hoc, MANET, VANET

Literature [19] evaluated the performance of mainstream simulation tools at that time, and literature [20] evaluated the performance of discrete event simulation tools. The simulation tools mentioned in this paper are compared based on the existing work and the author's practice. Table 2 summarizes the performance differences of the application scenarios for these simulation tools.

In MANET research, simulation tools are needed to support the wireless channel model, standard communication protocol (such as 802.11), MANET routing protocol

Table 3. Comparison List of Protocol and Mobility Models

Name	NS-3	OMNET++	NetSim	Mininet	EstiNet
Radio Propagation Models	Friis Loss Model, 3GPP Loss Model, Random Loss, etc.	Rician Fading, Rayleigh Fading, Free space path loss, etc.	Friis Loss Model, Free Space, Log Distance, COST231, HATA, indoor, Rayleigh, Nakagami	Friis Loss Model, Log-Distance Loss Model, Two-Ray Ground Loss Model, etc.	Friis Loss Model, Two-Ray Ground Loss Model, Rician Fading, Rayleigh Fading, etc.
Device (MAC&PHY)	3GPP, LTE, IEEE 802.11, Ethernet	Ethernet, IEEE 802.11, 5G, 3GPP, LTE	5G, LTE, IEEE 802.11 a/b/g/n/ac/e/p, Ethernet	IEEE 802.11	5G, Ethernet, IEEE 802.11 a/g, LTE, 3GPP
MANET Routing	AODV, TORA, DSR, DSDV	AODV, DSDV, DYMO, GPSR	DSR, AODV, ZRP, OKSR	BATMAN OLSR, BABEL	OLSRD2
Internet Protocol Stack	TCP, UDP, IPv4, IPv6	TCP, UDP, IPv4, IPv6, OSPF, BGP	TCP, IP, UDP	TCP, UDP, IP	TCP, UDP, IPv4, IPv6, OSPF, RIP, BGP,
APP	HTTP, FTP, TELNET	HTTP, FTP, TELNET, DNS	HTTP, FTP, P2P, Video Traffic, Email, Custom Model	User Defined	HTTP, FTP, DHCP, NAT, VPN, DNS, SSH, etc.
Mobility models	Random, Constant Acceleration, Constant Acceleration	Tractor, Random, GaussMarkov, Chiang Mobility, Deterministic Model	Random way point model, Group mobility, File Based Mobility, etc.	Random Walk, Gauss Markov, Truncated Levy Walk, etc.	Random

and mobility model. Table 3 investigates the protocols supported by various simulation tools, including application layer protocols. The content in Table 3 includes the functions of this software and the plug-in functions provided by their communities. The perfect ecology is also an important part of the simulation tool.

Through comparative studies, we find that the simulators are mature but limited performance of time for large-scale MANET, while the emulator has limited model support but has great potential.

With the continuous development of technology, network simulation technology is also developing. Regarding requirements, the development goals of network simulation are faster simulation speed, more simulation nodes, more accurate simulation results, easier use and easier to expand the software architecture.

1. With the development of parallel computing and distributed theory, more and more simulation software began to develop distributed processing and parallel simulation mechanisms to improve the computing speed of large-scale simulation.

2. The design of modularity is emphasized constantly. Network simulation tools should abstract hierarchical modules that conform to the OSI network model, improve logicality and readability, make simulation event mechanisms more similar to the real network, and improve the accuracy and credibility of simulation results.

3. Better user support and friend GUIs. The tools tend to abandon specialized programming languages in favor of widely used general-purpose programming languages such as C++, Python, and JAVA.

4. Various extension packages and tools are developed on open-source frameworks to meet different needs. For example, node mobility models and scene modeling are required in MANET, while many traditional simulation tools lack these functions. The open source community can work closely with users and update tools. For example, Mininet-WiFi [21], provided by the Mininet community and INET framework [22], provided by the OMNET++ community, offer MANET support.

Despite these developments, the architecture of the discrete event simulator itself is limited. For large-scale MANET simulations, the interactions between the nodes cause the simulator to perform long-time computations before the result is obtained. However, after the above summary comparison, we can find that the current emulator is still not very mature. Compared to the rich simulation tools, the only widely used open-source emulator is Mininet. Emulators are still evolving rapidly, and we'll look at their prospects and current problems in the next section.

6 Future Enhancement of Emulators for Large-Scale MANET

For large-scale MANET, discrete event simulators cost too much time, while the existing real-time emulators cannot provide rich models and the simulation results are not accurate enough. Therefore, there is still room for further development of MANET emulation tools, such as optimizing memory usage, providing more mobility models and scene modeling for real-time simulation tools, and developing emulation architectures with higher fidelity.

6.1 High Performance Open Source Emulators Based on Container

With the development of virtualization technology, container technology and cloud computing have been widely used, the container is suitable to act as the node of a network emulator, so the real-time simulation tool based on the container is also proposed. Such tools have better scalability and lower resource consumption.

Docker [23] is currently the most popular container technology, and real-time online simulators based on Docker have appeared. NestedNet [24] is an emulation environment for hierarchical SDN systems, and it also provides an example MANET with 12 nodes. The inter-node throughput in NestedNet can reach 32 Gbps. DockSDN [25] is a SDN emulator based on a docker container, and its performance is better than Mininet. Kathara [26] is an open-source real-time simulation system, and it is designed to be used in the teaching of standard computer networks.

However, the scale of emulator nodes generally does not exceed hundreds, and no MANET emulators can scale up to thousands of nodes horizontally. The performance of a single computer is limited, and the software running on it cannot strike a balance between simulation speed and memory footprint. One possible idea is to build distributed real-time emulators. In the Internet industry, thousands of containers can be managed using server clusters and Kubernetes [27] tools to deploy microservices in containers. The emulator can learn from the successful experience of the data center. A high-performance container overlay network is first required to build a distributed emulator. Container networks can provide a virtual layer on top of the real network with vnet and OvSwitch [28]. We need a tool to easily define the topologies of a virtual overlay network for real-time emulation. Then we can define the link parameters during the transmission of the virtual network to emulate the scenario in the real MANET.

6.2 High Fidelity Emulators with Physical Layer Emulation Capability

So far, all the emulators we have discussed are software running on single or several computers, and they can only provide link-level simulations. For wireless channel level emulation, we inevitably need to integrate simulation with emulation, as Mininet-WiFi and EstiNet do. This requires software to compute the result when packets transmit between nodes, which will destroy real-time performance.

DARPA has developed the Colosseum, an emulation platform that runs on a cluster of servers. Colosseum has real RF systems for traffic and high-precision wireless channel simulation based on FPGA and USRP (Universal Software Radio Peripheral). The Colosseum system provides both packet-level emulation in software and IQ radio-level emulation. It's a semi-physical emulator, and it builds a 256×256 RF channel simulator that can compute and simulate more than 65000 channel interactions between more than 256 wireless devices in real time. The Colosseum is well suited for MANET emulations. However, the Colosseum relies on high-performance servers and expensive radio frequency devices, which are more complex. The Colosseum only supports a fixed and limited number of nodes. It is necessary to design an emulation environment that provides both link-level and IQ-level emulation, and the number of nodes can be easily extended.

6.3 High Scalability Emulator with Real Devices and Virtual Nodes

Another issue that needs to be addressed is an emulating interface with real devices. Existing simulators can only provide limited interactive support for real devices, such as NS-3 and OMNET++. However, emulators based on containers can provide a more integrated heterogeneous emulation method. The virtual nodes simulated by containers should be equivalent to the physical nodes in the virtual topology. They should be able to send and receive protocol packets in real time with each other.

7 Conclusion

The discrete event simulator has a mature simulation paradigm and a lot of early research, while the real-time emulator has better scalability but lacks the MANET model. Open source simulators have a better community ecosystem and a richer model base, and commercial simulators have better user support, prettier GUIs, and better performance. NS-3 would be a better choice in open source simulators, and its performance is the best in discrete event simulators. This paper investigates and summarizes various simulators and emulators, which will help researchers in the field of MANET to select appropriate tools for protocol and algorithm research. We also found that none of the existing emulators can fully meet the research needs in the field of MANET, and a new emulator may be designed to meet all the requirements of network researchers in the future. The new emulator must support all types of networks, protocols, and mobility models. It must be extensible, scalable and provide good emulation support with real devices.

Acknowledgements. This work is supported by Science and Technology Project of State Grid Shanghai Municipal Electrical Power Company.
(SGSHXT00YJJS2100140).

References

1. Fujimoto, R.M., Riley, G.F., Perumalla, K.S.: Network Simulators, pp. 5–17. Springer, Cham (2007)
2. Bang, A.O., Ramteke, P.L.: MANET: history, challenges and applications. Int. J. Appl. Innov. Eng. Manage. (IJAIEM) **2**, 249–251 (2013)
3. Ramphull, D., Mungur, A., Armoogum, S., Pudaruth, S.: A review of mobile ad hoc network (MANET) protocols and their applications. In: 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 204–211 (2021)
4. Gustavo, C.: NS-3: network simulator 3. In: UTM Lab Meeting, April 2010
5. OPNET NETWORK SIMULATOR. https://opnetprojects.com/opnet-network-simulator/. Accessed 29 Aug 2022
6. What is OMNET. https://omnetpp.org/intro/. Accessed 29 Aug 2022
7. Mininet overview. http://mininet.org/overview/. Accessed 29 Aug 2022
8. Wang, S., Chou, C, L., Yang, C, M.: EstiNet openflow network simulator and emulator. IEEE Commun. Mag. **51**(9), 110–117 (2013)
9. Kafetzis, D., Vassilaras, S., Vardoulias, G., Koutsopoulos, I.: Software-defined networking meets software-defined radio in mobile ad hoc networks: state of the art and future directions. IEEE Access **10**, 9989–10014 (2022)

10. Manpreet, Malhotra, J.: A survey on MANET simulation tools. In: 2014 Innovative Applications of Computational Intelligence on Power, Energy and Controls with their impact on Humanity (CIPECH), pp. 495–498 (2014)
11. Dorathy, I., Chandrasekaran, M.: Simulation tools for mobile ad hoc networks: a survey. J. Appl. Res. Technol. 437–445 (2018)
12. Keshav, S.: REAL: a network simulator. Technical report. UCB/CSD-88-472, EECS Department, University of California, Berkeley (1988)
13. The Network Simulator - NS2. http://www.isi.edu/nsnam/ns/. Accessed 29 Aug 2022
14. Sharma, P., Gupta, R., Sharm, S.: A comparative study of NS-2 and NS-3. Int. J. Sci. Res. Dev. **2**(3), 428–431 (2014)
15. Lantz, B., Heller, B., McKeown, N.: A network in a laptop: rapid prototyping for software-defined networks. In: ACM Hotnets 2010, Monterey (2010)
16. Pfaff, B., Pettit, J., Koponen, T., Amidon, K., Casado, M., Shenkerz, S.: Extending networking into the virtualization layer, January 2009
17. Wang, S., Chou, C., Lin, C.: The design and implementation of the NCTUns network simulation engine. Simul. Model. Pract. Theory **15**(1), 57–81 (2007)
18. Wang, S.: Comparison of SDN OpenFlow network simulator and emulators: EstiNet vs. Mininet. In: 2014 IEEE Symposium on Computers and Communications (ISCC). pp. 1–6 (2014)
19. Yang, L.-Y., Li, Y.-K., Han, S.-S., Wang, X.: Network simulation and computational experiment platforms: the overview and prospect. J. Command Control **4**(4), 281–290 (2018)
20. Admassu, T.: Performance analysis of emulated software defined wireless network. Indonesian J. Electr. Eng. Comput. Sci. **16**, 311 (2019)
21. Fontes, R.R., Afzal, S., Brito, S.H., Santos, M.A., Rothenberg, C.E.: Mininet-WiFi: emulating software-defined wireless networks. In: 2015 11th International Conference on Network and Service Management (CNSM), pp. 384–389. IEEE (2015)
22. INET Framework. https://inet.omnetpp.org/. Accessed 29 Aug 2022
23. Docker Homepage. https://www.docker.com/. Accessed 29 Aug 2022
24. Zhang, X., Prabhu, N., Tessier, R.: NestedNet: a container-based prototyping tool for hierarchical software defined networks. In: 2020 International Workshop on Rapid System Prototyping (RSP), pp. 1–7 (2020)
25. Petersen, E., To, M.A.: DockSDN: a hybrid container-based SDN emulation tool. In: 2020 IEEE Latin-American Conference on Communications (LATIN-COM), pp. 1–6 (2020)
26. Scazzariello, M., Ariemma, L., Caiazzi, T.: Kathara: a lightweight network emulation system. In: NOMS 2020 IEEE/IFIP Network Operations and Management Symposium, pp. 1–2 (2020)
27. Bernstein, D.: Containers and cloud: from LXC to Docker to Kubernetes. IEEE Cloud Comput. **1**(3), 81–84 (2014)
28. Dua, R., Kohli, V., Konduri, S.K.: Learning Docker Networking. Packt Publishing, Birmingham (2016)