# Bringing Research Infrastructure into the Networking Classroom at a Public Teaching College

Ilknur Aydin
Farmingdale State College
Farmingdale, NY, USA
aydini@farmingdale.edu

Fraida Fund
New York University
Brooklyn, NY, USA
ffund@nyu.edu

## Abstract

Public research testbeds make sophisticated networking and cloud computing capabilities available to educators and students at non-R1 and teaching-focused institutions without requiring those institutions to own or maintain the infrastructure. However, most reports on educational use of networking testbeds come from research-intensive institutions. This experience report examines adoption in a required undergraduate computer networking course at Farmingdale State College, a public teaching-focused institution where many students commute, work while enrolled, and have limited prior exposure to Linux command line tools or research infrastructure. Since Fall 2022, one instructor has used public testbed-based labs across 16 course sections serving 373 students. We describe this experience and its outcomes, with a focus on the conditions that made adoption in this setting feasible: mentoring, reusable lab materials, instructor-created scaffolding, detailed student-facing instructions, and platform choices that reduce unproductive friction.

## CCS Concepts

• **Social and professional topics → Computing education**; • **Networks**;

## Keywords

Computer Science Education, Computer Networks, CloudLab

**ACM Reference Format:**
Ilknur Aydin and Fraida Fund. 2026. Bringing Research Infrastructure into the Networking Classroom at a Public Teaching College. In *Proceedings of SIGCOMM Education Workshop 2026: Networking Education for the AI Generation (A4NE '26)*. ACM, New York, NY, USA, 7 pages.



## 1 Introduction

Over the last 15 years, NSF-supported research platforms for networks and distributed systems (including, but not limited to: GENI [6], CloudLab [9], Chameleon [22], and FABRIC [4]) have changed how researchers in these fields run experiments. Public testbeds make it possible to reserve real compute, network, and cloud resources; reproduce other researchers' experiments; and evaluate systems at a scale that most individual campuses could not maintain on their own. This shared infrastructure has a democratizing effect for research: investigators at institutions without large local systems labs can still conduct verifiable, repeatable experiments on research-grade infrastructure.

The same democratizing effect should also extend to computer science and engineering education. Through educational use of public testbeds, students at institutions with limited local infrastructure can gain hands-on experience with networks, distributed systems, and cloud platforms. This access is especially relevant in computer networking courses, where students must connect layered protocol abstractions to behavior they can observe in running systems. Textbooks, packet traces, and simulators help, but students also benefit from working with real hosts, remote access, routing behavior, congestion control, and distributed systems they can configure themselves.

This hands-on work has only become more important as students gain access to AI tools that can generate explanations, code, and answers to conventional homework questions. Testbed-based labs ask students to configure resources, collect measurements, and explain behavior they observed in their own experiment. AI tools may help students interpret errors or terminology, but they do not replace the experience of operating remote systems and connecting network behavior to evidence from a live run.

However, while thousands of students to date have gained hands-on experience with networks and distributed systems through educational use of testbeds [1, 7, 18–20, 27, 29, 31–34, 36–38], most reports on this use come from research-intensive institutions. This paper instead reports on the use of public testbeds in an undergraduate networking course at Farmingdale State College, a public teaching-focused college. The goal of this paper is to provide a practical blueprint for educators at similar institutions, who want to bring authentic networking infrastructure into their classrooms. We describe the institutional context, the path by which the instructor adopted public testbeds, the supports that helped both the instructor and the students succeed, and the barriers that remained. We also discuss positive outcomes of this experience for both the instructor and the students. The experience points to three broader lessons: research infrastructure can work in teaching-focused settings, productive friction is context dependent, and infrastructure and materials may require mentoring and institutional fit before they become adoptable.

## 2 Related Work

Prior reports on the use of public research testbeds for hands-on computing education span several topic areas. In computer networking courses, instructors have used GENI and FABRIC to support experiments with network protocols and distributed systems [18–20, 32, 34, 37]. In network security courses, instructors have used CloudLab and GENI-based environments for network security labs [29, 31–33, 36], in addition to the specialized DETER infrastructure [27]. In cloud computing courses, instructors have used GENI, Emulab, and Chameleon for lab exercises and course activities [1, 7, 8, 38]. Chameleon has also been used to teach operational machine learning [17].

However, the overwhelming majority of these reports come from educators in institutions with a high level of research activity. One notable exception is the CyberPaths project, which develops and reports on the use of GENI-based network security lab materials at a liberal arts college [28, 29]. The goal and context of the CyberPaths project differ from ours, as the focus of that project is on the introduction of cybersecurity concepts and practices to non-technical majors in their first year of college.

This paper contributes an experience report from a public teaching-focused institution, where students used a research testbed (CloudLab) in a required undergraduate networking course. We focus on adoption conditions that are especially important in this context.

## 3 Institutional Context and Students

Farmingdale State College serves about 10,000 students. Most students commute while about 6% live on campus. The college serves a majority-minority student population, with 64% of students from underrepresented groups, including adult learners such as transfer students, second-degree seekers, and veterans. Many students balance school with employment and family responsibilities. About 73% of undergraduate students receive financial aid, and 82% work on or off campus. In recent years, Farmingdale State College has expanded its computing programs. The college launched its first master's degree programs in 2014 and received a $75M public investment in 2022 to build out capacity for computer science and information technology programs.

The Computer Systems Department offers two undergraduate programs. The Computer Programming and Information Systems program has served students for more than 25 years and enrolled more than 400 majors in 2025. The Computer Science program launched in Fall 2021, following guidance from the ACM/IEEE Computer Science Curricula, and enrolled about 300 majors in 2025. Courses are offered with a small student-to-faculty ratio (25:1) in a variety of formats, including in-person, hybrid, and online. The computer networks course is a required course for Computer Science majors and a technical elective for Computer Programming and Information Systems majors.

The goal of many students in these programs is industry employment, rather than graduate school or research. Many students have limited time outside of class. Most enter the course without prior experience using Unix-style command lines, SSH keys, file transfer, or remote machines. Testbed-based labs therefore needed to teach networking concepts while also onboarding students into the infrastructure itself, and without placing excessive demands on students' time.

## 4 Faculty Entry Point

The instructor is a computer systems professor at Farmingdale State College with about 15 years of academic experience and prior research experience in computer networking. At Farmingdale State College, she has a heavy teaching load (4 + 4 + extra summer teaching). Her first sustained exposure to public testbeds came through summer faculty residencies in 2020 and 2021 and a sabbatical residency in Fall 2021 at New York University. During this period, the instructor began collaborating with a research group that made extensive use of public testbeds, including GENI [6], CloudLab [9], and FABRIC [4], for both research and teaching.

The testbed-based teaching materials adopted by the Farmingdale State College networking course were used by the New York University research group in their own courses for many years [30], and the faculty collaborator at New York University provided mentoring to the Farmingdale State College instructor as she implemented these in her own course. These two elements - classroom-tested materials, and one-on-one mentoring by an experienced educator using those same materials - substantially lowered the barrier to adoption and made it possible for the Farmingdale State College instructor to implement testbed-based labs despite her limited previous experience with testbeds and heavy teaching load.

**Table 1: CloudLab assignment sequence.**

| Assignment | Topic |
|---|---|
| Hello, CloudLab [10] | Create and use a testbed account, instantiate a prepared experiment, identify reserved resources, access a host, and release resources when finished. |
| Hello, Linux [11] | Practice Unix-style command-line work on a remote host, including files, directories, processes, shell commands, and remote access habits used in later labs. |
| Socket Programming [13, 14] | Run and modify client/server programs across hosts, connect sockets to application-layer communication, and relate host, port, and transport-layer behavior to course concepts. |
| TCP Congestion Control [16] | Generate traffic across controlled paths, observe throughput and congestion window behavior, and understand how TCP responds to network conditions. |
| Static Routing [15] | Configure and test reachability across a multi-hop topology, inspect forwarding behavior, and connect routing table entries to packet delivery. |
| Learning Switch [12] | Observe how a switch learns MAC addresses, and how it forwards, filters, or floods a frame depending on the forwarding table. |

## 5 Course Context

The computer networks course at Farmingdale State College was designed to provide a solid foundation for undergraduate computing students. It covers the top four layers of the Internet protocol stack and closely follows Kurose and Ross's *Computer Networking: A Top-Down Approach* textbook [25]. The course combines conceptual material with hands-on activities: during a 15-week semester, students complete five Wireshark assignments (from the Kurose and Ross supplementary materials [26]) and six lab assignments that run on CloudLab, including warm-up assignments on each tool and one Wireshark and one CloudLab assignment for each network protocol layer.

Table 1 summarizes the CloudLab assignments and their role in the course. The Wireshark and CloudLab assignments played different, complementary roles. Wireshark assignments gave students packet traces and protocol fields to inspect. CloudLab assignments gave students remote hosts and network links they could control. In combination, the two tools helped students connect textbook concepts to packet-level evidence and to the behavior of running systems.

The instructor began teaching the course with testbed-based labs in Fall 2022 and has been the only instructor teaching it at Farmingdale State College since then. She has taught a total of 16 sections serving 373 students (1-3 sections per semester, with a range of 18 to 76 students per fall/spring semester). The course ran first in a post-COVID live-stream classroom format, with in-person lectures for a limited number of students and remote access for others. Later offerings used in-person format and mostly hybrid format (with one in-person meeting each week alongside online materials, including recorded videos).

## 6 Lab Design Choices and Supports

The CloudLab assignments were adopted from public teaching materials that were originally part of the GENI education ecosystem [18]. Comparable versions are now available for CloudLab, FABRIC, and Chameleon following the deprecation of GENI. As Table 1 summarizes, the lab assignment sequence includes two onboarding labs and four networking labs. The onboarding labs help students set up their testbed accounts, get to a running experiment, and learn the command-line workflow before the course asks them to reason about protocol behavior. The later labs are interleaved with Wireshark assignments so that students move between packet-level observation and testbed-based experimentation throughout the semester.

Each CloudLab assignment follows the same student-facing structure. Students start from a prepared CloudLab profile, reserve resources, log in to resources through the browser or SSH, run a sequence of commands on resources, collect output or screenshots, and answer questions that ask them to reason and explain what they observed. Each assignment includes complete step-by-step instructions on a single page; in most cases, students can execute the entire sequence of steps by copying and pasting commands from the lab instructions directly into a terminal. The grading scheme reflects this design: 80% of assignment credit comes from completing the experiment and documenting the run with screenshots or command output, while the remaining credit comes from questions that ask students to connect those observations to networking concepts.

A major focus in this adoption was to allow students in this institutional context to run experiments successfully and experience networking concepts as a concrete behavior in real systems. The learning objectives of the course were about computer networking: observing protocol behavior, connecting measurements to course concepts, and explaining what happened in a running system. Using a testbed was necessary for reaching those objectives, but it was not the primary objective of the course. Therefore, supports were designed to minimize the friction of using a testbed, in order to emphasize the intellectual work of reasoning about network behavior.

One major area where support was needed was in the onboarding process, which is often challenging (as has been observed in other work on teaching with public testbeds [37]). In this context specifically, many students entered the course with limited experience using Unix-style command lines, SSH keys, remote login, file transfer, or VNC. After the first one or two orientation labs, many students became comfortable enough to follow detailed instructions and complete later experiments. However, students who missed the early orientation work often struggled for the rest of the semester.

We also found that written instructions alone were not enough for many students. The instructor augmented the existing written lab materials with recorded walkthrough videos [2]. In these videos, the instructor completed the same steps students were expected to complete, using the same written instructions and showing the expected terminal workflow. Students could pause the video, compare their own terminal output to the instructor's output, and recover when they made mistakes. Videos are likely helpful in supporting written instructions in many contexts; walkthrough videos are used at New York University as well.

Students also occasionally ran into technical issues that were difficult to diagnose locally. In those cases, the Farmingdale State College instructor could often consult the New York University mentor. Thus, the mentoring relationship was useful not only during initial planning and adoption, but throughout the semester in case of problems that the Farmingdale State College instructor did not have the expertise to resolve.

Moving backward from classroom supports to adoption decisions, the choice of materials and platform also shaped students' success. Kapur's work [21] shows that productive friction can support learning. However, productive friction in one setting may be unproductive friction in another setting. Many students at Farmingdale State College are commuters, work while enrolled, and take the networking course because their program requires it. Their first goal is often completion. Time and cognitive effort spent parsing the instructions, instead of reasoning about the network behavior, is not necessarily productive for this student population. It was therefore a priority to reduce cognitive load in the setup process while preserving intellectual work in the parts of the lab tied to course learning goals.

Toward that end, the choice of lab material is important. The lab materials followed a do-observe-reflect model. In the do stage, students instantiated a prepared experiment and ran commands, often by copying commands step-by-step from the instructions into a terminal. This design minimized setup friction so that students could reach a working experiment. The observe and reflect stages then asked students to inspect output, interpret packet or host behavior, and connect observations to networking concepts. We treated the productive friction as the work of observing and explaining what happened, not the work of discovering how to get the experiment running.

Another important choice was the platform. Although similar versions of the lab materials are available for FABRIC and Chameleon, Farmingdale State College used the CloudLab versions because they reduced setup friction that would have been unproductive in this setting. On FABRIC and Chameleon, setting up an experiment can require either multiple clicks in an online graphical interface, where a wrong choice can derail the setup, or executing a Python notebook cell by cell. The notebook workflow is sensitive to out-of-order execution, naming choices, transient errors, and site outages, all of which require students to decide whether an error reflects their own mistake, a temporary platform condition, or an instruction they should rerun. Prior work on teaching with FABRIC reports that students found these setup issues challenging, including Jupyter-based confusion about whether to run commands in the notebook environment or in the terminal of the experimental node [37], and we observed similar confusion when working with students on FABRIC and Chameleon. These interfaces make the process of designing and setting up a new topology more visible, which can be desirable if the instructional goal is to prepare students for open-ended platform use. However, if the goal is to get students to a running experiment with minimal friction so they can focus on observing and reflecting on network behavior, we found the CloudLab user experience much smoother. We note again that what counts as productive friction in one instructional context can become unproductive friction in another.

As AI tools have become more widely available, they have not been a major source of assignment redesign pressure in this course. Students could still use AI tools to help explain terminal output, and such use would not necessarily be inconsistent with the goals of the assignment if it helped them connect observations to networking concepts. The concern is use that bypasses the intended work of running the experiment, inspecting the evidence, and interpreting the results. Current AI agents are likely capable of following lab instructions, running commands, and generating explanations from the resulting output, but to date we have not seen evidence of that pattern in this course. For this student population, the provided materials appear to remain the path with the least procedural friction through the assignments: the instructions, expected terminal outputs, screenshots, and recorded walkthroughs already provide the guidance students need to complete the experiment, while leaving the intended productive friction in the work of observing and interpreting results.

This may differ from more open-ended assignments, where students face a blank slate and the path with the least procedural friction may involve using AI tools for initial direction [5, 35]. In these labs, the starting point and a straightforward procedural path are already supplied.

## 7 Outcomes

We did not undertake any systematic investigation of outcomes, but anecdotally, students responded positively. They appreciated the chance to work with remote research infrastructure, and they especially responded to activities that made the remote systems feel concrete, e.g. using a graphical web browser in a VNC session on the remote host. They found that the testbed activities were engaging, made networking concepts concrete, and that the combination of written instructions, recorded videos, and additional support in instructor office hours made the experiments relatively painless to execute. We also observed that the quality of students' work on the labs was generally good, although it declined for a small number of students near the end of the semester, when heavier course workloads and outside responsibilities led them to rush the more involved assignments.

We also observed several positive outcomes for both students and the instructor that suggest the value of this experience beyond the course itself.

Five undergraduate students, recruited through campus bridge and mentored research-placement programs at Farmingdale State College, were able to work on independent study projects involving CloudLab and FABRIC. These students had limited prior research experience and had rarely considered graduate study or research careers. Through the course and independent-study projects, they used real hardware, practiced Unix command-line work, and handled practical troubleshooting in public testbed environments. Exposure to public testbeds gave Farmingdale State College students another vision of computing work and taught them skills they could draw on if they later reconsidered research or graduate school. Individual students sometimes attributed positive outcomes beyond the college itself (e.g. a student obtaining an IT job they had been pursuing, another student continuing to a master's program).

The collaboration also produced positive outcomes for the instructor. The ongoing relationship with the New York University research group led to a co-authored workshop paper on wireless networks and associated public dataset [3], technical presentations at New York University research group meetings, and co-mentorship of students at Farmingdale State College on testbed-based research projects. The instructor from Farmingdale State College had the opportunity to participate in the NSF Midscale Experimental Research Infrastructure Forum in 2024 as an education panelist and later joined a multi-institutional NSF cyberinfrastructure proposal as a co-PI. The instructor also developed a new wireless networking course as an upper-level technical elective, scheduled to debut in Spring 2027 at Farmingdale State College, in which students will use advanced hands-on labs across *multiple* wireless public testbeds, again building on publicly available material and mentoring from the New York University collaborator.

## 8 Limitations

The observations in this report, come from course implementation, student interactions, and instructor reflection, but not from any systematic investigation. We did not collect pre/post measures, surveys, or pass-rate comparisons that would support causal claims about learning outcomes. The contribution is an adoption report: it identifies the supports that allowed a teaching-focused institution to use research infrastructure repeatedly across 16 sections, for 8 consecutive semesters since Fall 2022. Furthermore, this contribution draws on one instructor's experience at one public teaching-focused institution, and may not generalize.

## 9 Conclusion

NSF-supported public testbeds provide researchers and students at institutions with limited local resources access to advanced networking and cloud computing infrastructure. At the same time, these platforms are built to support sophisticated, cutting-edge research experiments, which can make the learning curve steep. Three takeaways emerge from our experience, as described below.

First, **research infrastructure is not only for research institutions**: it can also work well in a teaching-focused setting when course design accounts for students' prior preparation, time constraints, and goals. For students at teaching-focused institutions, the opportunity to work with research infrastructure can be transformative and can lead them to understand computing in new ways. For instructors in this setting, it can also create a connection to the research community, support new course development, and open pathways for student research projects. In both cases, however, the opportunity becomes realizable only when support matches the local context.

For infrastructure operators, this means that education-facing access should be treated as a first-class part of the platform mission, not as a side effect of research access. Pathways that make research infrastructure usable by instructors and students outside research-intensive settings create opportunities for broader classroom adoption, earlier student exposure to research-grade systems, and new points of entry into the research community.

Second, **friction that may be acceptable or productive in one instructional context can become blocking in another**. In this course, reducing setup friction helped preserve students' effort for observing and explaining network behavior. In a setting where some students have not used SSH keys, remote Unix systems, or VNC, scaffolding determines whether students experience the platform as an opportunity or a barrier. Platform-onboarding lab assignments, written instructions, recorded demos, and troubleshooting support can prevent unfamiliar tools from becoming a semester-long obstacle. Course design should also reflect students' lives and goals: many students at commuter public institutions work while enrolled and focus on near-term industry employment. Instructors should connect testbed work to practical skills, show how command-line and remote systems experience supports industry roles, and pace labs so students have time for reflection.

For developers of reusable educational resources, it is important to note that a working experiment is not the same as one that is ready to adopt in any classroom. Materials produced as part of infrastructure or training projects often provide an executable experiment, but making that experiment adoptable in a setting with minimal tolerance for procedural friction requires additional pedagogical design, testing with students, and revision over time to identify points of friction and reduce them.

Third, **infrastructure availability and reusable materials are necessary, but not sufficient, conditions for adoption** in some settings. For instructors, public testbeds offer a powerful way to bring authentic research infrastructure into the classroom, but adoption can be daunting unless materials are ready for the realities of their institutional context. For teaching-focused institutions, reusable lab materials make adoption practical. Instructors often balance high teaching loads, advising, service, and curriculum development. Materials that are already "classroom-ready" can make the use of public infrastructure feasible, so that instructors can spend local preparation time on student-specific scaffolding rather than reconstructing the lab material itself. However, "classroom-ready" is context dependent: materials that work well for a graduate course at a research-intensive institution may still require substantial adaptation before they are appropriate for a required undergraduate course at a teaching-focused institution.

Instructor preparation and mentoring can change the adoption cost. In this course, the instructor's residencies, collaborations, and workshop participation gave her enough familiarity with GENI, CloudLab, and FABRIC to bring CloudLab into the classroom. This experience is consistent with research on innovation diffusion and cyberinfrastructure adoption, which finds that personalizing demonstrations, training, and building relationships based on trust are necessary strategies for promoting adoption of advanced cyberinfrastructure [23, 24]. Public testbed communities can support broader classroom use by offering faculty development, reusable examples, education-focused documentation, and ongoing mentoring for instructors.

## References


[1] Jay Aikat, Michael K. Reiter, and Kevin Jeffay. 2016. Education modules for networking, cloud computing, and security in systems courses. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (Memphis, Tennessee, USA) *(SIGCSE '16)*. Association for Computing Machinery, New York, NY, USA, 400. doi:10.1145/2839509.2850514

[2] Ilknur Aydin. 2026. Videos for CloudLab assignment in a computer networking course. https://www.youtube.com/playlist?list=PLZWNiOpi0-hJ5jjLzLkz6HDgvAGS8j6ja.

[3] Ilknur Aydin, Fraida Fund, and Shivendra Panwar. 2023. A data set and reference experiments for multipath wireless emulation on public testbeds. In *The 10th International Workshop on Computer and Networking Experimental Research using Testbeds (INFOCOM CNERT 2023)*.

[4] Ilya Baldin, Anita Nikolich, James Griffioen, Indermohan Inder S. Monga, Kuang-Ching Wang, Tom Lehman, and Paul Ruth. 2019. FABRIC: A national-scale programmable experimental network infrastructure. *IEEE Internet Computing* 23, 6 (2019), 38–47. doi:10.1109/MIC.2019.2958545

[5] Brett A. Becker, Paul Denny, James Finnie-Ansley, Andrew Luxton-Reilly, James Prather, and Eddie Antonio Santos. 2023. Programming is hard - or at least it used to be: Educational opportunities and challenges of AI code generation. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (Toronto ON, Canada) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 500–506. doi:10.1145/3545945.3569759

[6] Mark Berman, Jeffrey S. Chase, Lawrence Landweber, Akihiro Nakao, Max Ott, Dipankar Raychaudhuri, Robert Ricci, and Ivan Seskar. 2014. GENI: A federated testbed for innovative network experiments. *Computer Networks* 61 (2014), 5–23. doi:10.1016/j.bjp.2013.12.037 Special issue on Future Internet Testbeds – Part I.

[7] Prasad Calyam, Sripriya Seetharam, and Ronny Bazan Antequera. 2014. GENI laboratory exercises development for a cloud computing course. In *2014 Third GENI Research and Educational Experiment Workshop*. 19–24. doi:10.1109/GREE.2014.15

[8] Debzani Deb, Muztaba Fuad, and Keith Irwin. 2019. A module-based approach to teaching big data and cloud computing topics at CS undergraduate level. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. Association for Computing Machinery, New York, NY, USA, 2–8. doi:10.1145/3287324.3287494

[9] Dmitry Duplyakin, Robert Ricci, Aleksander Maricq, Gary Wong, Jonathon Duerig, Eric Eide, Leigh Stoller, Mike Hibler, David Johnson, Kirk Webb, Aditya Akella, Kuangching Wang, Glenn Ricart, Larry Landweber, Chip Elliott, Michael Zink, Emmanuel Cecchet, Snigdhaswin Kar, and Prabodh Mishra. 2019. The design and operation of CloudLab. In *Proceedings of the 2019 USENIX Conference on*

*Usenix Annual Technical Conference* (Renton, WA, USA) *(USENIX ATC '19)*. USENIX Association, USA, 1–14.

[10] Fraida Fund. 2026. Hello, CloudLab. https://teaching-on-testbeds.github.io/hello-cloudlab/.

[11] Fraida Fund. 2026. Hello, Linux. https://teaching-on-testbeds.github.io/hello-linux/.

[12] Fraida Fund. 2026. Learning switch. https://teaching-on-testbeds.github.io/blog/learning-switch.

[13] Fraida Fund. 2026. Socket programming in Java. https://teaching-on-testbeds.github.io/blog/sockets-java.

[14] Fraida Fund. 2026. Socket programming in Python. https://teaching-on-testbeds.github.io/blog/sockets-python.

[15] Fraida Fund. 2026. Static routing. https://teaching-on-testbeds.github.io/blog/static-routing.

[16] Fraida Fund. 2026. TCP congestion control. https://teaching-on-testbeds.github.io/blog/tcp-congestion-control.

[17] Fraida Fund, Kate Keahey, Cody Hammock, Marc Richardson, Mark Powers, and Michael Sherman. 2025. The cost of teaching operational ML. In *Proceedings of the SC '25 Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC Workshops '25)*. Association for Computing Machinery, New York, NY, USA, 393–400. doi:10.1145/3731599.3767385

[18] Fraida Fund and Shivendra S. Panwar. 2020. Teaching computer networks with GENI. In *Proceedings of the 2020 ACM SIGCOMM Education Workshop*. https://gaia.cs.umass.edu/sigcomm_education_workshop_2020/papers/sigcommedu20-final40.pdf

[19] James Griffioen, Zongming Fei, Hussamuddin Nasir, Xiongqi Wu, Jeremy Reed, and Charles Carpenter. 2012. Teaching with the emerging GENI network. In *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing, 1.

[20] James Griffioen, Zongming Fei, Hussamuddin Nasir, Xiongqi Wu, Jeremy Reed, and Charles Carpenter. 2013. GENI-enabled programming experiments for networking classes. In *2013 Second GENI Research and Educational Experiment Workshop*. 111–118. doi:10.1109/GREE.2013.30

[21] Manu Kapur. 2008. Productive failure. *Cognition and Instruction* 26, 3 (2008), 379–424. doi:10.1080/07370000802212669

[22] Kate Keahey, Jason Anderson, Zhuo Zhen, Pierre Riteau, Paul Ruth, Dan Stanzione, Mert Cevik, Jacob Colleran, Haryadi S. Gunawi, Cody Hammock, Joe Mambretti, Alexander Barnes, François Halbach, Alex Rocha, and Joe Stubbs. 2020. Lessons learned from the Chameleon testbed. In *Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC '20)*.

[23] Kerk F. Kee. 2024. Advancing cyberinfrastructure adoption: Practical insights from social science research on innovation diffusion. Keynote presentation at the Midwest Education and Research Infrastructure Forum (MERIF) Annual Meeting. https://drive.google.com/file/d/1p2GqdnNCK3hptjKqGwsz83DfRqWSPem4/view?usp=sharing

[24] Kerk F Kee, Bethanie Le, and Kulsawasd Jitkajornwanich. 2021. If you build it, promote it, and they trust you, then they will come: Diffusion strategies for science gateways and cyberinfrastructure adoption to harness big data in the science, technology, engineering, and mathematics (STEM) community. *Concurrency and Computation: Practice and Experience* 33, 19 (2021), e6192.

[25] James F. Kurose and Keith W. Ross. 2025. Computer networking: A top-down approach.

[26] James F. Kurose and Keith W. Ross. 2026. Wireshark labs. https://gaia.cs.umass.edu/kurose_ross/wireshark.php.

[27] Jelena Mirkovic and Terry Benzel. 2012. Teaching cybersecurity with DeterLab. *IEEE Security & Privacy* 10, 1 (2012), 73–76.

[28] Xenia Mountrouidou, Xiangyang Li, and Quinn Burke. 2018. Cybersecurity in liberal arts general education curriculum. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education* (Larnaca, Cyprus) *(ITiCSE 2018)*. Association for Computing Machinery, New York, NY, USA, 182–187. doi:10.1145/3197091.3197110

[29] Xenia Mountrouidou and Vic Thomas. 2019. CyberPaths: Cyber security labs for liberal arts institutions using the NSF Global Environment for Network Innovations (GENI). In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. Association for Computing Machinery, New York, NY, USA, 1241. doi:10.1145/3287324.3287541

[30] Shivendra Panwar and Fraida Fund. 2026. Forty years of teaching computer networks: A retrospective. In *Proceedings of the SIGCOMM Education Workshop 2026: Networking Education for the AI Generation (A4NE) (A4NE '26)*. Association for Computing Machinery, New York, NY, USA.

[31] Younghee Park, Hongxin Hu, Xiaohong Yuan, and Hongda Li. 2018. Enhancing security education through designing SDN security labs in CloudLab. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (Baltimore, Maryland, USA) *(SIGCSE '18)*. Association for Computing Machinery, New York, NY, USA, 185–190. doi:10.1145/3159450.3159514

[32] Mohamed Rahouti, Kaiqi Xiong, and Jing Lin. 2021. Leveraging a cloud-based testbed and software-defined networking for cybersecurity and networking education. *Engineering Reports* 3, 10 (2021), e12395. doi:10.1002/eng2.12395

[33] Mohamed Rahouti, Kaiqi Xiong, and Jing Lin. 2023. Developing blockchain learning lab experiments for enhancing cybersecurity knowledge and hands-on skills in the cloud. In *Computer Science and Education*, Wenxing Hong and Yang Weng (Eds.). Springer Nature Singapore, Singapore, 438–448.

[34] Vicraj Thomas, Niky Riga, Sarah Edwards, Fraida Fund, and Thanasis Korakis. 2016. *GENI in the classroom*. Springer International Publishing, 433–449. doi:10.1007/978-3-319-33769-2_18

[35] Priyan Vaithilingam, Tianyi Zhang, and Elena L. Glassman. 2022. Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI EA '22)*. Association for Computing Machinery, New York, NY, USA, Article 332, 7 pages. doi:10.1145/3491101.3519665

[36] Yongzhi Wang, Wen-Jung Hsin, and Manish Lamsal. 2022. EdGENI: Making GENI user-friendly for general computer education. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1* (Providence, RI, USA) *(SIGCSE 2022)*. Association for Computing Machinery, New York, NY, USA, 801–807. doi:10.1145/3478431.3499300

[37] Alexander Wolosewicz, Prajwal Somendyapanahalli Venkateshmurthy, and Nik Sultana. 2025. Experience report: Using the FABRIC testbed to teach a graduate computer networking course. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1* (Pittsburgh, PA, USA) *(SIGCSETS 2025)*. Association for Computing Machinery, New York, NY, USA, 1246–1252. doi:10.1145/3641554.3701923

[38] Yanyan Zhuang, Chris Matthews, Stephen Tredger, Steven Ness, Jesse Short-Gershman, Li Ji, Niko Rebenich, Andrew French, Josh Erickson, Kyliah Clarkson, Yvonne Coady, and Rick McGeer. 2014. Taking a walk on the wild side: Teaching cloud computing on distributed research testbeds. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (Atlanta, Georgia, USA) *(SIGCSE '14)*. Association for Computing Machinery, New York, NY, USA, 535–540. doi:10.1145/2538862.2538931