



# RH RQ

Bringing great research ideas  
into open source communities

## Jonathan Appavoo

*No more gatekeepers:  
Why technological ignorance is radically  
dangerous and how an open world will help*



**RISC-V extensions**

**Open research clouds get  
the skills to pay the bills**

**Protecting data privacy:  
a look in the toolkit**



**Red Hat  
Research Quarterly**

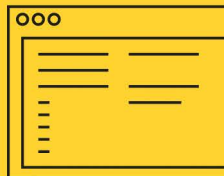
Volume 5:3 | November 2023 | ISSN 2691-5278



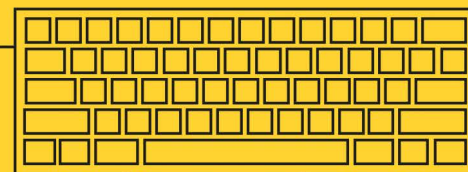
Your **research,**



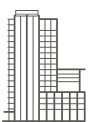
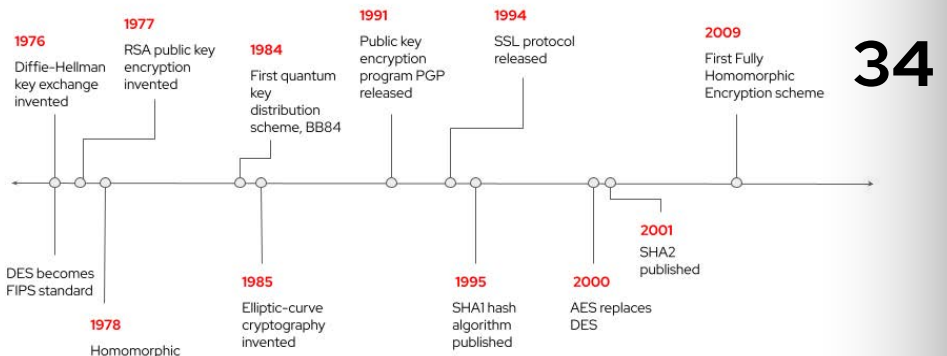
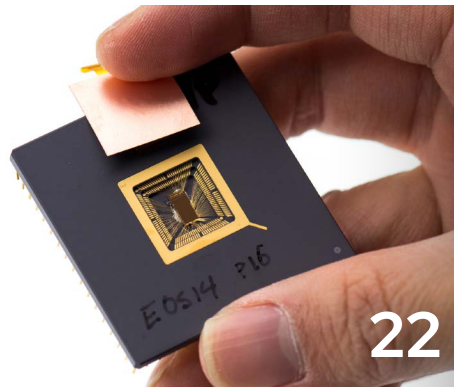
**projects**



and **education partner.**



# Table of Contents



**ABOUT RED HAT** Red Hat is the world's leading provider of open source software solutions, using a community-powered approach to provide reliable and high-performing cloud, Linux®, middleware, storage, and virtualization technologies. Red Hat also offers award-winning support, training, and consulting services. As a connective hub in a global network of enterprises, partners, and open source communities, Red Hat helps create relevant, innovative technologies that liberate resources for growth and prepare customers for the future of IT.

NORTH AMERICA  
1 888 REDHAT1

EUROPE, MIDDLE EAST,  
AND AFRICA  
00800 7334 2835  
europe@redhat.com

ASIA PACIFIC  
+65 6490 4200  
apac@redhat.com

LATIN AMERICA  
+54 11 4329 7300  
info-latam@redhat.com



facebook.com/redhatinc  
@RedHat  
linkedin.com/company/red-hat

## Departments

- 04 From the director
- 05 News: AI for everyone: NERC expands access
- 09 News: Research demo shows hardware-software co-design in action
- 38 Focus on edge: security, sustainability, and performance

## Features

- 10 No more gatekeepers: an interview with Jonathan Appavoo
- 22 RISC-V extensions: what's available and how to find them
- 30 Open research clouds get the skills to pay the bills
- 34 Protecting data privacy: a look in our current toolkit

**From the director****About the Author****Hugh Brock**

is the Research Director for Red Hat, coordinating Red Hat research and collaboration with universities, governments, and industry worldwide. A Red Hatter since 2002, Hugh brings intimate knowledge of the complex relationship between upstream projects and shippable products to the task of finding research to bring into the open source world.

## Join the research journey


by *Hugh Brock*

**T**his issue's interview subject, Boston University professor Jonathan Appavoo, stands out in just about any crowd. Unless you've spoken with him, however, you won't know that his unorthodox appearance finds a parallel in the research ideas he pursues to surprising conclusions. A perfect fit for Red Hat, he is passionate about the power of open source and open ideas to "enable flight, not create bonds." Professor Appavoo is in fact one of the most interesting people I have ever met, and we've been very happy to support him in research ideas that run the gamut from AI-driven branch prediction for CPUs to developing open textbooks and course materials using the newly christened Red Hat OpenShift AI platform. He is driven to enable other people to do things we haven't yet imagined, and we're lucky to be along for the journey.

While we're on remarkable journeys, I'm thrilled to be able to publish our in-depth survey of the RISC-V processor extension landscape. Author Rich Jones and I have been working together since before we both came to Red Hat, and I distinctly recall asking him in 2015 or so if he could maybe spend less time on building RISC-V for Fedora and more time on the work I needed him to do. Of course, he was right, and I wasn't. Today, the RISC-V landscape is critically important not only for Red Hat Research but also for Red Hat products. If you've wondered what is so different about RISC-V beyond the fact that its ISA is open source, Rich's guide to the world of RISC-V extensions will answer you. I think this is one of those articles that will become a resource for all kinds of people working in this space. If you

want to understand where RISC-V is going and what makes it so different, it's a must-read.

I'm also excited to report that our university partners BU and Harvard launched the New England Research Cloud, or NERC. NERC provides containers managed by Red Hat OpenShift, VMs managed by Red Hat OpenStack Platform, and data science training and hosting tools from Red Hat OpenShift AI. Getting these pieces put together and operating smoothly—including billing and charge-back—in a university infrastructure was no mean feat. Tzu-Mainn Chen pulls back the curtain to show how we and the university IT folks we worked with pulled it off. We'd like this to be a pattern for other research infrastructure installations; if you'd like to know more, Mainn will be glad to point you to the open source code repos with the various recipes.

Of course, research using a cloud like the NERC requires a lot of data, and it is more and more likely that some of that data will be affected by privacy constraints. Fortunately, interesting research avenues are opening up in the data privacy and data sovereignty space. Whether you're trying to account for how much privacy you have lost, use a non-trustworthy computing platform without worry, or do research on disconnected data sets, there is probably a software solution out there that will help you. Red Hat Research staff writer Gordon Haff has put together an excellent survey of what's available and what is happening in research in each space. Although there's no such thing as anonymized data, at least there are ways to share some things without compromise. 



## AI for everyone: NERC expands access

A partnership between Red Hat and the MOC Alliance is making a low-cost, open production cloud for AI research available to non-commercial users.

by *Shaun Strohmer*

**M**uch of the innovation in AI is happening in academic institutions. However, applying the potential of artificial intelligence and machine learning to the major social and environmental challenges of our time requires a volume of data and level of computational power not available to most academic researchers today. Currently, only the most well-resourced companies and institutions have access, limiting the diversity of ideas and contributing to the problem of systemic biases in AI.

Red Hat Research and the [MOC Alliance](#) are working to address the challenge of AI research access through the [New England Research Cloud \(NERC\)](#) and [Red Hat OpenShift AI](#). The NERC platform now offers AI-as-a-service to researchers and institutions for non-commercial use at a significantly lower cost than what is available from standard cloud providers. OpenShift AI provides a fully supported environment where data scientists

and researchers can rapidly train, deploy, and monitor AI/ML workloads. With this addition, NERC levels and expands the playing field for AI research by charging researchers only the real costs of compute, support, and storage.

The open nature of NERC will also enable rapid innovation by a broad community as AI evolves. As part of the MOC Alliance, which also includes the [Red Hat Collaboratory](#) at the [Hariri Institute](#) for Computing and Computation Science and Engineering at Boston University, NERC benefits from the contributions of systems researchers specializing in research infrastructure. According to BU professor Orran Krieger, MOC Alliance director and co-director of the Red Hat Collaboratory, "The exciting value of the open cloud platform we have at NERC is not just the value it brings by making inexpensive production cloud services available to AI researchers, but the fact that many system researchers can get involved in enhancing and developing new systems."

### About the Author

**Shaun Strohmer** is the editor of *Red Hat Research Quarterly*. She has worked as a writer and editor in academic publishing for over twenty years, and since 2014 she has focused on software development, cybersecurity, and computer science.



**DEMOCRATIZING ACCESS**

NERC is housed at the [Massachusetts Green High-Performance Computing Center](#) (MGHPCC) and operated as an institutionally supported professional service by research IT offices at Harvard University and BU. As part of the open cloud ecosystem, NERC provides a public production cloud aimed at serving scientists and researchers in any domain. Research domains currently using NERC include chemistry, physics, computer science, ecology, social science, neurobiology, genetics, education, and public health.

Unlike enterprise-oriented public clouds such as Amazon Web Service (AWS), Microsoft Azure, and Google Cloud, NERC charges users enough to be self-sustaining and nothing more. Fees cover only equipment, operational staff, rack space, software licensing, and limited administrative costs. Users can expect to pay between one-half to one-third of standard public cloud rates, and rates are expected to drop substantially as the project scales. Both institutions and individuals can sign up for the service.

“The dramatically lower rates of NERC as compared to public clouds, combined with an AI platform like OpenShift AI that allows researchers to focus on their problems, will make NERC the platform of choice for much of the AI research and education happening nationwide,” Krieger said.

**SCALING UP**

Researchers will be able to access extensive resources through NERC, including more than 500 servers and substantial distributed storage.

By the end of 2023, NERC will also add 64 NVIDIA A100 GPUs in 16 liquid-cooled servers to its current stock of A100s, V100s, and K80s. The addition comes thanks to a partnership with Lenovo that sees Lenovo participating in the risk and reward of the open cloud rather than merely servicing NERC as a customer.

“It’s exciting for us to build a new consumption model besides purchasing, which enables access for more users, and it’s exciting to be a part of these universities getting together and building something that’s going to be a win-win for everyone. It’s a win for the MOC Alliance, a win for researchers, and a win for Lenovo because we get to be a part of it,” said Vlad Rozanovich, Senior Vice President, Infrastructure Solutions Group, Lenovo. Rozanovich added, “Solving humanity’s greatest challenges is at the heart of Lenovo’s mission, and contributing to this project gives us an opportunity to walk the talk on that.”

NERC is prepared to scale existing services quickly as new users come on board, for example, as the US National Artificial Intelligence Research Resource ([NAIRR](#)) Task Force begins distributing a proposed \$1 billion in grants for AI research. The open nature of the project makes it possible for partners to rapidly replicate NERC, creating a federated open cloud to meet national demand.

**INNOVATION POTENTIAL**

A diverse community has formed around the potential of the open cloud. Participation comes from multiple sectors, including tech investment firms Two Sigma and

G-Research, cloud solutions providers VMWare and Cloudflare, and hardware makers Intel, AMD, Lenovo, and Schneider Electric, among others. This involvement provides a substantial foundation for galvanizing the rapid development of innovations in cloud services to enable systems support for AI. Projects like the Red Hat Collaboratory, the [Open Cloud Testbed](#), the proposed national [Center for Systems Innovation at Scale Participation](#) (i-Scale), and the open source medical AI platform [ChRIS](#) are designed to enable the system research and open source community by developing and enhancing services for NERC users.

The open nature of NERC also extends its capacity as an engine for innovation. “The platform is open source from the applications down through all software levels to the kernel, and NERC users can see and make changes to the code stack as needed to try new things,” said Red Hat US Research Director Heidi Dempsey. Red Hat and the MOC Alliance have been working together to make all the telemetry and information about the cloud available to the open source and systems research community to enable broad collaboration. All operational and configuration information will be available in open source repositories, allowing organizations to replicate the NERC platform in other locations.

Said Krieger, “The OpenShift AI platform on NERC can rapidly evolve with a set of real and demanding users solving real problems. Given the fast pace of progress in AI and the extensive systems research community around the open cloud,

innovations can happen much more quickly here than on the closed platforms OpenShift AI competes with." As a result, open source platforms will have a significant competitive advantage in a wide variety of computing environments.

### ENHANCING EDUCATION


OpenShift AI is already used at BU as a scalable environment for faculty and students that enables interactive lectures and presentations, student exploration and homework using open source software, and collaborative authoring, publishing, and sharing of course materials. With the expansion of NERC access, faculty using OpenShift AI for education can shift from using AWS to using NERC at a fraction of the cost in the coming year.

Student participation in building NERC provides a different kind of educational benefit. "Open systems and open software give students equal access in building this environment for AI, so they can have the chance to directly influence the systems they are working on and not just consume services," said Dempsey. University students have had the opportunity to do hands-on work in projects that impact real infrastructure, software, and operations tools in the NERC through internships, student-proposed projects, PhD thesis research, and Red Hat Collaboratory grants.

### GET INVOLVED

Researchers and educators interested in using the NERC platform can send a message to [help@nerc.mghpcc.com](mailto:help@nerc.mghpcc.com) to learn more. NERC also

welcomes additional collaborators to this effort, including institutions interested in participating in NERC and the community of researchers, open

source developers, and companies engaged in solving the problems of an open cloud. Please contact [Orran Krieger](#) and [Heidi Dempsey](#). 

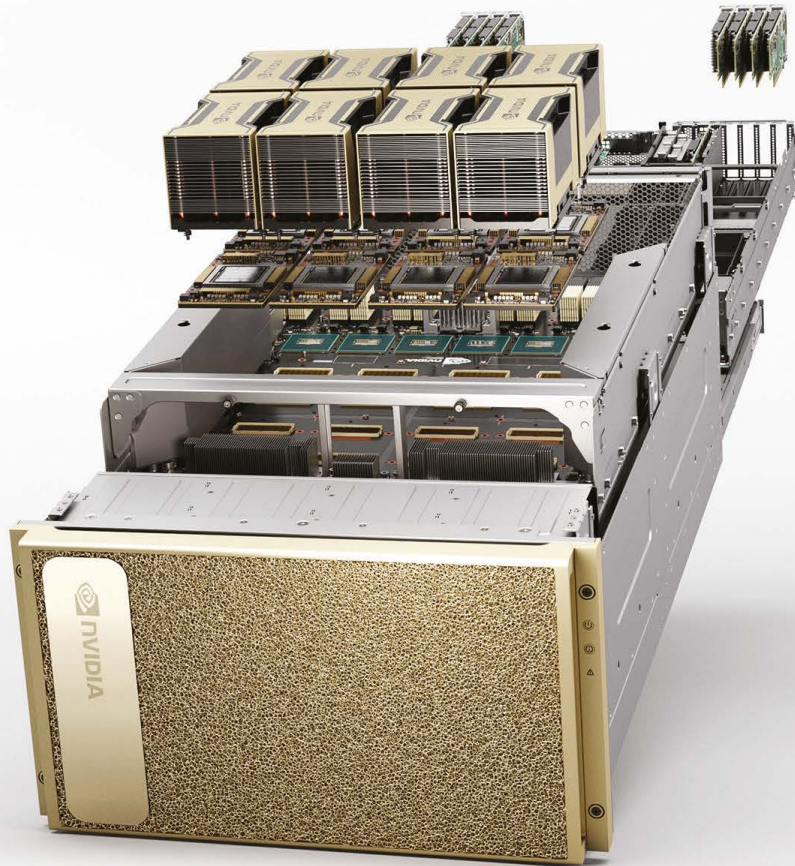
## NEVER MISS AN ISSUE!



SUBSCRIBE NOW

Scan QR code to subscribe to the Red Hat Research Quarterly for free and keep up to date with the latest research in open source

[red.ht/rhrq](https://red.ht/rhrq)



# THE UNIVERSAL AI SYSTEM FOR HIGHER EDUCATION AND RESEARCH

## NVIDIA DGX A100

Higher education and research institutions are the pioneers of innovation, entrusted to train future academics, faculty, and researchers on emerging technologies like AI, data analytics, scientific simulation, and visualization. These technologies require powerful compute infrastructure, enabling the fastest time to scientific exploration and insights. NVIDIA® DGX™ A100 unifies all workloads with top performance, simplifies infrastructure deployment, delivers cost savings, and equips the next generation with a powerful, state-of-the-art GPU infrastructure.

Learn More About **DGX** @ [nvida.ws/dgx-pod](https://nvidia.ws/dgx-pod)

Learn More About **DGX on OpenShift** @ [nvida.ws/dgx-openshift](https://nvida.ws/dgx-openshift)



# Research demo shows hardware–software co–design in action

The Dynamic Infrastructure Services Layer (DISL) puts the benefits of programmable hardware within reach.

Red Hat Research engineers and edge computing specialists Ahmed Sanallah and Jason Schlessman demonstrated the potential of the Dynamic Infrastructure Services Layer (DISL) at the October 2023 Red Hat Research Day. The presentation, [CoDesign in Action: Dynamic Infrastructure Services Layer \(DISL\)](#), introduced an open source abstraction layer that streamlines customization of the hardware stack to meet specific application needs. The project comes from the [CoDes research lab](#) located at the Red Hat Collaboratory at Boston University.

## A DEVELOPER-FRIENDLY SOLUTION

The pressure on developers to optimize applications for speed, performance, and efficiency keeps growing. Programmable hardware such as Field Programmable Gate Arrays (FPGAs) offers those improvements, but proprietary tooling and inefficient development workflows have made them inaccessible to developers without specialized expertise.


DISL, a collaborative effort between engineers and researchers at Red Hat and BU, solves that problem with an interface that allows the entire hardware stack to be fully expressed and customized using only configuration

files. It also provides a library of tools for managing deployments in both wired and wireless configurations. Developers can now take advantage of the benefits of specialized hardware in a way that is portable and can be dynamically updated wirelessly as needs change.

Ben Cushing, Chief Architect, Health & Life Sciences, Red Hat, led the Research Day conversation with Ahmed and Jason as they walked the audience through the DISL project. The team demoed building a custom wireless security system using off-the-shelf components to illustrate how DISL supports quick builds of far-edge systems.

Those who missed the live Research Day event can find the recording and slides on [the event page](#), as well as a 38-minute demo version without discussion. In both recordings, the team discusses the value of far-edge research and FPGAs, the difficulties of using FPGAs, and the details of the DISL design before launching the demo.

## TO GET INVOLVED

Researchers are actively seeking ideas for potential use cases. Contact [Ahmed Sanallah](#) and [Jason Schlessman](#) to share your thoughts. 



# No more *gatekeepers*

Why technological ignorance is radically dangerous and how an open world will help

---

An interview with **Jonathan Appavoo**  
conducted by **Jason Schlessman**



## Interview

**W**hat is the role of the technologist when building the future? According to Boston University professor Jonathan Appavoo, “We must enable flight, not create bonds!” Professor Appavoo began his career as a Research Staff Member at IBM before returning to academia and winning the National Science Foundation’s CAREER award, the most prestigious NSF award for junior faculty who successfully integrate education and research. Now on sabbatical with Red Hat Research for the academic year 2023-24, he has been engaged in multiple projects that enable flight in distinct ways, from developing a Linux-based unikernel to building an open education platform giving faculty and students access to open source technologies through a web browser.

We asked Principal Software Engineer Jason Schlessman to interview Professor Appavoo, thinking that their mutual interests in AI and machine learning, the impact of technology on people, and expanding access to technological resources would spark a provocative and fascinating conversation—and wow, were we right!

**Jason Schlessman:** Do you recall when you first got interested in computation?

**Jonathan Appavoo:** It depends how you define computation. If you define it as staring at trunks full of electronics in our family’s basement, it was as early as I could get down into the basement. In terms of getting fascinated by ideas, it was as soon as it became a possible alternative to doing my schoolwork.

**Jason Schlessman:** I was thinking of computation as opposed to just being into computers. Because there was a point early on where you could have a Commodore 64 and write BASIC with it. That’s the difference between playing Oregon Trail and thinking about computation.

**Jonathan Appavoo:** For me, the two blurred very quickly—partially because it stemmed first from staring at trunks of electronics. What was fascinating to me was

not so much the world of writing code but the fact that you could make something work. There was something visceral about seeing things that were animating by thought.

That was always how it took on meaning for me: I liked to write code, but that was a side effect of realizing that the computer was very different from the other machines in my life. It was clear to me that a hammer was a hammer, but when it came to the machine, I was dealing with something human-like.

That’s what got me interested in the idea of computation, even before I understood the notion of how we define computation. It’s the same thing that drives the research now. The far end of my research today is still at that border of what it means to embody human thought in a machine.

**Jason Schlessman:** So the electronics in the basement—they were inanimate. As you got



### About the Author Jason Schlessman

is a Principal Software Engineer at Red Hat Research, focused on novel AI and machine learning innovations that lead to pragmatic and feasible solutions. He especially targets projects that serve the well-being of humanity, fostering ethical uses of technology.



*Jonathan Appavoo presents at the MOC Alliance Workshop 2023 at Boston University.*

older, you saw a humanistic side to those heretofore inanimate objects, which sparked your curiosity.

**Jonathan Appavoo:** Yes, and it was especially visceral as I was watching my father, an electronics engineer, wielding those devices to make games. He made one where he welded together a steel gun-like thing, put a light in it, and then had my brother paint a picture of a tiger and put a light sensor in the tiger's mouth. If you could make the light hit the sensor in the mouth, you won the game. We did these sorts of things prior to building our first motherboards. I saw that this wasn't just about digital electronics or analog electronics. It was about embodying ideas into things that were animated by electricity and ideas.

**Jason Schlessman:** Do you find the element of play has guided your interests since then?

**Jonathan Appavoo:** Absolutely! Most of my students look at me sideways when they first meet me and I say, "Look, all I care about is that you have fun." If we're not having fun, then something is fundamentally flawed. The only way to get to the heart of something in research is to really fall in love with it, and I know no better way than having fun. If your only motivation is just "I've got to get it done," it's going to be pretty hard to do meaningful science. When I started my education, I was interested in how to impart meaning to what you see on the screen. I double majored in psychology because I wanted to understand some of the neuroscience. How do these things

relate? A thread in all my work is trying to understand the relationship between statistical mechanisms, inference, and deterministic computation.

**Jason Schlessman:** I noticed in your website bio that at one point you were working on vision but moved to operating systems. As a vision guy myself, I'm curious about that change.

**Jonathan Appavoo:** It's probably more of a failing of my IQ than the field. I was really interested in conjectures by various robotics researchers that the biological and physical constraints of an organism contribute to the way its intelligence arises, and I became convinced that that was the direction I wanted to go. While at the University of Toronto, I worked on a project building a vision-based wheelchair for disabled children—in particular on a binocular head for a robotic chair. I did some studies on depth perception with it, but I quickly realized that my lack of background in signal processing and fundamental electrical engineering meant I wasn't ready to be there.

On the other hand, building the underlying system and starting to grapple with those problems was much closer to my heart. When I was having that soul-searching moment, I realized I was at a unique place where we were still building some large-scale shared memory multiprocessors of our own, from the bottom up. We were building the boards, writing the tool chains, operating systems, and applications. I thought, if someday I'm going to get back into this idea of pursuing intelligent, biologically inspired machines, surely having a grounded focus on being able to build



stuff and understand computational systems and parallelism would always make me a marketable quantity.

It turns out that I'm not really suited for the real world where you produce products, but that's okay. That was an incredibly useful bridge, because when I did switch labs and started working on shared memory multiprocessors, the lab was focused on a theme that would stick with me literally for the rest of my career: can you build systems that scale? And if you squint, that's deeply related to the questions of what scalable computation is and what biology has to say about scalable computation.

**Jason Schlessman:** We haven't touched on how open source influenced your career in your formative years, but I know it was frustrating for me, early on, that everything for vision was very closed source. At what point did open source become important to you?

**Jonathan Appavoo:** It took me a little while to understand the deep importance of the relationship between computing and open everything. When I did that work on depth perception, I used a machine that was proprietary, and all the code I wrote was all very proprietary. The algorithm I was trying to implement was from an open academic publication about how to use orthogonally oriented Gabor filters, but the code I was writing was for these closed GPU-like vector processors that were specific to a task.

That was already bugging me, because I hate the move toward

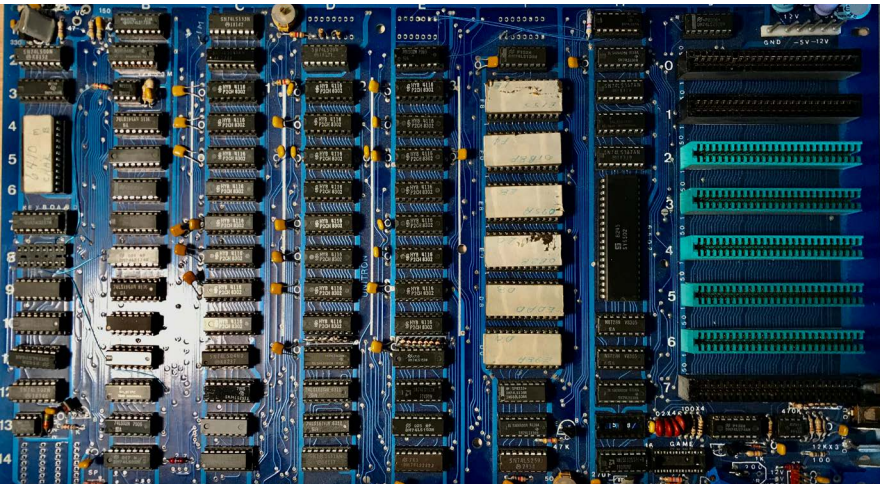
GPUs and all these custom things. It really starts to break down some of the early promises and benefits of openness in computing. In the 70s, we got the microprocessor, making it affordable for us to get machines and to literally hack them. An old MOSS manual, or even an old Intel manual, will tell you how to build a machine in just a few pages. There are no encryption keys. There's nothing telling me what I can and cannot do with the machine. It was a general-purpose device that we could all have our take on and then apply our creativity. That was fundamentally liberating to many kids like me. As a first-generation immigrant living in rural Ontario, I was not the prime candidate to have access to things.

It took me quite a while in graduate school to realize the danger of the world that I was participating in. I was, and still am in some sense, part of those who build large-scale computations. We've seen what large-scale computation can do to the world. Not only does it start to create concentrated digital power, it sucks up large fractions of human resources. Then we gate it by limiting who has access to those machines. We gate it by putting keys in the processors ensuring that only some people know how to boot the machine. You can't actually boot some of our hardware without a co-processor with encrypted code in it. That seems antithetical to all the things I benefited from.

Is this the right way to spend our resources? It's not for me to decide. I'm a scientist whose job should be to make sure that society has equal participation. I did my last

Technology doesn't  
just arise and then  
affect society—it's  
a product of it.  
Scientists are a part  
of society.


### My First Computer Experience



### My Students' First Computer Experience



*For many students, a computer is an opaque piece of glass that somebody else controls.*

 See “Open source education: from philosophy to reality” (RHRQ 5:1) to learn more about OPE.

sabbatical working with people in ethnography and looking at questions of agency. I think it's folly for scientists to be arrogantly talking about “the impact of my technology on society.” Technology doesn't just arise and then affect society—it's a product of it. Scientists are a part of society. The decisions I make, the problems I choose to work on, the things I tell my graduate students—all of those things create technology in the context

of my bias and my goals. To make it seem like it popped out of nowhere and now we must deal with its impact on society is not true.

When my close colleague and dear friend (BU professor) Orran Krieger and I were working on the Tornado operating system at the University of Toronto, the beauty was we had to build everything—every line of source code. Linux was only starting to show its glimmers of existence, so instead, we were comparing ourselves to proprietary UNIX-based systems and closed source hardware. The systems we were building were academic, and open.

Then as Linux started to arise and became popular, there was real transparency. You could see every line of code that went into making the system work. So while it took me a while to fully understand the importance of an open world, it became a running theme for all my work, from what I did at IBM to my projects in education now. The effort to ensure that all these things taking up so many resources are equally accessible to all of humanity is really critical.

**Jason Schlessman:** As you were talking just now I was thinking about little Jonathan in rural Ontario. If that microcontroller weren't open and available, who knows what would have happened differently? Similarly, the opportunities you're making with the Open Education Project (OPE), removing economic or other boundaries, means that what was fascinating to us can fascinate others.

**Jonathan Appavoo:** First, I want to reiterate one of the most important things in what you said: it's about enabling and unleashing others' potential. For those of us in systems, the most pleasurable thing is when we find just the right primitive that enables other people to do things that we didn't imagine.

One of the most important aspects of the open education initiative was the idea that



understanding how everything works is radically important. If people believe the world of computation and the digital world is just magic and inherently belongs in the hands of others, that's radically dangerous to me. We should be investing in ensuring that people know that this digital world is grounded in reality and that they can comprehend it. It's the confidence of knowing that you can understand—and the euphoria of comprehending what you thought was incomprehensible. And then you can use it to express your own creativity and build something that you want to build.

When I realized that the [Jupyter](#) landscape was a widely deployed web interface to a UNIX runtime, with an ecosystem that was completely open, and all you needed to access it was a web browser, the first thing I considered was that we could help people get access in a public library—that is, if I convince people it would be useful to donate computation to run the back ends on open platforms. It was my goal to ensure that the assignments and the material and the examples are all live, and that you can get access to them from a web browser, if somebody's willing to host some back-end computation for you.

**Jason Schlessman:** What have you seen as far as deployment, like people using it and interacting with it?

**Jonathan Appavoo:** It's extended in a direction I was only hopeful for. On a practical front, I have a whole bunch of kludged stuff that I built for my Introduction to Computer Systems class. It was motivated by one of my colleagues, Professor

Mark Crovella, who had built an online Jupyter Book textbook for a linear algebra class. (Jupyter Book is a part of the Jupyter ecosystem for interactive books that uses the Jupyter environment to both author and present content.)

Poking around, I realized that opening a terminal and ensuring that there was textbook material as well as terminal-based assignment and lab material could work. That's when I started writing *Under the Covers: The Secret Life of Software*, with the idea that it would marry these things together. I kludged up a bunch of things with my very broken Python to get terminal-based reproducible assignments and examples working.

But we also quickly realized that there was a deeper opportunity here to encourage and seed an open education platform: a platform tying together things like the [MOC Alliance](#), datacenters that are publicly funded, and cloud infrastructure from open public software. We could take something like Red Hat OpenShift and say that the underlying mechanisms for everything that's going to build up a computational environment can never be taken away. There are no bits that are proprietary. And the glue that holds all of this together, Linux, is an open source operating system.

What I didn't quite grok at first was Git and the power of using everything that we as technical people have developed for open collaboration and tracking. We aren't doing things motivated by fear. You want to fork our stuff—fork it! You don't want to acknowledge me—don't!

For those of us in systems, the most pleasurable thing is when we find just the right primitive that enables other people to do things that we didn't imagine.

This idea of distributed repository control is really a model for empowering and operationalizing communities.

One of the things I struggled with a lot was how to remove the dependency on PowerPoint, Google Slides, Keynote, and certain image formats. Professor Crovella had already demonstrated the power of a Jupyter extension called [RISE](#), which lets you create and run presentations. So we did some tooling around RISE to make it easier to build presentations.

Proprietary data formats were not in my bailiwick, but it really bothered me that images, which are so powerful in education, were all being fundamentally gatekept by the formats. I want to know that you can get access to the content and easily track and edit it. That's when we bumped into the [Draw.io](#) ecosystem, so that at least the underlying representation of an image as an XML text document could be saved, repository-controlled, and edited through clients for which the Java source was open and available. I try to make sure that the book infrastructure we've tried to seed encourages people to use only formats that can be kept under repository control, rather than in a binary format that would lock us out for the future.

**Jason Schlessman:** Git also creates the opportunity for an educational component. If I can see how you already did something, I can follow your model. Do you see that happening?

**Jonathan Appavoo:** Yes, absolutely.

The way I explain it to students is that it's the "Making Of." You watch a movie, then you watch the director's cut and get to see the Making Of.

Being able to see stories of how things got done provides massive amounts of educational value. And it also shows us collaborating. This idea of distributed repository control is really a model for empowering and operationalizing communities. I think we as technical people can be very proud of that: we are helping contribute not just platitudes about how the world should cooperate but structures and models for how to do it.

**Jason Schlessman:** Absolutely. You mentioned the MOC Alliance as far as the provision of cloud resources. This work also involved the Red Hat Collaboratory, is that correct?

**Jonathan Appavoo:** Yes. I'll be really honest here: when Orran first told me we were going to have this really cool collaboration with Red Hat, I was like, "Look man, I left industry, I'm not interested." I've seen how this goes down: corporations come to you and you work on what they care about. I'm so glad to be proven wrong. It's been radically powerful, and I think that it has something to do with the fundamental open source DNA of Red Hat. It really changed how people work and collaborate with us. Instead of saying, "Oh, here's some money, go away, and maybe these are the things we'd like you to focus on," we are engaging in a real conversation, and I've been thrilled.

I cannot imagine going someplace and saying, "I know that our collaboration was supposed to be about certain research things, and that's what I'm known as for a scientist, but I think the educational mission of getting people to understand how systems work at a fundamental level and making that

open matters.” But I did, and they didn’t just say, “Yeah, that’s great. Have fun.” I was given support. I was given resources. The investment of people’s time and energy to just listen and care has been phenomenal. And the OpenShift AI environment (formerly Red Hat OpenShift Data Science)—yes, it’s called open data science, but it turns out that we can actually use it as an education platform for teaching systems. Nobody balked at that either.

As a matter of fact, Red Hat is really benefiting from the risks Red Hat Research has taken. I’m thrilled to see Red Hat Research taking a platform such as OpenShift AI that’s meaningful to the corporate side, helping me operationalize it for education, and then in the end, give utility back to the corporation.

**Jason Schlessman:** My experience has been that OPE is just next level in terms of what’s being done to lower barriers and inspire creativity and passion. Can we bring the passion and fascination that you and I feel to the masses?

**Jonathan Appavoo:** Yes, and I think the obligation to do so is higher than ever. Large language models will really magnify the value of being intrigued and passionate. I’m just thinking about how that happened for something like OPE. It really took vision on the part of people like (Director of Research) Hugh Brock, (US Research Director) Heidi Dempsey, and Red Hat engineers like Danni Shi and Isaiah Stapleton. These folks have been willing to take a bet on something that you couldn’t argue has value as a product. OPE came

about because corporate researchers embedded with academics who have a teaching mission were willing to invest a little time to see where things could go. That’s the kind of thing that’s not going to happen from an LLM. You can’t ask a large language model, “Should I go do this?” It’s not a sure bet.

---

I think we have a deep obligation to listen to people who are well outside our normative set: people in fields from philosophy to political science to ethnography to sociology.

---

**Jason Schlessman:** What about openness with LLMs? Hugely important or not?

**Jonathan Appavoo:** There are several fronts to that. A lot of very smart, capable people are thinking very hard about LLMs and their impact on society, so I don’t want to go into areas that aren’t my expertise. I think we have a deep obligation to listen to people who are well outside our normative set: people in fields from philosophy to political science to ethnography to sociology. It’s important that the entire human tent is engaged in these conversations because of the scale at which we can operationalize these technologies, and

because of the speed at which we can release these viruses onto the world.

I can give you my opinions, but they are just my opinions. My concern is that there is a limited number of folks who have control in the conversation. Let’s engage as many people as possible. And to me, the only way to accomplish that is by defining openness very broadly. It’s open access, but also open understanding. That’s why simple goals like getting people to understand how computers work matter. I don’t need to be a rocket scientist to engage in a conversation about these technologies.

**Jason Schlessman:** Going back to the OPE project again: since you are continuing the OPE project during your sabbatical, where do you envisage that project going next, and what challenges do you foresee?

**Jonathan Appavoo:** There are some practical things that really require industrial help and commitment. One of the things we’re trying to do is codify best practices. We’re not building that much new stuff. Rather, we’re trying to find the right open source tools, the right way to package things, and the right way to give people guidance on how to build things. I want to provide exemplars that allow people to accelerate their ability to produce high-quality open educational materials. We should be able to put some investment into ensuring that if you take these things and you put them together, the container will be stable to the point where you can run a 300-person



One of the great things about the open source world is you can spit and you're going to hit something interesting.

class and know it'll work for the whole semester and people will actually get the educational value of it.

There's also the hard work of ensuring that versions of things are compatible, they work, and they are consistent. One of the great things about the open source world is you can spit and you're going to hit something interesting. You can do a Google search for anything you want to do and find thousands of open source projects. For OPE, we have to curate and find examples and exemplars where people have invested in ensuring that they are operationally viable. When open education goes beyond the world of computer scientists or engineers, it is fundamentally critical that we make things usable for others.

At first, the repositories had a structure in which you cloned the OPE repository to build a book. But that wasn't quite right, because that inherently ties up histories and repositories. Instead, we constructed a tool that lets you create and seed repositories with template content and authoring tools. So OPE is more like a toolchain for building your own projects, and your own personal coursebook or materials. The projects themselves are their own repositories and can take on their own life, structured in a way so you can see examples and improve your productivity.


**Jason Schlessman:** Is the sabbatical an opportune time to look at those best practices that a place like Red Hat is putting into effect every day?

**Jonathan Appavoo:** One of the huge values Red Hat provides is taking an open source world, curating it, and ensuring that, if you take these bits

in this way, they will work. The reason I use Fedora and RHEL is because somebody's done a lot of work. My sabbatical gives me the opportunity to work with these really talented engineers who are willing to passionately spend time on OPE and think it makes sense. We're figuring out what the right architecture looks like, how to automate things, and how to move it into the NERC. It's a fascinating problem: how do we ensure that when we render a book it looks and interacts correctly?

**Jason Schlessman:** I want to close by asking about this quote on your BU home page: "Striving for stable insanity and fascination with each moment." Would you say you're still doing that?

**Jonathan Appavoo:** The people to ask are probably the people who work with me! But yes, absolutely. That's why I came back to academia, because one of the things the academic environment still affords is the opportunity to operationalize diversity. Our future depends on our heterogeneity and the opportunity to do crazy things and have ridiculous conversations.

Our field is full of revisionist history. We want to say that the founding members of our field were straight arrows. Yes, they were brilliant logicians, brilliant mathematicians, but many of them did strange things, and we sanitized it afterward. We've reduced the ability to look like an idiot. And I am Idiot Number One. The ability to say stupid things in a safe environment with people you trust, in order to go where no one has gone before, that to me is sacrosanct. That's why being willing to be a little insane is okay. And actually necessary. 

# MOC ALLIANCE WORKSHOP

## February 28–29, 2024

**Boston University George Sherman Union (GSU),  
775 Commonwealth Ave, Boston, MA 02215**

The 2024 MOC Alliance Workshop invites researchers and industry leaders in systems and AI engineering, the Open Source community, Research IT partners (operations and facilitations) from the affiliated institutions, and users of MOC Alliance affiliated projects to celebrate our shared journey and achievements. This event offers a platform to connect, exchange ideas, and inspire each other. It's an opportunity to reflect on where we've been and to plan our objectives for the next year, bringing together our rich community in a single event.



**More info and registration:**  
[massopen.cloud/2024-workshop/](https://massopen.cloud/2024-workshop/)

*Housed at:*

**Boston University** Rafik B. Hariri Institute for  
Computing and Computational Science & Engineering



Clouds that  
compete can't  
connect.



Says who?

/Keep your options open  
[redhat.com/options](https://redhat.com/options)



Copyright © 2023 Red Hat, Inc. Red Hat and the Red Hat logo are trademarks or registered trademarks of Red Hat, Inc., in the U.S. and other countries.



- A AWS
- B Azure
- C Google Cloud
- D All of the above

## Feature

**About the Author****Richard Jones**

has been using Linux since the early 1990s, joining Red Hat in 2007. Richard is now a Senior Principal Software Engineer in Red Hat's R&D Platform team.

## RISC-V extensions: what's available and how to find them

Extensions available in RISC-V enable the customizations that make it ideal as a basis for open innovation. Here's the extension situation as it stands today.

by *Richard Jones*

**R**ISC-V is a new Instruction Set Architecture (ISA) that, over the next decade, will compete with x86-64 and ARM in all areas, from the lowest-end IoT devices all the way to huge servers. Unlike existing ISAs, RISC-V is a truly open source, permissionless architecture. You can start making chips using designs downloaded from GitHub without signing any legal agreement. If this sounds like Linux versus proprietary operating systems all over again, then you understand why Red Hat is interested.

One unique aspect of RISC-V is that it is designed from the ground up for ISA extensions. As the name suggests, these add extra instructions to the ISA for implementing features like vector operations or accelerated encryption.

In this article, we'll look at how extensions work at the lowest levels, how they are ratified and eventually standardized through RISC-V International (RVI), what extensions

are out there, how extensions are grouped into profiles, and how you can discover which extensions are supported in your hardware. I will also cover QEMU support for extensions.

As this article is about extending the ISA, I will not cover non-ISA extensions in any detail.

### **WHY EXTENSIONS?**

Broadly, you can extend and accelerate the capabilities of a CPU in two ways: add new instructions or implement a hardware accelerator. On other ISAs, accelerator peripherals exist for TCP offloading, AI, digital signal processing, encryption, and so on. Why would an extension be better—or, when is an extension better?

One way to think about this question is that extensions are part of the stream of instructions. They are, therefore, extremely low latency: there may be little to no overhead to using the instruction. This is in contrast to making an I/O request, where

you might have to form a request packet and batch requests to get the best performance, and, of course, the request goes off the CPU and over a PCIe network.

That said, extending the CPU is far less easy than plugging in a peripheral.

### HOW EXTENSIONS ARE ENCODED

RISC-V has a very regular instruction encoding. Because arbitrary extensions are allowed, RISC-V uses a variable-length encoding, with all instructions (currently) encoded either in 32 bits or in 16 bits for a compressed subset. For this article, I won't talk much about compressed instructions— see the section below for more on that. And I won't discuss variable-length encodings that are 48 bits and wider, since they are not used by any extensions today.

With those assumptions in mind, we can assume the least significant 2 bits of each 32-bit instruction are always **1 1**. The next 5 bits are the major opcode that control how the instruction is decoded (see **Figure 1**).

This is not a comprehensive guide to decoding RISC-V instructions, as that is well covered in the [RISC-V ISA spec](#), but some things are worth pointing out:

- Some instructions have further opcode fields. For example, **OP** (**0110011**) has two extra opcode fields totaling another 10 bits. This major opcode is shared with the base ISA, the Multiply extension, the Zicond extension, the Packed SIMD extension,

and more. There is no general way to tell if a particular opcode corresponds to the base ISA or an extension (without a large table).

- Some major opcodes do correspond entirely to specific extensions. For example, **AMO** contains Atomic instructions (the **A** extension), and **OP-V** was originally reserved but is now used by the Vector extension.
- Large sections of the opcode space are used for some fairly obscure features, like fused multiply-add.
- **custom-0** through **custom-3** are for vendors to add their own extensions that they don't intend to ratify. Essentially, **custom-\*** is a free-for-all (except that the proposed 128-bit RV128I ISA will eventually use **custom-2** and **custom-3**).

In this article, I will concentrate only on extensions that are non-custom, non-vendor-specific, and on the path to ratification by RVI.

It is most important to note that (non-custom) extensions are not distinct, separate parts of the decoding space. RVI appears to want to thread extensions into the gaps between existing base instructions. Major extensions like Vector that have their own major opcode (**OP-V**) also have instructions in **LOAD-FP** and elsewhere. I have heard extensions that have their own major opcode called green-field extensions. Smaller extensions use existing major opcodes, fit into the gaps, and are known as brown-

Major opcode			
000	00	11	LOAD
000	01	11	LOAD-FP
000	10	11	custom-0
000	11	11	MISC-MEM
001	00	11	OP-IMM
001	01	11	AUIPC
001	10	11	OP-IMM-32
001	11	11	(48-bit instructions)
010	00	11	STORE
010	01	11	STORE-FP
010	10	11	custom-1
010	11	11	AMO (Atomic extension)
011	00	11	OP
011	01	11	LUI
011	10	11	OP-32
011	11	11	(64-bit instructions)
100	00	11	MADD (fused multiply add)
100	01	11	MSUB ( " " )
100	10	11	NMSUB ( " " )
100	11	11	NMADD ( " " )
101	00	11	OP-FP
101	01	11	OP-V (Vector extension)
101	10	11	custom-2 / RV128I
101	11	11	(48 bit instructions)
110	00	11	BRANCH
110	01	11	JALR
110	10	11	reserved
110	11	11	JAL
111	00	11	SYSTEM
111	01	11	OP-P (Packed SIMD extension)
111	10	11	custom-3 / RV128I
111	11	11	(80 bit and longer instructions)

LSB 1 1 means non-compressed

**Figure 1.** Major opcode

field extensions. All else being equal, brown-field extensions have a higher chance of being accepted. This means that when proposing a new extension, you will need to consider existing extensions, which is good practice anyway.

### Byte ordering of instructions

Instructions are always stored little endian, with the least significant byte first in memory. This applies even on





**Figure 2.** How `auipc` appears in documentation and `objdump` output

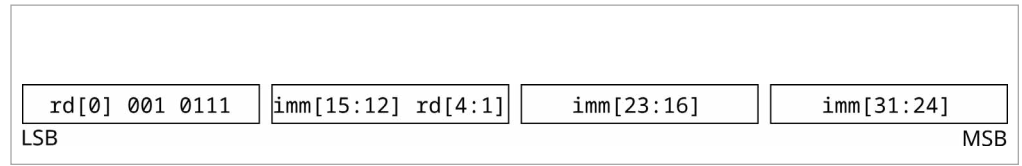
the (theoretical) big endian RISC-V machine. Note that disassembly tools like `objdump` display the bytes as big endian. Thus an instruction like `auipc` appears in documentation and `objdump` output as shown in **Figure 2**, but in memory as shown in **Figure 3**. In other words, when decoding RISC-V instructions, you can tell much about how to decode the instruction and where the next instruction begins from the first byte. This is in stark contrast to x86, where simply determining the boundaries between instructions is a research project in itself!

### The classic extensions

RISC-V was originally envisaged as the base ISA—RV32I, RV64I, or RV128I—surrounded by what I will call the classic extensions:

- **I**: Base integer instructions, add, subtract, jump, etc.
- **M**: Multiply and divide
- **A**: Atomic operations
- **F**: Single-precision floating-point arithmetic
- **D**: Double-precision floating-point arithmetic
- **C**: Compressed instructions

Notice how some of these map fairly well to the major



**Figure 3.** How `auipc` appears in memory

opcodes: for example, Atomic operations are implemented in the **AMO** opcode space.

We might add:

- **G**: an early attempt to define a basic profile, **G = IMAFD**
- **E**: an embedded subset with 16 base registers instead of the usual 32
- **Q**: quad-precision floating point
- **H**, **U**, and **S**: pseudo-extensions used in the `misa` CSR (Machine ISA Control and Status Register, more below) to refer to support for hypervisor, user, and supervisor modes
- the **X**-prefix for custom extensions
- the **Z**-prefix

It soon became obvious we were going to run out of single letters, so three prefixes were reserved for named extensions: **S** (e.g., **Smmtt**) for supervisor-mode extensions, **Z** (e.g., **Zimop**) for general extensions, and **X** for custom, vendor-specific extensions.

**Zicsr** and **Zifencei** were retrospectively detached from the base ISA after we realized that CSRs might not be present on very low-end hardware, and the **FENCE.I** instruction didn't work very well.

One extension may also require another; for example, **D** requires **F**.

### Naming extensions

There is a well-defined naming scheme describing which extensions are supported by hardware. I'll describe below how you can find this out for your own hardware. At the time of writing in 2023, the vast majority of hardware can be described as:

### RV64IMAFDCZicsr\_Zifencei

Extension versions can also be encoded here (e.g., RV64I1p0 would be base ISA version 1.0).

### The curious case of compressed encoding

If you were paying close attention to how RISC-V extensions are encoded, you will see that I assumed 32-bit, non-compressed instructions.

Current RISC-V implementations support a compressed 16-bit encoding for common instructions that is, of course, limited in the range of registers that may be accessed and the opcodes available. This is similar in spirit to armv7 Thumb instructions. The compressed extension was added very early on by the original RISC-V designers to help save the instruction cache

and fetch bandwidth. Later, Linux distributions started to assume that the compressed extension is always available.

The downside is that compressed instructions consume three-fourths of the available opcode space (non-compressed, 32-bit instructions must have both least significant bits **1 1**). Also, instructions are no longer automatically 4-byte aligned and may also cross cache lines and page boundaries, making decoding harder, albeit still much easier than complex architectures like x86.

High-end RISC-V server vendors are pushing back against supporting compressed instructions, arguing that their machines will have huge instruction caches (so code size is not critical). These vendors would prefer to use the opcode space in other ways and would like to have uniformly sized instructions. We have yet to see how this will pan out, but don't be surprised if servers appear that don't implement the **C** extension.

This is very much a concern for Red Hat. **C** is a special extension as these instructions appear frequently in binaries—as many as half of all instructions can be compressed—and trap-and-emulate would be impractical. Shipping two distro variants with and without compressed instructions is not attractive. Instead, we must decide whether to require it in hardware or ban it in software.

## THE NEW EXTENSIONS

The classic extensions, in hindsight, don't cover much of what is needed for a modern server. Since then,

dozens of extensions have been proposed and ratified. This section highlights the most important extensions first (for servers). You can get a complete list of extensions and their status on the [RISC-V wiki](#).

### Vector, encryption, math

The most important extensions to the classic set are Vector (**V**, **Zv\***), Bit manipulation (**Zb\***), and Packed SIMD (**P**, **Zbpmo**, **Zp\***). These very roughly correspond to MMX/SSE/AVX on x86, but RISC-V adds more flexibility and a different—and simpler—programming paradigm. We expect that when hardware with these instructions appears, they will be widely used in binaries (as happens on x86).

A whole article could be written about the vector extension (which is, in fact, a large collection of extensions). Here are two:

- [Adventures with RISC-V Vectors and LLVM](#)
- [Programming with RISC-V Vector Instructions](#)

RISC-V also has two important sets of extensions for cryptography, called Scalar Crypto and Vector Crypto. Scalar Crypto was folded into the Bit manipulation extensions (**Zbkb**)—for example, the **zlp** instruction in **Zbkb** is called out for being useful to implement SHA3. Vector Crypto (**Zvknhb**, **Zvbc**, **Zvkn**, **Zvks**, and more!) contains extended Vector instructions useful for Elliptic curve cryptography, various Message Authentication Codes, AES, AES-GCM, and many more.

Another important group of extensions extends floating-point support, adding Bfloat16 (**Zfbfmin**, **Zvfbfmin**, **Zvfbfwma**), common floating-point constants and many useful floating-point operations that are not present in the classic set (**Zfa**), and the “f-in-x” extensions (**Zfinx**, **Zdinx**, **Zhinx**, **Zhinxmin**) that allow floating-point and integer registers to be shared.

### Virtualization

RISC-V support for running virtual machines (the Hypervisor extension) was demonstrated as far back as 2017 and ratified in 2021, but it is only expected to appear in hardware in 2024. This will be vital for RISC-V adoption on servers. **H** (hypervisor) is mostly a complicated addition to the [privileged spec](#) involving new modes and CSRs, but some new instructions were added. In particular, there are instructions to access memory while translating guest virtual addresses (useful for emulating I/O) and extra fencing instructions.

### Interrupts, cache, and memory

Many extensions have been proposed and ratified that are beneficial for server-class operating systems. The most important are probably the ones that fix the interrupt architecture of the original design, which was notably inefficient—in particular, the Advanced Interrupt Architecture (**Smaia**, **Ssaia**) and the older Fast Interrupt specification (**S\*clic\***). (Recall that extension names prefixed with **S** apply to supervisor mode). Also worth mentioning is **Smrnmi**, which fixes another issue with the base standard: after a Non-Maskable Interrupt,

the interrupted program could not resume running. This adds a new `mnret` instruction to resume after NMI.

The original RISC-V design assumed a relaxed memory consistency similar to Arm, but some machines would prefer the stricter ordering found on x86 (for example, because you need to emulate or port code from x86). The `Ztso` extension changes load and store operations to use total store ordering.

Cache management operations (CMOs) are important in modern operating systems, and RISC-V defines a family of extensions for cache block operations. `Zicbom` are cache block management instructions for things like invalidating blocks of cache. `Zicbop` instructions are prefetch hints. `Zicboz` instructions store zeros over blocks of cache.

### Safety and security

Hardening against attacks is a key concern for servers. One area being actively developed on all architectures is control flow integrity (CFI). RISC-V is ratifying two extensions for CFI. `Zicfiss` defines a shadow stack and provides new instructions to push and pop values there. `Zicfilp` defines places where code is allowed to branch to (especially through “computed gotos”), known as landing pads. These techniques are designed to prevent return-oriented programming (ROP) attacks after stack-smashing exploits. Landing pads themselves are defined by further extensions—`Zimop`, `Zcmop`—

that reserve some opcode space for [may-be-operations](#) (MOPs).

### Miscellaneous

Other extensions relevant to servers:

- `Svinval` can be used for selective TLB invalidation.
- `Zawrs` adds instructions that make polling memory locations more efficient, typically used in spinlocks or when polling on a lockless queue. `Zihintpause` adds a new pause instruction that can also be used to reduce power consumption and memory traffic in spinlocks.
- `Zihintntl` may be used to hint that memory accesses are non-temporal (i.e., do not need to be cached).
- `Zacas` adds atomic compare and swap, omitted from the original Atomic instructions.
- `Zicond` adds conditional instruction prefixes, similar to armv7 conditional operations.

### EXTENSION STANDARDIZATION AND RATIFICATION

RVI has a process for taking non-custom extensions, shepherding them through standardization, and eventually ratifying them. Unlike proprietary ISAs, this process (and the arguing!) happens in the open, on GitHub pages, in mailing lists, and on open video calls. Extensions on their way through ratification are listed on the [RISC-V wiki](#), along with links to the process and a description of the lifecycle extensions go through before ratification.

### PROFILES

Software generally needs some kind of baseline target to run.

While some extensions can be detected at runtime and different code paths chosen, much software will be written that expects a basic set of extensions to exist.

For this reason, RVI defines a set of profiles named after the year they were defined and grouped into two families. Thus, at the time of writing, the latest profile is RVA22, and RVA23 is in development. A stands for the “Application processors running rich operating systems” family (i.e., servers); 22/23 are the year codes.

RVA22 includes all the classic extensions and a smattering of older system extensions. More notable is what it omits. It predates the ratification of the Vector extension, so this is only optional, and Vector Crypto and Packed SIMD are also missing. See the full [RVA22 profile](#) for more details.

We expect this to change, as the current profiles don’t reflect the many branches of the RISC-V ecosystem. In the future, we expect to see stricter requirements for backward compatibility, only fully ratified extensions allowed, and unused opcode space forced to trap (allowing some forward compatibility through trap-and-emulate).

### DISCOVERING EXTENSIONS—WHERE’S MY CPUID?

x86 has CPUID, a comprehensive method to detect at runtime which features the processor supports and many other aspects of the CPU, like cache sizes and so forth. There is nothing this comprehensive available in RISC-V at the moment.



For RISC-V, there are three ways to determine which extensions are available in the hardware. The oldest mechanism, now mostly deprecated, is to read the `misa` CSR. This register lets you read the machine `XLEN` (i.e., the base ISA: RV32I, RV64I, or RV128I), but your code won't run unless it uses the right instructions in the first place, so you must know this already. It also contains 26 bits corresponding to the 26 letters of the alphabet, anticipating up to 26 extensions (minus reserved letters). As discussed before, believing there would be only 26 extensions was naive. Another problem with `misa` is that you cannot read versions of extensions, but the two other methods allow you to get all extensions and (in theory) their versions.

The second method is to use information from device tree (DT). The deprecated `riscv,isa` field contains a full extension string with optional versions. Linux ignores the versions and contains workarounds for buggy strings in existing implementations. The replacement is `riscv,isa-base` and `riscv,isa-extensions`, which [has a cleaner implementation](#).

The third method is to use information from the ACPI RISC-V

Hart Capabilities Table (RHCT). This encodes a full extension string with optional versions.

All these methods are available directly only to code running in Machine or Supervisor modes. To pass the information up to userspace, Linux provides `/proc/cpuinfo` and a new system call, `riscv_hwprobe`. However, the information available through these is very sparse at the moment, even relative to what is available from the hardware.

If you know the machine is using DT, then in Linux the `riscv,isa` field can be read out directly from `/sys`. **Figure 4** shows an example from a QEMU guest.

### HOW QEMU IMPLEMENTS EXTENSIONS

QEMU, a virtual machine emulator and hypervisor, can emulate RISC-V in software. This is useful when you don't have RISC-V hardware or don't have hardware that supports a particular extension. The software emulation in QEMU is called the Tiny Code Generator (TCG).

TCG works by translating [basic blocks](#) as they are encountered. The translated code is stored in a QEMU **TranslationBlock** (TB) structure and referenced through a hash table

of (CPU state, physical address). TBs persist so that code doesn't need to be retranslated, but it can be invalidated by things such as writes happening to the same code page.

TCG defines a set of basic operations like integer adds, loads, stores, labels, branches, and so on. You can recognize these when you see `tcg_gen_*` called in QEMU code. For example, `tcg_gen_qemu_ld_i64` would be called when translating a block of code and would generate a TCG instruction to do a 64-bit load and append it to the list of translated instructions.

However, anything complicated (such as CSRs or vector instructions) is translated into a call to a helper function. You will see helper functions defined in QEMU using the macro `HELPER(<name>)`. When translating, a call to the helper would be generated using `gen_helper_<name>`. Since most RISC-V extensions are complicated, they are almost always implemented as a set of helpers.

A file in the QEMU source `target/riscv/insn32.decode` describes how instruction bit patterns are decoded. Extensions must list their new instructions here. The file `target/riscv/cpu.c` contains two tables listing ISA extensions, their names, and

```
$ cat '/sys/firmware/devicetree/base/cpus/cpu@0/riscv,isa'
rv64imafdch_zicbom_zicboz_zicsr_zifencei_zihintntl_zihintpause_zawrs_zfa_zca_zcd_zba_zbb_zbc_zbs_sstc_svadu
```

**Figure 4.** An example of extension discovery from a QEMU guest

versions. This is a very useful reference for finding out which extensions have been implemented in QEMU.

At the time of writing, QEMU supports these extensions, making it probably the most capable RISC-V platform:

- The base RV32I and RV64I ISAs
- The classic extensions: **M A F D**
- Compressed instructions: **C, Zca, Zcb, Zcf, Zcd, Zce, Zcmp, Zcmt**
- The embedded extension: **E**
- The hypervisor extension: **H**
- User and Supervisor modes (but note, not Machine mode): **U S**
- Dynamic languages: **J**
- Cache management (partial): **Zicbom, Zicboz**
- Conditional ops: **Zicond**
- Read and write CSRs: **Zicsr**
- FENCE.I instruction: **Zifencei**
- Pause hint: **Zihintpause**
- Wait on reservation set: **Zawrs**
- Additional scalar FP: **Zfa**
- Bfloat16 (partial): **Zfbfmin**
- Half-width FP: **Zfh, Zfhmin**
- FP using integer regs: **Zfinx, Zdinx, Zhinx**
- Bit manipulation: **Zba, Zbb, Zbc, Zbs**
- Crypto scalar: **Zbkb, Zbkc, Zbkx, Zk\***
- Vector (mostly complete): **V, Zv\***
- Advanced Interrupt Architecture: **Smaia, Ssaia**

- State enable: **Smstateen**
- Count overflow and filtering: **Sscofpmf**
- Time compare: **Sstc**
- Hardware update of PTE A/D bits: **Svadu**
- Fast TLB invalidation: **Svinval**
- NAPOT pages: **Svnapot**
- Page-based memory types: **Svpbmt**
- T-HEAD multiple custom extensions
- Ventana custom extensions for conditional ops

### EMULATION OF RISC-V EXTENSIONS ON RISC-V

RISC-V extensions may also be emulated on RISC-V hardware using trap-and-emulate. There are two broad approaches taken:

- Modify the OpenSBI illegal instruction handler (`lib/sbi/sbi_illegal_insn.c`) to catch the illegal instruction and emulate it.
- Modify the Linux kernel illegal instruction handler (`arch/riscv/kernel/traps.c`).

The first method was used to implement a mostly complete [emulation of the Hypervisor extension](#). The second method was used to implement the Vector extension for machines that lack it.

Modifying OpenSBI has some downsides you should be aware of:

- On some machines, SBI is part of the platform firmware and might not be open source or user-replaceable.

- M-mode does not use paging, so the emulation must do its own page table walk if the extension uses virtual addresses.
- M-mode traps to SBI have extra overhead in hardware.
- There are also security and operational concerns as M-mode has complete access to the hardware, but a bug in the operating system might be limited and recoverable (e.g., by a software watchdog).

Modifying Linux has the downside that the emulation is only available for Linux and won't work for other operating systems, nor for the code that runs before Linux, such as SBL, SBI, u-boot, and EDK2.

### RISC-V IN RESEARCH

RISC-V-based microarchitectures are an important part of all FPGA-based research projects at the Red Hat Collaboratory at Boston University, in part because of its support for custom extensions. Visit these project pages to learn more:

- [DISL: A dynamic infrastructure services layer for reconfigurable hardware](#)
- [Open source toolchain optimization for FPGA CAD](#)
- [Practical programming of FPGAs with open source tools](#)

You can learn more about the role of RISC-V in research on open hardware in the RHRQ articles "[RISC-V in FPGAs: benefits and opportunities](#)" (RHRQ 4:1) and "[Fostering open innovation in hardware](#)" (RHRQ 2:2). 





# AI ON

# INTEL<sup>®</sup>



**NOW BUILD THE AI YOU WANT  
ON THE CPU YOU KNOW.**

**Learn more at [ai.intel.com](https://ai.intel.com)**



## Feature



**About the Author**  
**Tzu-Mainn Chen**  
is a Principal  
Software Engineer  
with Red Hat  
Research.

## Open research clouds get the skills to pay the bills

How do you charge for a cloud? Researchers at the New England Research Cloud have developed a stack to make understanding and charging for usage much simpler.

by *Tzu-Mainn Chen*

Universities and research institutions are increasingly embracing the cloud as a means to bring down costs and fully utilize the technical resources they have on hand. But creating and maintaining a cloud is not free, which leads to a question that is mundane and unglamorous but still critical: who is paying for all of this? This question is especially important to the [New England Research Cloud \(NERC\)](#), which provides cloud services such as OpenStack and Red Hat OpenShift to multiple member institutions, including Harvard University, the Massachusetts Institute of Technology, and Boston University.

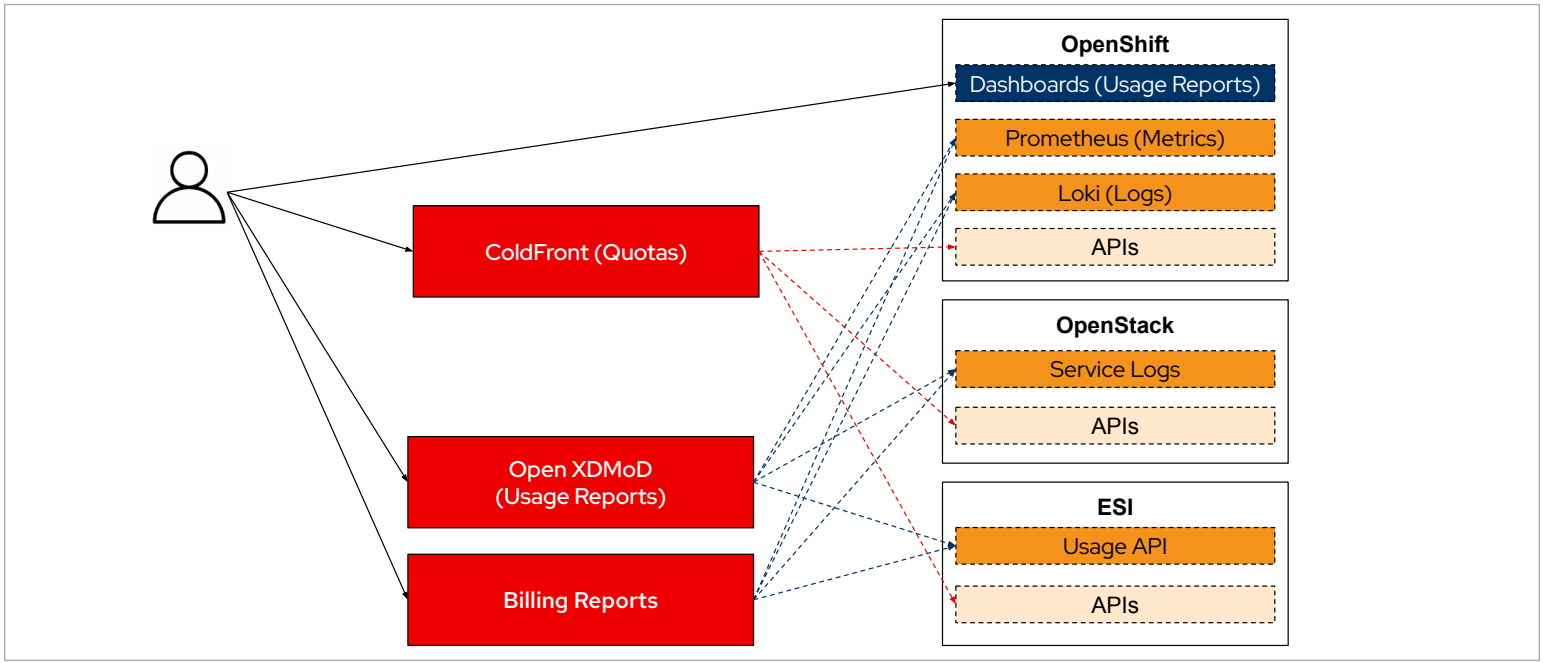
Let's start with the obvious and seemingly simple answer: the people using the service should be billed for it. But how do we define "use"? After all, no report in OpenShift says, "User A used six pieces of cloud this past week." Instead, we need to look at lower-level utilization metrics, such as CPU, memory,

or storage. Sum that up over a month, and you'll have a good understanding of the amount of resources a user has consumed.

It's not quite that easy, however, because resources are not uniform. For example, a server with a GPU is more expensive (and thus more valuable) than one without. In a cloud environment, it's also important to account for shared usage, with the aforementioned GPU potentially being shared among multiple projects.

### **SOLUTIONS**

Those are some of the difficulties present when considering billing requirements. Fortunately, there are also solutions. [OpenShift includes a monitoring stack](#) that stores various utilization metrics in time series form. This stack includes [Prometheus](#), an open source monitoring solution that allows for easy custom queries of the exact metrics we need. NERC uses Python scripts to run those



The NERC billing stack

custom queries, generating monthly usage reports to bill consumers.

We use a different approach with OpenStack, where NERC queries for VM creation and deletion events. This allows them to create a precise view of a project’s resource consumption. Similarly, [Loki](#) provides log aggregation for OpenShift, enabling operators to sift through events to create a fine-grained picture of resource usage and availability. This allows them to detect outages, during which users should not incur charges.

Custom platforms under development can also be built with billing requirements in mind. For example, NERC is in the process of deploying a bare metal cloud using [ESI](#), which allows projects

to lease bare metal nodes. These leases are modeled so that creating usage reports is as simple as running a single command.

---

Custom platforms under development can also be built with billing requirements in mind.

---

**VISIBILITY FOR USERS**

Generating usage reports is just one-half of the picture. The other is breaking down this information to users so they can make informed choices about their usage. In NERC, this all starts with the [ColdFront](#)

interface, an open source resource allocation management system that uses a [plugin architecture](#) for easy integration with the APIs provided by OpenShift and OpenStack. Users can view their quotas and request additional resources. To ease a user’s understanding of what they are requesting, NERC uses the concept of services: a collection of resources such as CPUs, storage, and external addresses that are bundled and billed together as a unit. Examples include OpenStack CPU, OpenStack GPU100, and OpenStack GPU100, the use of which each entitles a user to a different set of OpenStack resources.

It’s also important to display resource consumption data to the user. NERC’s tool of choice here is [Open XDMoD](#), an open

Generating usage reports is just one-half of the picture. The other is breaking down this information to users so they can make informed choices about their usage.

source tool used by NERC in the past and thus familiar to users. XDMoD contains native support for OpenStack; however, integrating with OpenShift requires a bit more finesse, involving a custom script that queries Prometheus to generate an XDMoD-compatible log file. In the long term, NERC may shift to native OpenShift solutions. OpenShift already provides dashboards that allow users to query Prometheus to view the vast number of available metrics easily. For storage reasons, these metrics are only provided in the short term, but a new OpenShift operator called [Curator](#) solves this problem with the help of the Koku Metrics operator, which condenses the raw data to a resolution more suitable for long-term reports.

---

It's also important to display resource consumption data to the user.

---


The final piece of the puzzle is billing: sending out a periodic invoice to users of NERC so they know how much to pay. Currently, NERC uses a custom script to create spreadsheets that are adjusted into invoices before being emailed to each user. This isn't sustainable in the long term, and they'll be moving to a new tool to automate much of this workflow. This tool will have a broad set of requirements. The most obvious one is invoicing, but nearly

as important is a web dashboard that gathers usage data across all NERC offerings for presentation to the user. The tool also aims to accommodate the notion of credits for new users or contributors.

#### **CONTINUING DEVELOPMENT**

This view into NERC's infrastructure highlights the challenges and solutions involved in the seemingly innocuous billing requirement. And there will be more as NERC continues to grow its capabilities. The aforementioned ESI bare metal cloud is being readied; further down the road, NERC may make new types of resources available in the cloud, such as FPGAs or infrastructure suitable for the wireless edge.

There's also the question of disconnected infrastructure, where metrics are not collected for reasons ranging from policy (a requirement not to collect data identifying students) to capability (disconnected infrastructure). All of these will require NERC to evaluate and update its billing practices so that usage charges match their maintenance costs, while also working with engineers to ensure that the data exists to charge users accurately.

A vast number of moving pieces go into this billing stack, but with that complexity comes the opportunity to work on any or all of the technology involved and potentially break new ground in metering and billing for the entire cloud industry. I've included links throughout this article so you can investigate further if you're curious and potentially get involved if you feel the call! 



UMass Lowell is proud to collaborate with Red Hat, a Select Preferred Partner, and celebrate more than a decade of working together on research, philanthropy and building the next generation of Red Hat's workforce.





## Feature

**About the Author  
Gordon Haff**

is a Technology Advocate at Red Hat, where he works on emerging technology product strategy, writes about tech trends and their business impact, and is a frequent speaker at customer and industry events. His books include *How Open Source Ate Software*, and his podcast, in which he interviews industry experts, is *Innovate @ Open*.

## Protecting data privacy: a look in our current toolkit

The research uses for data could be endless, but without meeting stringent privacy requirements, some of the most promising analyses may never begin.

*by Gordon Haff*

“Data is the new oil” is a shorthand generally credited to UK mathematician Clive Humby. The saying got considerable play when “Big Data” was the latest catchphrase around a decade ago. As some were quick to point out, Humby’s complete quotation notes that, like oil, data needs to be refined before we can actually use it.

Today, we see data refinement in the [form of large language models \(LLMs\)](#) and other innovations. At the same time, there’s a growing awareness of the liabilities that source data can bring to organizations, including legal action, reputational risk, and regulatory scrutiny.

Over the past few years, Red Hat Research and academic partners have been involved with projects exploring a variety of security and privacy questions related to data, including secure multiparty computation, differential privacy, confidential computing techniques (including both trusted execution environments and fully homomorphic encryption), and digital sovereignty. This article reviews these technologies and provides some updates.

### **SECURE MULTIPARTY COMPUTATION**

Secure multiparty computation (MPC) is a cryptographic protocol that allows multiple parties to jointly compute a function over their inputs without revealing their private data to each other. It is a technique for secure distributed computing that enables parties to collaboratively analyze data without compromising its confidentiality or privacy.

MPC is based on the idea of secret sharing, which involves dividing a secret into multiple shares and distributing them among the parties. Each party has a share of the secret, but no single party has enough information to reconstruct the secret on its own. This ensures that the parties cannot learn each other’s private data, even if some of them are colluding with each other. Essentially, MPC replaces a trusted third party with a cryptographic protocol.

A concrete past example of using this technique comes from research led by Boston University’s Azer Bestavros. It used payroll data from Boston-area companies that was securely collected and redistributed for [wage-gap analysis](#) without any company having access to the dataset as

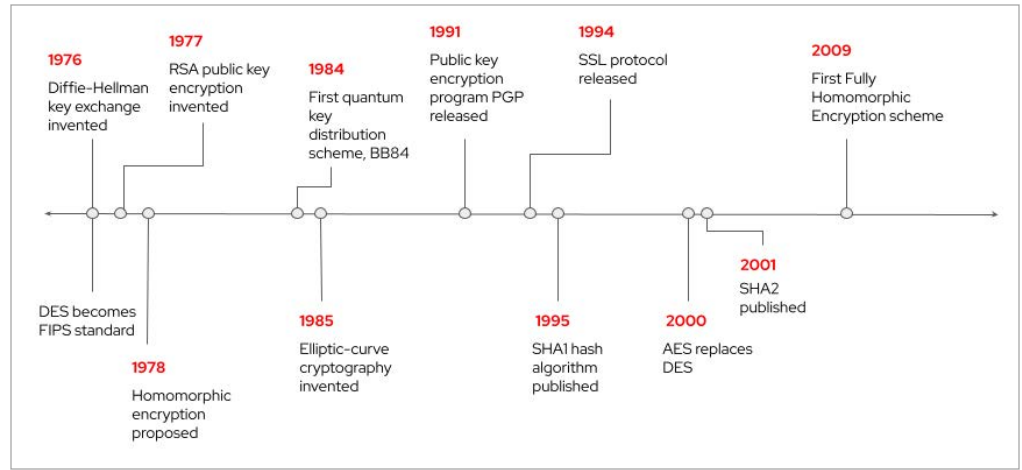
a whole. (See also “[Conclave: secure multiparty computation on big data](#)” and “[Role-based ecosystem for the design, development, and deployment of secure multiparty data analytics applications](#)”.) Work has also begun in the Red Hat Collaboratory at Boston University on developing a unikernel implementation of [Secrecy](#), a relational MPC framework for privacy-preserving collaborative analytics as a service.

## DIFFERENTIAL PRIVACY

Data from healthcare records can be a considerable boon for scientific research. However, patient data has enormous privacy implications. Even if an organization using the data for research is generally considered trustworthy, data leaks happen all the time.

The obvious solution is to anonymize the data. But this turns out to be surprisingly hard. It’s not always clear what can be used to identify someone and what can’t—especially once you start correlating with other data sources, including public ones. A widely published story from the 1990s tells how [Latanya Sweeney](#), an MIT graduate student, was able to identify a supposedly anonymized healthcare record as belonging to then Massachusetts governor William Weld after he collapsed at a local event merely by correlating it with voter records.

Organizations like the US Census have long had to deal with the challenges of publishing large numbers of tables that cut up data in many different ways. Over time, there’s been a great deal of research into the topic, which has led to the creation of various guidelines for working with data in



A timeline of encryption milestones leading to the development of FHE

this manner. One of the problems with traditional anonymization methods is that it’s often not well understood how successful they are at actually protecting privacy. Techniques that collectively fall under the umbrella of statistical disclosure control are often based on intuition and empirical observation.

However, in the 2006 paper “[Calibrating noise to sensitivity in private data analysis](#),” Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith provided a mathematical definition for the privacy loss associated with any data release drawn from a statistical database. This approach brought more rigor to the process of preserving privacy in statistical databases. It’s called differential privacy (specifically  $\epsilon$ -differential privacy). A differential privacy algorithm injects random data into a data set in a mathematically rigorous way to protect individual privacy.

Because the data is “fuzzed,” to the extent that any given response could

instead have plausibly been any other valid response, the results may not be quite as accurate as the raw data, depending upon the technique used. But other research has shown that it’s possible to produce very accurate statistics from a database while still ensuring high levels of privacy.

Differential privacy remains an area of active research. However, the technique is already in use: the [Census Bureau](#) used differential privacy to protect the results of the 2020 census.

## TRUSTED EXECUTION ENVIRONMENTS

Other techniques focus more on cloud computing, where data is inherently out of an organization’s direct control. Cryptographic techniques that protect stored and in-transit data are well established. But how about data that is currently being used?

One technique is Trusted Execution Environments (TEE). This form of confidential computing uses secure

enclaves within a processor that provide isolation and protection for sensitive data and computations. TEEs are typically implemented using hardware-based security features to create a secure environment isolated from the rest of the system. This isolation ensures that sensitive data and computations cannot be accessed or tampered with, even if the rest of the system is compromised.

A great deal of work in this area is taking place in the Confidential Computing Consortium (CCC), a project community of the Linux Foundation. There are currently [seven projects](#) under the CCC umbrella.

### **FULLY HOMOMORPHIC ENCRYPTION**

A second confidential computing technique is fully homomorphic encryption (FHE). It lets a third party perform complicated data processing without being able to see the data set itself. Homomorphic encryption is essentially a technique to extend public-key cryptography and was in fact first mentioned shortly after the RSA cryptosystem was initially invented.

A big challenge for FHE is that the technique is very expensive computationally and is mostly not practical yet. However, if realized, it would provide an additional level of protection against data leaks when using a public cloud or other service providers to analyze data sets.

One approach to this challenge has been investigated in a project at the Red Hat Collaboratory: using FPGAs to accelerate operations (see Rashmi Agrawal and Lily Sturmann,

[“Preserving privacy in the open cloud: speeding up homomorphic encryption with custom hardware”](#) in RHRQ 4:3). CPUs aren’t good at exploiting the parallelism of FHE algorithms. GPUs are better but their floating-point units end up sitting idle. Neither can effectively meet the high memory bandwidth requirements of FHE workloads. FPGAs, on the other hand, map well to the requirements while being cheaper and more future-proof than custom ASICs.

The [project’s long-term vision](#) is to design a practical, efficient hardware accelerator supporting all four of the homomorphic encryption schemes under consideration by the International Organization for Standardization, and then deploy it in the open cloud to enable privacy-preserving computing systems in the Red Hat Collaboratory.

---

A big challenge for FHE is that the technique is very expensive computationally and is mostly not practical yet.

---

### **DIGITAL SOVEREIGNTY**

The final topic isn’t a specific technology—though technology will play into some solutions—so much as it’s about geopolitical trends and concerns.

Shifting global regulations are creating a requirement for workload placement with greater control over

security and data as well as avoiding dependencies on organizations in countries that are hostile to varying degrees or may become so. Even allies may adopt regulations that are unfriendly to certain business models. The [Capgemini Research Institute](#) notes that the issues “are not new and have been gaining impetus over the past few years. However, it is a subject that is now under increasing scrutiny because of rising geopolitical tensions; changing data and privacy laws in different countries; the dominant role of cloud players concentrated in a few regions; and the lessons learned through the pandemic.”

The pursuit of digital sovereignty is a complex and evolving issue, and there is no one-size-fits-all approach. Countries will need to find their own unique path to achieving digital sovereignty, taking into account their own specific circumstances and priorities. There will certainly be domino effects. As it gains momentum, the effects on hyperscalers, local partnering requirements, and converged observability across clouds will all be open questions, as will the trajectory of regulatory regimes.

### **FURTHER READING**

You can explore all Red Hat Research projects related to privacy in the database on our website ([research.redhat.com](#)). RHRQ has also taken a broader look at this issue in past articles including [“How do we reconcile privacy with machine learning?”](#) (RHRQ 1:2, 2019) and [“Voyage into the open dataverse: an interview with James Honaker and Mercè Crosas”](#) (RHRQ 2:2 2020). 





# THERE ARE MANY UNIVERSITIES IN MASSACHUSETTS, BUT ONLY ONE FLAGSHIP FOR MASSACHUSETTS.

As the Commonwealth's flagship public research university, UMass Amherst is committed to pursuing progress for our great state in computer science, technology, engineering, and more. Learn why we've soared to #26 in *U.S. News & World Report* rankings of top-tier public universities and find your degree. Learn more at [umass.edu](http://umass.edu)

University of  
Massachusetts  
Amherst



## Column

**About the Author****Ahmed Sanaullah**

is a Senior Data Scientist and Edge Team Lead at Red Hat Research. His current focus is building open source tooling for FPGAs that enables developers to easily and efficiently create custom hardware solutions, regardless of prior hardware development expertise.

## Focus on edge: security, sustainability, and performance

Open programmable hardware, FPGAs, and RISC-V take center stage in research collaborations.

by Ahmed Sanaullah

---

*Red Hat Research and its university partners focus strategically on projects with the most promise to shape the future of how we use technology. Each quarter, RHRQ will publish an overview of our research in a specific area, such as AI and machine learning, hybrid cloud, and security. In this issue, we focus on the fast-growing field of edge computing.*

---

The edge-focused projects now in process within Red Hat Research aim to tap into the immense value proposition of edge computing through two major themes. The first theme is enabling secure and sustainable compute at the edge. This includes exploring support in Red Hat Enterprise Linux and Fedora for many potentially impactful technologies, including, but not limited to:

- ISAs, especially RISC-V
- Interconnects, for example, CXL, FPGA VirtIO
- Hardware offloads, such as eBPF
- Software stacks, including unikernels
- Hardware types, for example, CPUs, FPGAs, and microcontrollers
- Environments like automotive, IoT, and cloud high-performance computing

The second theme is motivated by a core challenge of edge computing. The “edge” in edge computing is not a discrete point—it exists

on a continuum that can go from the datacenter to the data source, with each edge facing its own blend of pressures on performance, energy, resources, and even productivity budgets. To address this, our research focuses on building an extensible framework that combines hardware reconfigurability with the use of machine learning for computation optimization and system auto-tuning, so that we get portability, scalability, and performance for a continuum of workload types, hardware sizes, deployment configurations, and so forth.

Our projects are a combination of internal efforts and academic collaborations with Boston University and the University of Massachusetts-Lowell. Here are some of the latest developments in each.

**INTERNAL PROJECTS**

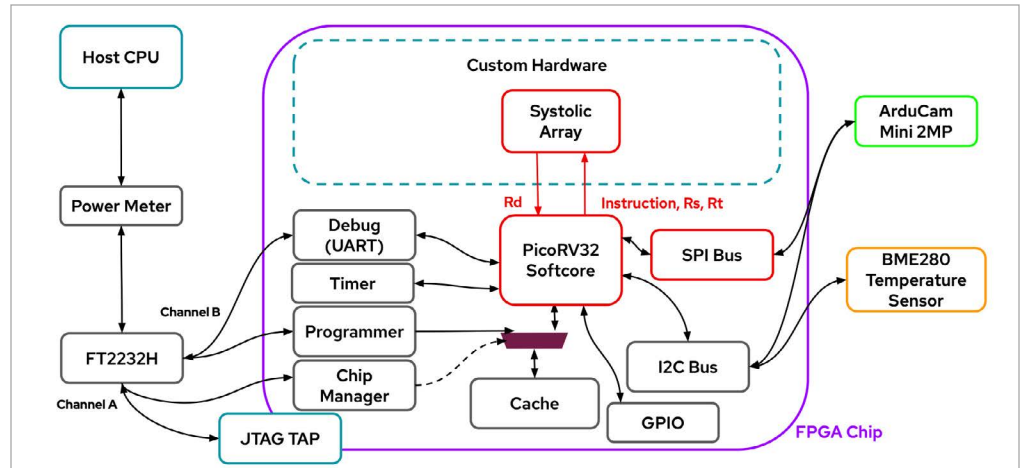
In October 2023, we demonstrated the [Dynamic Infrastructure Service Layer \(DISL\) project](#), which makes FPGA programming more practical, productive, and accessible for software

developers. As part of [the demo](#), we showcased rapid FPGA system hardware generation, secure wireless device management, and deploying workloads using RISC-V softcores. By the way, did you know Fedora 38 runs on RISC-V?

## RED HAT COLLABORATORY AT BOSTON UNIVERSITY

In March 2023, we announced the launch of the Co-Design (CoDes) Lab at Boston University as part of the Red Hat Collaboratory. The [CoDes Lab](#) aims to advance co-design with specialized hardware through systems research that requires low-level control of hardware—something that is hard to provide securely in a shared research cloud/edge infrastructure. The collaboratory also made substantial progress on several projects, with many important milestones hit:

- FPGA enumerated as a network and console device with VirtIO drivers on Fedora
- Custom compute architectures specifically designed to run eBPF packet filters with high performance and efficiency
- A working open source DDR3 memory controller with application-specific customizability
- A Relational Memory (RelMem) Controller that reduces memory bottlenecks and improves cache utilization by leveraging FPGA flexibility to give more control over how data is fetched and stored in DDR memory
- Unikernel Linux (UKL) patches pushed upstream
- A working AnnotationGym framework enabling substantial



The demo's smart video doorbell system reduces the image capture overhead through hardware offload.

performance improvement for CPUs and FPGAs through automatically annotating source code

- Graph Neural Networks (GNNs) used to generate code embeddings for High-Level Synthesis (HLS) code tuning using reinforcement learning

A paper from the UKL project was featured at EuroSys'23, a premier conference on systems software research and development. We also received a Best Demo award at VLDB'23 for the Relational Memory Controller project.

## UNIVERSITY OF MASSACHUSETTS-LOWELL


Our FPGA Place & Route (P&R) project developed benchmarking techniques for identifying bottlenecks in P&R tooling. It also demonstrated the use of machine learning to control the quality of hardware generated by open source P&R tools.

## OUTLOOK

Our focus for the coming year is to translate the above momentum into

hitting more milestones for ongoing projects, as well as tapping into new research opportunities. This includes:

- More DISL demos, such as FPGA VirtIO, eBPF networking, and RISC-V debugging
- Contributing to porting Fedora 40 to RISC-V
- Demos of application-specific memory controllers on a working open source DDR4 controller
- Advancing UKL
- Frameworks for high-quality application-specific hardware generation using Domain-Specific Languages (DSLs)
- Offloading trust-centric services such as multiparty computation and fully homomorphic encryption to FPGAs

More information on all of our projects, including publications and presentations, can be found in the [research directory](#) on the Red Hat Research website. 

# GET A LAPTOP THAT IS AI READY

AMD RYZEN™ AI TECHNOLOGY IS NOW BUILT IN

**AMD**  
**RYZEN AI**



\*Available on selected systems

