

COMMUNICATIONS

CACM.ACM.ORG OF THE ACM 09/2022 VOL.65 NO.09



Deploying Decentralized, Privacy-Preserving Proximity Tracing

Middleware 101

When SDN and Blockchain Shake Hands

How AI Is Driving the Esports Boom

Point/Counterpoint:
On the Model of Computation

SIGGRAPH ASIA 2022 DAEGU

The 15th ACM SIGGRAPH Conference and Exhibition on
Computer Graphics and Interactive Techniques in Asia

Conference 6 - 9 December 2022

Exhibition 7 - 9 December 2022

EXCO, Daegu, South Korea

COLORFUL WORLD

SA2022.SIGGRAPH.ORG

[#SIGGRAPHAsia](https://twitter.com/SIGGRAPHAsia) | [#SIGGRAPHAsia2022](https://twitter.com/SIGGRAPHAsia2022)

Sponsored by



Organized by



koelnmesse





ACM BOOKS

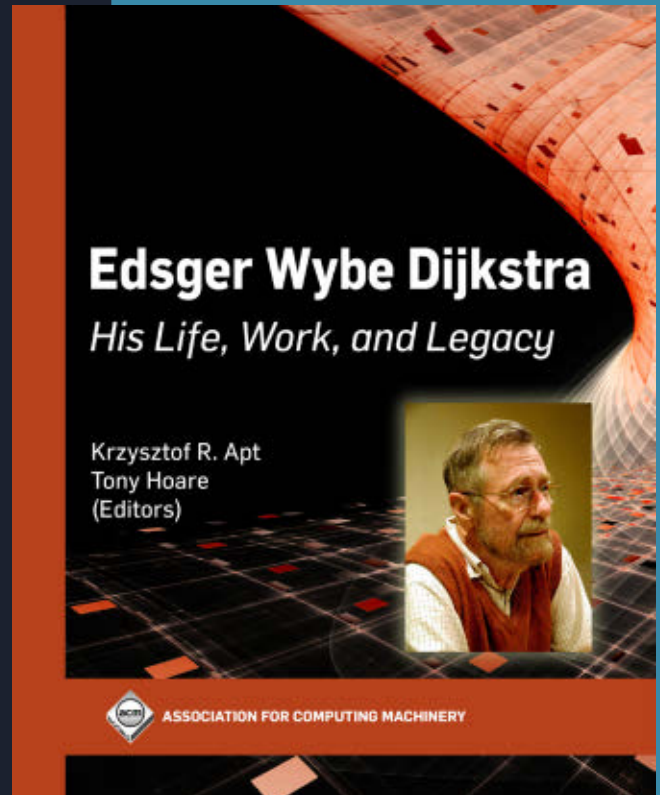
Collection II

Edsger Wybe Dijkstra (1930–2002) was one of the most influential researchers in the history of computer science, making fundamental contributions to both the theory and practice of computing. Early in his career, he proposed the single-source shortest path algorithm, now commonly referred to as Dijkstra's algorithm. He wrote (with Jaap Zonneveld) the first ALGOL 60 compiler, and designed and implemented with his colleagues the influential THE operating system. Dijkstra invented the field of concurrent algorithms, with concepts such as mutual exclusion, deadlock detection, and synchronization. A prolific writer and forceful proponent of the concept of structured programming, he convincingly argued against the use of the Go To statement. In 1972 he was awarded the ACM Turing Award for "fundamental contributions to programming as a high, intellectual challenge; for eloquent insistence and practical demonstration that programs should be composed correctly, not just debugged into correctness; for illuminating perception of problems at the foundations of program design." Subsequently he invented the concept of self-stabilization relevant to fault-tolerant computing. He also devised an elegant language for nondeterministic programming and its weakest precondition semantics, featured in his influential 1976 book *A Discipline of Programming* in which he advocated the development of programs in concert with their correctness proofs. In the later stages of his life, he devoted much attention to the development and presentation of mathematical proofs, providing further support to his long-held view that the programming process should be viewed as a mathematical activity.

In this unique new book, 31 computer scientists, including five recipients of the Turing Award, present and discuss Dijkstra's numerous contributions to computing science and assess their impact. Several authors knew Dijkstra as a friend, teacher, lecturer, or colleague. Their biographical essays and tributes provide a fascinating multi-author picture of Dijkstra, from the early days of his career up to the end of his life

<http://books.acm.org>

<http://store.morganclaypool.com/acm>



Edsger Wybe Dijkstra

His Life, Work, and Legacy

Krzysztof R. Apt
Tony Hoare, Editors

ISBN: 978-1-4503-9771-1
DOI: 10.1145/3544585

Departments

- 5 **Vardi's Insights**
Technology and Democracy
By Moshe Y. Vardi
-
- 7 **Career Paths in Computing**
The Making of an IT Strategy Consultant
By Kristian Sørensen
-
- 8 **BLOG@CACM**
Changing the Nature of AI Research
Subbarao Kambhampati considers how artificial intelligence may be straying from its roots.

Last Byte

- 104 **Q&A**
Advancing the Ability of Robots to Help
ACM Athena Lecturer Ayanna Howard considers the benefits of robotics and the potential drawbacks of overtrust.
By Leah Hoffmann

News



- 11 **Competition Makes Big Datasets the Winners**
Measurement has driven research groups to home in on the most popular datasets, but that may change as metrics shift to real-world quality.
By Chris Edwards
-
- 14 **The Road to 6G**
Looking past 5G to sixth-generation wireless technology.
By Keith Kirkpatrick
-
- 17 **How AI Is Driving the Esports Boom**
Artificial intelligence is helping the esports industry take the world by storm.
By Logan Kugler

Viewpoints

- 20 **Law and Technology**
These Are Not the Apes You Are Looking For
Considering copyright licensing issues involving non-fungible tokens to manage creative works.
By Andres Guadamuz
-
- 23 **Security**
Security by Labeling
Protecting and empowering the digital consumer.
By Andreas Kuehn
-
- 26 **The Profession of IT**
The Atlas Milestone
Celebrating virtual memory, which has made such a difference in how we approach programming, memory management, and secure computing.
By Peter Denning and Roland Ibbett
-
- 30 **Point/Counterpoint**
On the Model of Computation
Point: We Must Extend Our Model of Computation to Account for Cost and Location
By William Dally
Counterpoint: Parallel Programming Wall and Multicore Software Spiral: Denial Hence Crisis
By Uzi Vishkin
-
- 35 **Viewpoint**
Let Us Not Put All Our Eggs in One Basket
Toward new research directions in computer science.
By Florence Maraninchi

Practice



38

38 **Middleware 101**
What to know now and for the future.
By Alexandros Gazis and Eleftheria Katsiri

43 **Persistence Programming**
Are we doing this right?
By Archie L. Cobbs

Q Articles' development led by **acmqueue**
queue.acm.org



About the Cover:
In early 2020, as the COVID-19 pandemic began showing its teeth, smartphone-based digital contact-tracing technology (DCT) came to prominence to limit the virus's spread. This article discusses efforts to develop and deploy DCT, including designing underlying cryptographic protocols, developing mechanisms to ensure end-to-end user privacy, and integrating apps within public health systems. Cover by Andrij Borys Associates; photo by Goffkein.pro.

Contributed Articles



48

48 **Deploying Decentralized, Privacy-Preserving Proximity Tracing**
Lessons from a pandemic.



Watch the authors discuss this work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/proximity-tracing>

58 **Deconstructing the Bakery to Build a Distributed State Machine**
A rigorous journey from the bakery algorithm to a distributed state machine.
By Leslie Lamport



Watch the author discuss this work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/deconstructing-the-bakery>

Review Articles



68

68 **When SDN and Blockchain Shake Hands**
A survey of recent efforts to combine SDN and BC shows promising results and points to directions for future research.
By Majd Latah and Kubra Kalkan

Research Highlights

82 **Technical Perspective**
Physical Layer Resilience through Deep Learning in Software Radios
By Falko Dressler

83 **Polymorphic Wireless Receivers**
By Francesco Restuccia and Tommaso Melodia

92 **Technical Perspective**
The Effectiveness of Security Measures
By Nicolas Christin

93 **Measuring Security Practices**
By Louis F. DeKoven, Audrey Randall, Ariana Mirian, Gautam Akiwate, Ansel Blume, Lawrence K. Saul, Aaron Schulman, Geoffrey M. Voelker, and Stefan Savage



ACM, the world's largest educational and scientific computing society, delivers resources that advance computing as a science and profession. ACM provides the computing field's premier Digital Library and serves its members and the computing profession with leading-edge publications, conferences, and career resources.

Executive Director and CEO
Vicki L. Hanson
Deputy Executive Director and COO
Patricia Ryan
Director, Office of Information Systems
Wayne Graves
Director, Office of Financial Services
James Schembari
Director, Office of SIG Services
Donna Cappel
Director, Office of Publications
Scott E. Delman

ACM COUNCIL
President
Yannis Ioannidis
Vice-President
Elisa Bertino
Secretary/Treasurer
John West
Past President
Gabriele Kotsis
Chair, SGB Board
Jens Palsberg
Co-Chairs, Publications Board
Joseph Konstan and Divesh Srivastava
Members-at-Large
Nancy M. Amato; Tom Crick;
Susan Dumais; Rashmi Mohan;
Mehran Sahami; Alejandro Saucedo;
Michelle Zhou
SGB Council Representatives
Jeanna Neeff Matthews and Vivek Sarkar

BOARD CHAIRS
Education Board
Elizabeth Hawthorne and Chris Stephenson
Practitioners Board
Terry Coatta

REGIONAL COUNCIL CHAIRS
ACM Europe Council
Chris Hankin
ACM India Council
Abhiram Ranade
ACM China Council
Wenguang Chen

PUBLICATIONS BOARD
Co-Chairs
Joseph Konstan and Divesh Srivastava
Board Members
Jonathan Aldrich; Apala Lahiri Chavan;
Tom Crick; Jack Davidson; Chris Hankin;
Mike Heroux; James Larus; Marc Najork;
Michael L. Nelson; Holly Rushmeier;
Bobby Schnabel; Eugene H. Spafford;
Bhavani Thuraisingham; Julie Williamson

DIGITAL LIBRARY BOARD
Chair
Jack Davidson
Board Members
Phoebe Ayers; Yannis Ioannidis;
Michael Ley; Michael L. Nelson;
Louisa Raschid; Theo Schlossnagle;
Julie Williamson

COMMUNICATIONS OF THE ACM

Trusted insights for computing's leading professionals.

Communications of the ACM is the leading monthly print and online magazine for the computing and information technology fields. *Communications* is recognized as the most trusted and knowledgeable source of industry information for today's computing professional. *Communications* brings its readership in-depth coverage of emerging areas of computer science, new trends in information technology, and practical applications. Industry leaders use *Communications* as a platform to present and debate various technology implications, public policies, engineering challenges, and market trends. The prestige and unmatched reputation that *Communications of the ACM* enjoys today is built upon a 50-year commitment to high-quality editorial content and a steadfast dedication to advancing the arts, sciences, and applications of information technology.

STAFF
DIRECTOR OF PUBLICATIONS
Scott E. Delman
cacm-publisher@cacm.acm.org

Executive Editor
Diane Crawford
Managing Editor
Thomas E. Lambert
Senior Editor
Ralph Raiola
Senior Editor/News
Lawrence M. Fisher
Web Editor
David Roman
Editorial Assistant
Danbi Yu

Art Director
Andrij Borys
Associate Art Director
Margaret Gray
Assistant Art Director
Mia Angelica Balaquiot
Production Manager
Bernadette Shade
Intellectual Property Rights Coordinator
Barbara Ryan
Advertising Sales Account Manager
Ilia Rodriguez

Columnists
Michael L. Best; Michael Cusumano;
Peter J. Denning; Thomas Haigh;
Leah Hoffmann; Mari Sako;
Pamela Samuelson; Marshall Van Alstyne

CONTACT POINTS
Copyright permission
permissions@hq.acm.org
Calendar items
calendar@cacm.acm.org
Change of address
acmhhelp@cacm.acm.org
Letters to the Editor
letters@cacm.acm.org

REGIONAL SPECIAL SECTIONS
Co-Chairs
Jakob Rehof, Haibo Chen, and P J Narayanan
Board Members
Sherif G. Aly; Panagiotis Fatourou;
Chris Hankin; Sue Moon; Tao Xie;
Kenjiro Taura

WEBSITE
<https://cacm.acm.org>

WEB BOARD
Chair
James Landay
Board Members
Marti Hearst; Jason I. Hong;
Wendy E. Mackay

AUTHOR GUIDELINES
<https://cacm.acm.org/about-communications/author-center>

ACM U.S. TECHNOLOGY POLICY OFFICE
Adam Eisgrau
Director of Global Policy and Public Affairs
1701 Pennsylvania Ave NW, Suite 200,
Washington, DC 20006 USA
T (202) 580-6555; acmpo@acm.org

COMPUTER SCIENCE TEACHERS ASSOCIATION
Jake Baskin
Executive Director

EDITORIAL BOARD
EDITOR-IN-CHIEF
James Larus
eic@cacm.acm.org
Deputy to the Editor-in-Chief
Morgan Denlow
cacm.deputy.to.eic@gmail.com
SENIOR EDITORS
Andrew A. Chien
Moshe Y. Vardi

NEWS
Chair
Tom Conte
Board Members
Siobhán Clarke; Mei Kobayashi;
Michael R. Lyu; Rajeev Rastogi;
Albert Zomaya

VIEWPOINTS
Co-Chairs
Susanne E. Hambrusch; John Leslie King;
Jeanna Matthews
Board Members
Terry Benzel; Judith Bishop;
Florence M. Chee; Vijay Chidambaram;
Danish Contractor; Lorrie Cranor;
Janice Cury; James Grimmelman;
Mark Guzdial; Britney Johnson; Neha Kumar;
Beng Chin Ooi; Franca Rossi; Len Shustek;
Loren Terveen; Marshall Van Alstyne;
Susan J. Winter

PRACTICE
Co-Chairs
Stephen Bourne and George Neville-Neil
Board Members
Peter Alvaro; Samy Bahra; Betsy Beyer;
Terry Coatta; Stuart Feldman;
Nicole Forsgren; Camille Fournier;
Jessie Frazelle; Benjamin Fried;
Tom Killalea; Tom Limoncelli;
Kate Matsudaira; Erik Meijer;
Kelly Shortridge; Phil Vachon; Jim Waldo

CONTRIBUTED ARTICLES
Co-Chairs
m.c. schraefel and Gail Murphy
Board Members
Nathan Baker; Pramod Bhatotia;
Alan Bundy; Peter Buneman; Haibo Chen;
Premkumar T. Devanbu; Sally Fincher;
Kathi Fisler; Jane Cleland-Huang;
Yannis Ioannidis; Rebecca Isaacs;
Trent Jaeger; Gal A. Kaminka; Ben C. Lee;
Joe Peppard; Katie A. Siek; Hannes Werthner;
Ryen White; Reinhard Wilhelm

RESEARCH HIGHLIGHTS
Co-Chairs
Shriram Krishnamurthi
and Orna Kupferman
Board Members
Martin Abadi; Sanjeev Arora;
Maria-Florina Balcan; Azer Bestavros;
David Brooks; Stuart K. Card; Jon Crowcroft;
Lieven Eeckhout; Gernot Heiser;
Takeo Igarashi; Srinivasan Keshav;
Sven Koening; Karen Liu;
Joanna McGrenere; Tamer Özsu;
Tim Roughgarden; Guy Steele, Jr.;
Wang-Chiew Tan; Robert Williamson;
Andreas Zeller

Association for Computing Machinery (ACM)
1601 Broadway, 10th Floor
New York, NY 10019-7434 USA
T (212) 869-7440; F (212) 869-0481

ACM Copyright Notice
Copyright © 2022 by Association for Computing Machinery, Inc. (ACM). Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or fee. Request permission to publish from permissions@hq.acm.org or fax (212) 869-0481.

For other copying of articles that carry a code at the bottom of the first or last page or screen display, copying is permitted provided that the per-copy fee indicated in the code is paid through the Copyright Clearance Center; www.copyright.com.

Subscriptions
An annual subscription cost is included in ACM member dues of \$99 (\$40 of which is allocated to a subscription to *Communications*); for students, cost is included in \$42 dues (\$20 of which is allocated to a *Communications* subscription). A nonmember annual subscription is \$269.

Single Copies
Single copies of *Communications of the ACM* are available for purchase. Please contact acmhhelp@acm.org.

ACM ADVERTISING DEPARTMENT
1601 Broadway, 10th Floor
New York, NY 10019-7434 USA
T (212) 626-0686
F (212) 869-0481

Advertising Sales Account Manager
Ilia Rodriguez
ilia.rodriguez@hq.acm.org

Media Kit acmmembersales@acm.org

COMMUNICATIONS OF THE ACM (ISSN 0001-0782) is published monthly by ACM Media, 1601 Broadway, 10th Floor New York, NY 10019-7434 USA. Periodicals postage paid at New York, NY 10001, and other mailing offices.

POSTMASTER
Please send address changes to *Communications of the ACM*
1601 Broadway, 10th Floor
New York, NY 10019-7434 USA

Printed in the USA.



Association for Computing Machinery





Moshe Y. Vardi

DOI:10.1145/3550062

Technology and Democracy

THIS HAS BEEN a decade of ACM milestones. In 2012, ACM celebrated the Turing Centenary.^a In 2017, ACM celebrated 50 Years of the ACM A.M. Turing Award.^b On June 10 of this year, ACM celebrated ACM's 75th Anniversary (ACM@75).^c But the differences in tone were palpable. The 2012 and 2017 events celebrated the achievements of computing and its remarkable ascendance as a technology. While the 2017 event did end with a panel on "Challenges in Ethics and Computing," such challenges were a major focus in 2022, and a participant found "the whole thing a little ... depressing."

The somber tone of ACM@75 cannot be separated from concurrent events. On June 9, a U.S. House of Representatives select committee opened public, televised hearings investigating the Jan. 6, 2021 attack on the U.S. Capitol, laying out evidence of an attack on U.S. democracy orchestrated at the highest level of U.S. government. The school shooting in Uvalde, TX, on May 24 was also on many minds, remembering that an 18-year-old gunman fatally shot 19 students and two teachers and wounded 17 others. Brian Bennett wrote in *Time* magazine, "Even as America's firearm massacres provoke profound shock, change seems out of reach."

U.S. society is in the throes of deep polarization that not only leads to political paralysis, but also threatens the very foundations of democracy. The phrase "The Disunited States of America" (tracing back to Harry Turtledove's 2011 novel with this title) is often mentioned. "The U.S. is heading into its greatest political and constitutional crisis since the Civil War," wrote Robert Kagan in the *Washington Post*, raising the specter of mass

violence. How did we get here? What went wrong? Historians will spend the next 50 years trying to answer such questions, but the crisis is upon us. We need some answers now!

The last 40 years have launched a tsunami of technology on the world. The IBM Personal Computer—Model 5150, commonly known as the IBM PC—was released on Aug. 12, 1981, and quickly became a smashing success. For its Jan. 3, 1983 issue, *Time* magazine replaced its customary person-of-the-year cover with a graphical depiction of the IBM PC, "Machine of the Year." A computer on every work desk became reality for knowledge workers within a few years. These knowledge workers soon also had a computer at home. With the introduction of the World Wide Web in 1989, many millions could access the Web. The commercialization of the Internet in 1995, and the introduction of the iPhone in 2007, extended access to billions.

The socioeconomic-political context of this technology tsunami is significant. There was a resurgence of neoliberalism marked by the election of Margaret Thatcher as Prime Minister of the U.K. in 1979, and by Ronald Reagan as President of the U.S. in 1980. Neoliberalism is free-market capitalism generally associated with policies of economic liberalization, privatization, deregulation, globalization, free trade, monetarism, austerity, and reductions in government spending. Neoliberalism increases the role of the private sector in the economy and society and diminishes the role of government. These trends have exerted significant competitive pressure on the economies of the developed world. To stay competitive, the manufacturing sector automated extensively, with the nascent distributed-computing technology playing a significant role. The implications are still with us.

In my 2015 column, "Is Informa-

tion Technology Destroying the Middle Class?"^d I reported a 2014 paper by MIT economist David Autor, which argued that information technology was destroying wide swaths of routine office and manufacturing jobs, creating in their place high-skill jobs. This labor polarization appeared to bring about a shrinking middle class. Autor's data for the U.S. and 16 European countries showed shrinkage in the middle and growth at the high and low ends of the labor-skill spectrum. This polarization greatly increased income and wealth disparities.

As information technology allowed the flooding of Internet users with more information than they could digest, tech companies supplied mass customization that allowed users to concentrate on information that confirmed preconceived opinions, resulting in deeper societal polarization. This exacerbated further the "filter bubbles" that were created earlier in the broadcast media, following the abolition, in 1987, by the U.S. Federal Communications Commission, of the "Fairness Doctrine," which required holders of broadcast licenses both to present controversial issues of public importance and to do so in a manner that reflected differing viewpoints fairly.

Computing has become highly important in everyday life during the past 75 years. In addition to its many benefits, however, it has also played a major role in driving societal polarization. The somber tone of ACM@75 appropriately recognized this. □

^d <https://bit.ly/3zdWv0n>

Moshe Y. Vardi (vardi@cs.rice.edu) is University Professor and the Karen Ostrum George Distinguished Service Professor in Computational Engineering at Rice University, Houston, TX, USA. He is the former Editor-in-Chief of *Communications*.

Copyright held by author.

^a <https://turing100.acm.org/index.cfm?p=home>
^b <https://www.acm.org/turing-award-50/conference>
^c <https://www.acm.org/75-celebration-event>



CALL FOR AWARDS NOMINATIONS

ACM seeks the community's help in generating nominations for the ACM awards listed below. Please refer to each Award's Nomination page for guidelines on how to nominate: <https://awards.acm.org/award-nominations>

Nominations are due **December 15, 2022**, with the exception of the Doctoral Dissertation Award which is due **October 31, 2022**.

A.M. Turing Award: ACM's most prestigious award recognizes contributions of a technical nature which are of lasting and major technical importance to the computing community. The award is accompanied by a prize of \$1,000,000 with financial support provided by Google Inc.

ACM Prize in Computing: recognizes an early-to mid-career fundamental, innovative contribution in computing that, through its depth, impact and broad implications, exemplifies the greatest achievements in the discipline. The award carries a prize of \$250,000 with financial support provided by Infosys Ltd.

ACM Charles P. "Chuck" Thacker Breakthrough In Computing Award: recognizes an individual or group of individuals who reflect Thacker's pioneering contributions in making a surprising or disruptive leapfrog in computing ideas or technologies, and his inspiration of generations of young computer scientists. This biennial award is accompanied by a prize of \$100,000 with financial support provided by Microsoft.

ACM Grace Murray Hopper Award: presented to an outstanding young computer professional, selected on the basis of a single recent major technical or service contribution. The candidate must have been 35 years of age or less at the time the qualifying contribution was made. A prize of \$35,000 accompanies the award with financial support provided by Microsoft Research.

ACM Paris Kanellakis Theory and Practice Award: honors specific theoretical accomplishments that have had a significant and demonstrable effect on the practice of computing. This award is accompanied by a prize of \$10,000 and is endowed by contributions from the Kanellakis family, and financial support by ACM's SIGACT, SIGDA, SIGMOD, SIGPLAN, the ACM SIG Project Fund, and individual contributions.

ACM – AAAI Allen Newell Award: recognizes career contributions that have impact across sub-disciplines or that bridge computer science and other disciplines. The \$10,000 prize is provided by ACM and AAAI, and by individual contributions.

ACM Software System Award: recognizes an institution or individual(s) for developing a software system that has had a lasting influence, reflected in contributions to concepts, in commercial acceptance, or both. A prize of \$35,000 accompanies the award with financial support provided by IBM.

Doctoral Dissertation Award: presented annually to the author(s) of the best doctoral dissertation(s) in computer science and engineering, and is accompanied by a prize of \$20,000. The Honorable Mention Award is accompanied by a prize totaling \$10,000. Winning dissertations are published in the ACM Digital Library and the ACM Books Series.

ACM Distinguished Service Award: recognizes outstanding service contributions to the computing community as a whole. The contribution should not be limited to service to ACM but should include activities in other computing organizations and should emphasize contributions to the computing community at large.

Outstanding Contribution to ACM Award: recognizes outstanding service contributions to the Association. Candidates are selected based on the value and degree of service overall.

ACM Karl V. Karlstrom Outstanding Educator Award: presented to outstanding educators appointed to a recognized educational baccalaureate institution. The award is accompanied by a prize of \$10,000 with financial support provided by Pearson Education.

ACM Athena Lecturer Award: celebrates women researchers who have made fundamental contributions to Computer Science. The award carries a cash prize of \$25,000 and includes travel expenses to the conference of the recipients choosing with financial support provided by Two Sigma.

ACM Eugene L. Lawler Award for Humanitarian Contributions within Computer Science and Informatics: recognizes an individual or group who has made a significant contribution through the use of computing technology. This biennial award is accompanied by a prize of \$5,000.

Roy Levin, ACM Awards Committee Co-Chair

John R. White, ACM Awards Committee Co-Chair

Jade Morris, ACM Awards Committee Liaison



CAREER PATHS IN COMPUTING

DOI:10.1145/3546937

The Making of an IT Strategy Consultant



NAME

Kristian Sørensen

BACKGROUND

Danish

CURRENT JOB TITLE/EMPLOYER

**Partner & CEO at IT ADVISORY
(www.itadvisory.dk)**

EDUCATION

**M.Sc. in Computer
Science in 2005, Aalborg
University, Denmark**

MANY YEARS AGO, during an internship at IBM in Tokyo, my mentor asked me how I envisioned my career. I replied, “It’s important that my work is fun.” She replied rather dryly, “You can’t live off fun, Kristian.” Today, my response would be: “It’s not living without it.”

But my story really starts even further back in time—to 1987—when I was just six years old. I can still recall the exhilaration of playing with that Commodore 64 as if it were yesterday. This experience manifested as a life-long fascination with technology, and I eventually graduated with a master’s degree in computer science from Aalborg University (Denmark) in 2005. My first job, that internship at IBM Tokyo Security Research Lab, turned out to be the most remarkable experience, personally as well as professionally. Culture, expertise, people, in-

ternational collaborations, innovation, fun—it was all there, and I wanted more.

After several years as a project manager at the largest bank in Denmark, I accepted a position as CIO of the Agency for Health in Greenland, which came with IT responsibility for the entire country’s healthcare system. Greenland was breathtaking and completely different from anything I had ever experienced. Notably, there are zero roads to connect the 16 cities and 56 settlements, making IT critical to connecting hospitals, general practitioners, specialists, and patients. I became a “Swiss army knife”—CIO, with days ranging from political strategy alignment to vendor management and patching switches in the data-center. The experience sharpened my planning abilities, ingenuity, and leadership skills dramatically.

The last 12 years, however, I have been an entrepreneur and independent IT management consultant in addition to business director for the French consulting agency Devoteam.

I left Devoteam for an extended paternity leave in 2017. To my delight, former clients started calling. And that was the beginning of my current company and passion—IT ADVISORY. The concept is simple: truly independent IT advice for top management. Although IT ADVISORY is more about producing documents, coaching, and facilitation, I would not be nearly as effective at it without a solid foundation in computer science. The theoretical and mathematical foundation enables me to understand, combine, and utilize new technological innovations.

Over the last couple of years, IT ADVISORY has transformed into a truly network-based business model. This approach provides access to experts in a way that other consulting models simply do not support: while we think in terms

of expertise, teamwork, value, and outcome, our competitors are thinking CVs and billable hours. The difference is tremendous. It’s not just about assigning suitable people to each project. It’s about bringing together the very best. And this makes each day fun and inspiring.

To prospective computer science professionals with an eye on consulting, I offer the following advice:

1. *There is no work/life balance—just life.* True wealth is discretionary time. When you focus on delivering value instead of racking up billable hours, your mindset transforms, and you become inspiring to work with.

2. *Developing yourself will give you more return on investment than anything else.* While university teaches you academic skills, you have a lot to learn about yourself. Be curious and devote time to explore your strengths and weaknesses.

3. *Writing skills set you apart.* Even though you may have done a lot of writing during your studies, you have a lot to learn about management communication. When, for example, you can concisely communicate a plan with balanced insight into options and risks, you will be taken seriously. Find a seasoned mentor and get feedback from peers.

4. *Collaborate and embrace conflicts.* Surround yourself with the best people you can find—not the most agreeable. Learn to manage conflict and accept that conflict is part of life and need not be intrinsically destructive. With a base of mutual trust, conflict leads to innovation.

Your education is a life-long journey. I am still learning new things every week at age 41, and I love it! As author James Clear (*Atomic Habits*) puts it: “When you fall in love with the process rather than the product, you don’t have to wait to give yourself permission to be happy. You can be satisfied anytime your system is running.”

The *Communications* website, <https://cacm.acm.org>, features more than a dozen bloggers in the BLOG@CACM community. In each issue of *Communications*, we'll publish selected posts or excerpts.



Follow us on Twitter at <http://twitter.com/blogCACM>

DOI:10.1145/3546954

<https://cacm.acm.org/blogs/blog-cacm>

Changing the Nature of AI Research

Subbarao Kambhampati considers how artificial intelligence may be straying from its roots.



Subbarao Kambhampati
AI as (an Ersatz) Natural Science?

<https://bit.ly/3Rcf5NW>
June 8, 2022

In many ways, we are living in quite a wondrous time for artificial intelligence (AI), with every week bringing some awe-inspiring feat in yet another tacit knowledge (<https://bit.ly/3qYrAOY>) task that we were sure would be out of reach of computers for quite some time to come. Of particular recent interest are the large learned systems based on transformer architectures that are trained with billions of parameters over massive Web-scale multimodal corpora. Prominent examples include large language models (<https://bit.ly/3iGdekA>) like GPT3 and PALM that respond to free-form text prompts, and language/image models like DALL-E and Imagen that can map text prompts to photorealistic images (and even those with claims to general behaviors, such as GATO).

The emergence of these large learned models is also changing the nature of AI research in fundamen-

tal ways. Just the other day, some researchers were playing with DALL-E and thought that it seems to have developed a secret language of its own (<https://bit.ly/3ahH1Py>) which, if we can master, might allow us to interact with it better. Other researchers found that GPT3's responses to reasoning questions can be improved by adding certain seemingly magical incantations to the prompt (<https://bit.ly/3aelxmI>), the most prominent of these being "Let's think step by step." It is almost as if the large learned models like GPT3 and DALL-E are alien organisms whose behavior we are trying to decipher.

This is certainly a strange turn of events for AI. Since its inception, AI has existed in the no-man's land between engineering (which aims at designing systems for specific functions), and "Science" (which aims to discover the regularities in naturally occurring phenomena). The science part of AI came from its original pretensions to provide insights into the nature of (human) intelligence, while the engineering part came from a focus on intelligent function (get computers to demonstrate intelligent be-

havior) rather than on insights about natural intelligence.

This situation is changing rapidly—especially as AI is becoming synonymous with large learned models. Some of these systems are coming to a point where we not only do not know how the models we trained are able to show specific capabilities, we are very much in the dark even about what capabilities they might have (PALM's alleged capability of "explaining jokes"—<https://bit.ly/3yJk1m4>— is a case in point). Often, even their creators are caught off guard by things these systems seem capable of doing. Indeed, probing these systems to get a sense of the scope of their "emergent behaviors" has become quite a trend in AI research of late.

Given this state of affairs, it is increasingly clear that at least part of AI is straying firmly away from its "engineering" roots. It is increasingly hard to consider large learned systems as "designed" in the traditional sense of the word, with a specific purpose in mind. After all, we don't go around saying we are "designing" our kids (seminal work and gestation notwithstanding). Besides, engineering disciplines do not

typically spend their time celebrating emergent properties of the designed artifacts (you never see a civil engineer jumping up with joy because the bridge they designed to withstand a Category Five hurricane has also been found to levitate on alternate Saturdays!).

Increasingly, the study of these large trained (but undesigned) systems seems destined to become a kind of natural science, even if an ersatz one: observing the capabilities they seem to have, doing a few ablation studies here and there, and trying to develop at least a qualitative understanding of the best practices for getting good performance out of them.

Modulo the fact that these are going to be studies of *in vitro* rather than *in vivo* artifacts, they are similar to the grand goals of biology, which is to “figure out” while being content to get by without proofs or guarantees. Indeed, machine learning is replete with research efforts focused more on why the system is doing what it is doing (sort of “fMRI studies” of large learned systems, if you will), instead of proving that we designed the system to do so. The knowledge we glean from such studies might allow us to intervene in modulating the system’s behavior a little (as medicine does). The *in vitro* part does, of course, allow for far more targeted interventions than *in vivo* settings do.

AI’s turn to natural science also has implications to computer science at large—given the outsized impact AI seems to be having on almost all areas of computing. The “science” suffix of computer science has sometimes been questioned and caricatured (<https://bit.ly/3Ayp2PT>); perhaps not any longer, as AI becomes an ersatz natural science studying large learned artifacts. Of course, there might be significant methodological resistance and reservations to this shift. After all, CS has long been used to the “correct by construction” holy grail, and from there it is quite a shift to getting used to living with systems that are at best incentivized (“dog trained”) to be sort of correct—sort of like us humans! Indeed, in a 2003 lecture, Turing laureate Leslie Lamport sounded alarms about the very possibility of the future of computing belonging to biology rather than logic, saying it will lead us

to living in a world of homeopathy and faith healing (<https://bit.ly/3ahrSAI>)! To think that his angst was mostly at complex software systems that were still human-coded, rather than about these even more mysterious large learned models!

As we go from being a field focused primarily on intentionally designed artifacts and “correct by construction guarantees” towards one trying to explore/understand some existing (undesigned) artifact, it is perhaps worth thinking aloud the methodological shifts it will bring. After all, unlike biology that (mostly) studies organisms that exist in the wild, AI will be studying artifacts that we created (although not “designed”), and there will certainly be ethical questions about what ill-understood organisms we should be willing to create and deploy. For one, large learned models are unlikely to support provable capability-relevant guarantees—be it regarding accuracy, transparency, or fairness (<https://bit.ly/3IhdL8i>). This brings up critical questions about best practices of deploying these systems. While humans also cannot provide iron-clad proofs about the correctness of their decisions and behavior, we do have legal systems in place for keeping us in line with penalties—fines, censure, or even jail time. What would be the equivalent for large learned systems?

The aesthetics of computing research will no doubt change, too. A dear colleague of mine used to preen that he rates papers—including his own—by the ratio of theorems to definitions. As our objectives become more like those of natural sciences such as biology, we will certainly need to develop new methodological aesthetics (as zero theorems by zero definitions ratio won’t be all that discriminative!). There are already indications that computational complexity analyses have taken a back seat in AI research (<https://bit.ly/3P5sJQZ>)!

Subbarao Kambhampati is a professor at the School of Computing & AI at Arizona State University, and a former president of the Association for the Advancement of Artificial Intelligence. He studies fundamental problems in planning and decision making, motivated in particular by the challenges of human-aware AI systems. He can be followed on Twitter @rao2z.

© 2022 ACM 0001-0782/22/9 \$15.00



Association for
Computing Machinery

ACM Transactions on Internet of Things

ACM TIOT publishes novel research contributions and experience reports in several research domains whose synergy and interrelations enable the IoT vision. TIOT focuses on system designs, end-to-end architectures, and enabling technologies, and on publishing results and insights corroborated by a strong experimental component.



For further information
or to submit your
manuscript,
visit tiot.acm.org

Open for Submissions

ACM Journal on Responsible Computing

Editor-in-Chief

Kenneth R. Fleischmann

The University of Texas at Austin, USA



Original research at the intersection of computing, ethics, information, law, policy, responsible innovation, and social responsibility

The *ACM Journal on Responsible Computing* (JRC) publishes high-quality original research at the intersection of computing, ethics, information, law, policy, responsible innovation, and social responsibility from a wide range of convergent, interdisciplinary, multidisciplinary, and transdisciplinary perspectives. We welcome papers using any or a combination of computational, conceptual, qualitative, quantitative, and other methods to make contributions to knowledge, methods, practice, and theory, broadly defined.

The journal encourages contributions that address emerging areas in computing and information, including but not limited to artificial intelligence, extended reality, internet of things, machine learning, and quantum computing, as well as a wide range of ethical frameworks and perspectives from a contemporary global perspective. Authors from the Global South, from groups currently underrepresented in computing and information, and from communities adversely affected by inequities in computing and information are particularly encouraged to submit. We welcome papers that include an orientation toward culturally relevant and situated ethical perspectives and human values.

For more information and to submit your work, please visit jrc.acm.org



Association for
Computing Machinery

Competition Makes Big Datasets the Winners

Measurement has driven research groups to home in on the most popular datasets, but that may change as metrics shift to real-world quality.

IF THERE IS one dataset that has become practically synonymous with deep learning, it is ImageNet. So much so that dataset creators routinely tout their offerings as “the ImageNet of ...” for everything from chunks of software source code, as in IBM’s Project CodeNet, to MusicNet, the University of Washington’s collection of labelled music recordings.

The main aim of the team at Stanford University that created ImageNet was scale. The researchers recognized the tendency of machine learning models at that time to overfit relatively small training datasets, limiting their ability to handle real-world inputs well. Crowdsourcing the job by recruiting casual workers from Amazon’s Mechanical Turk website delivered a much larger dataset. At its launch at the 2009 Conference on Computer Vision and Pattern Recognition (CVPR), ImageNet contained more than three million categorized and labeled images, which rapidly expanded to almost 15 million.

The huge number of labeled images proved fundamental to the success of the AlexNet model based on deep



neural networks (DNNs) developed by a team led by Geoffrey Hinton, professor of computer science at the University of Toronto, that in 2012 won the third annual competition built around a subset of the ImageNet dataset, easily surpassing the results from the traditional artificial intelligence (AI) models. Since then, the development of increasingly accurate DNNs

and large-scale datasets have gone hand in hand.

Teams around the world have collected and released to the academic world or the wider public thousands of datasets designed for use in both developing and assessing AI models. The Machine Learning Repository at the University of California at Irvine, for example, hosts more than 600

different datasets that range from abalone descriptions to wine quality. Google's Dataset Search indexes some 25 million open datasets developed for general scientific use, and not just machine learning. However, few of the datasets released to the wild achieve widespread use.

Bernard Koch, a graduate student at the University of California at Los Angeles, teamed with Emily Denton, a senior research scientist at Google, and two other researchers from the University of California; the team found in their work presented at the Conference on Neural Information Processing (NeurIPS) last year a long tail of rarely used sources headed by a very small group of highly popular datasets. To work out how much certain datasets predominated, they analyzed five years of submissions to the Papers With Code website, which collates academic papers on machine learning and their source data and software. Just eight datasets, including ImageNet, each appeared more than 500 times in the collected papers. Most datasets were cited in fewer than 10 papers.

Much of the focus on the most popular datasets revolves around competitions, which have contributed to machine learning's rapid advancement, Koch says. "You make it easy for everybody to understand how far we've advanced on a problem," Koch says.

Groups release datasets in concert with competitions in the hope that the pairing will lead to more attention on their field. An example is the Open Catalyst Project (OCP), a joint endeavor between Carnegie Mellon University and Facebook AI Research that is trying to use machine learning to speed up the process of identifying materials that can work as chemical catalysts. It can take days to simulate their behavior, even using approximations derived from quantum mechanics formulas. AI models have been shown to be much faster, but work is needed to improve their accuracy.

Using simulation results for a variety of elements and alloys, the OCP team built a dataset they used to underpin a competition that debuted at NeurIPS 2021. Microsoft Asia won this round with a model that borrows techniques from the Transformers

used in NLP research, rather than the graphical neural networks (GNNs) that had been the favored approach for AI models in this area.

"One of the reasons that I am so excited about this area right now is precisely that machine learning model improvements are necessary," says Zachary Ulissi, a professor of chemical engineering at CMU who sees the competition format as one that can help drive this innovation. "I really hope to see more developments both in new types of models, maybe even outside GNNs and transformers, and incorporating known physics into these models."

Real-world performance is at the heart of the OCP's objectives, but problems can easily arise when the benchmarks themselves come to dominate research objectives. In natural language processing (NLP), the enormous capacity of the Transformer-based models built by industrial groups such as Google and OpenAI have called into question the widespread use of existing benchmarks and their datasets, such as RACE and SQuAD. As with ImageNet, the AI models often score better than humans on the benchmarks, but fail on experiments that probe performance more deeply. Investigation into the results has found that the models often rely on unintended hints in the benchmark tests themselves.

Similar problems emerged in ImageNet and other datasets, where it became apparent the models can rely more on cues provided by groupings of objects than on the target objects themselves. To keep costs down, images in

Groups release datasets in concert with competitions in the hope that the pairing will lead to more attention on their field.

visual datasets often are sourced from photo-sharing sites such as Flickr, and some categories inevitably will be poorly represented. Work presented at the 2017 Conference on Empirical Methods in NLP by Jieyu Zhao of the University of Virginia and colleagues showed how the increased prevalence of women cooking in two common datasets made an uncorrected model far more likely to associate women with that task than men. Princeton University Ph.D. student Angelina Wang and her supervisor, assistant professor of computer science Olga Russakovsky, showed in a paper presented at the 2021 International Conference on Machine Learning how models perform this kind of "directional bias amplification."

Developers of self-driving vehicles and other robots face a related problem. Most of the existing real-world footage they can use is uneventful and of little use for training systems to recognize potential problems. To train their systems to avoid accidents, they need far more unusual events, such as people running into the road or cases of dangerous driving by others in the scene. The solution to which the community has turned is simulation: creating a much wider range of scenarios than would be possible even with millions of miles of recorded driving. For image-recognition datasets that might replace ImageNet, generative adversarial networks (GANs) provide a way to create synthetic people and scenes that deliver more balanced training and evaluation datasets. However, at the current state of today's technology, this has limits; though GANs today can generate convincing faces, creating more complex scenes remains challenging.

As AI models and datasets have moved from being purely tools for research to production applications, some of which are now used for surveillance and policing, ethical issues and problems caused by biased data have become more pressing. Following an investigation by the *Financial Times* in 2019, Microsoft withdrew its MS Celeb dataset, originally created to support a facial-recognition contest at the 2017 International Conference on Computer Vision (ICCV). The dataset contained multiple images of 100,000 people, many of which had

been scraped from publicly available online sources, and the newspaper's investigation found the subjects they contacted had not given permission for their images to be used.

Amid concerns over the use of pictures of people in the dataset, the Stanford group was faced with the possibility of withdrawing ImageNet from use. Russakovsky, who works as a member of the ImageNet team, says this would have proven to be near-impossible in practice for such a widely used dataset. For example, a workshop at the 2019 ICCV utilized a downsampled version of MS Celeb almost six months after Microsoft's withdrawal announcement.

Russakovsky says the ImageNet group decided to take "small steps towards mitigating some of the concerns." This was helped by the fact that ImageNet is focused more on object recognition than it relied on personal identification, as in MS Celeb. One change was to improve the privacy of people in the background of images by blurring their faces, while ensuring models would still be able to predict accurately whether "this is a photo of a barber's chair, a Husky, or a beer bottle."

One way to moderate the influence of potentially harmful data in community sources is to limit how they are used and restrict their impact to pure research by removing the legal right for corporations and governments to use the datasets for production models. A number of researchers who have studied ethical issues in datasets have looked at other scientific fields where large datasets play an important role in research, to see which good practices ideally should be copied over into machine learning.

A number of researchers have called for greater diversity in the creation and use of datasets for machine learning. However, the high cost of developing and maintaining the collections, particularly if greater supervision of crowdsourcing is needed to reduce the introduction of biased data, may lead to a further concentration of effort in institutions with the deepest pockets. The cost of higher-quality labeling and selection may be balanced by an increasing focus on data-centric AI, where the emphasis is far more on the quality of the datasets

One way to moderate the influence of potentially harmful data in community sources is to limit how they are used and restrict their impact to pure research.

rather than on their raw size.

Work in the data-centric AI community typically is more focused on tuning datasets to the task at hand, which in turn may reduce the tendency of the machine-learning community to focus on a small number of dominant datasets, and instead utilize highly customized labeled data in concert with better metrics, rather than trying to leverage an ImageNet of anything. **C**

Further Reading

Denton, E., Hanna, A., Amironesei, R., Smart, A., and Nicole, H.

On the genealogy of machine learning datasets: a critical history of ImageNet *Big Data & Society*, July–December 1–14 (2021)

Wang, A. and Russakovsky, O.

Directional bias amplification *Proceedings of the 38th International Conference on Machine Learning*, PMLR 139, 2021.

Koch, B., Denton, E., Hanna, A., and Foster, J.G.

Reduced, Reused and Recycled: The Life of a Dataset in Machine Learning Research *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021)*

Motamedi, M., Sakharnykh, N., and Kaldewey, T.

A data-centric approach for training deep neural networks with less data *arXiv preprint - arXiv:2110.03613 (2021)*.

Papers With Code: The Latest in Machine Learning
<https://paperswithcode.com/>

Chris Edwards is a Surrey, U.K.-based writer who reports on electronics, IT, and synthetic biology.

© 2022 ACM 0001-0782/22/9 \$15.00

ACM Member News

AT THE INTERSECTION OF HUMAN AND TECHNICAL TRUST



L. Jean Camp is a professor in the School of Informatics, Computer Science, and Engineering at

Indiana University.

She earned undergraduate degrees in Mathematics and Electrical Engineering, and a master's degree in Electrical Engineering, from the University of North Carolina.

Camp notes her interests have shifted as her education has progressed; she went on to receive a Ph.D. in engineering and public policy from Carnegie Mellon University.

"If there are any graduate students reading this, please know that changing your dissertation topic is not a path to failure," says Camp.

On obtaining her doctorate, Camp joined the staff of Sandia National Laboratories in Livermore, CA. She then spent eight years as a professor of Harvard University's Kennedy School in Boston, MA, before departing to lead the security group of the newly formed School of Informatics of Indiana University.

Camp's research focuses on the intersection of human and technical trust, leveraging economic models and human-centered design to create safe, secure systems with end-to-end autonomy, encryption, and privacy controls.

Today, Camp says, "I am finishing up a five-year effort on securing the Internet of Things (IoT), to give the privacy needed for an IoT that can be trusted."

In the future, Camp would like to see a more holistic perspective toward exposure in authentication, in IoT, and in cyber-physical systems. She points out that data compilations that violate privacy are also security risks, adding that she thinks information exfiltration is risky.

"If we could just accept that, that would be fabulous," Camp says.

—John Delaney

The Road to 6G

Looking past 5G to sixth-generation wireless technology.

ALTHOUGH 5G TECHNOLOGY is still in its relative infancy, top technology companies from wireless carriers to chipset manufacturers to meta technology vendors are actively working on the development of the next milestone for wireless communications, known as the sixth generation, or 6G.

The demand for even faster networks with greater capacity is being driven by a desire to support more complex and data-intensive applications, connect an even greater number of devices and data sources, and enjoy persistent, latency-free data connections.

When fully developed, 6G technology may one day support data transfer rates of 1 terabit per second (100 times faster than the 10 gigabits-per-second hypothetical top speed offered by 5G), and network capacities of 50 to 100 times that of 5G networks, thereby allowing a much larger ecosystem of connected devices, allowing consumer, industrial, and infrastructure-based devices to operate on the same network with no adverse performance impact. Further, where 5G networks generally support latency rates of about 4 milliseconds (ms), 6G could reduce that latency to near zero, and each access point likely will be able to support multiple clients simultaneously.

However, the vision for 6G and its technical underpinnings is still being formed, as a wide range of technology companies, governments, and industry groups each work on the technology that could enable a persistent, reliable, and speedy communications infrastructure that would support mobile applications, smart cities, V2x communications, virtual and augmented reality technology, and even personal biologic-data systems.

“I think the key thing with 6G is, and I think this is quite refreshing, is that it’s going to be a network of networks, an amalgam of complementary technologies,” says Stephen



Douglas, head of market strategy for Spirent, a U.K.-based provider of automated testing and assurance solutions. “In addition to having a macro terrestrial network, you’re potentially going to have these body area networks where humans are part of it as well.” Douglas adds it is likely 6G will allow the interlinking of wireless networks with satellite, drone, maritime, and fiber-linked networks, resulting in a fully connected ecosystem.

No Standards, but Lots of Market Activity

At present, no standards for 6G have been developed or published, simply

“It’s going to be a network of networks, an amalgam of complementary technologies.”

because there is still significant work that needs to be done, particularly with respect to managing the technology’s power consumption and signal propagation for the transmission of wireless signals. The U.S. Federal Communications Commission (FCC) has allocated 95 GHz and 3 THz for early research, development, and testing, and in October 2020, the Alliance for Telecommunications Industry Solutions (ATIS), a Washington, D.C.-based standards group that develops technical and operational standards for mobile technologies, formed the Next G Alliance. Including all four major telecom companies (AT&T, T-Mobile, U.S. Cellular, and Verizon), as well as companies such as Ericsson, Facebook, Intel, LG, Microsoft, and Qualcomm, ATIS was formed to advance U.S. leadership in the rollout of 6G, focusing on the alignment of the technology’s market participants across the life cycle of 6G deployment, from R&D and manufacturing to standardization and market readiness.

Of course, other countries and companies also are working to develop technologies that could be used in the development of 6G networks.

News reports indicate Finland's University of Oulu launched the 6Genesis research project to develop a 6G vision for 2030, and has signed a collaboration agreement with Japan's Beyond 5G Promotion Consortium to coordinate the work of Finnish 6G Flagship research on 6G technologies. Meanwhile, South Korea's Electronics and Telecommunications Research Institute is conducting research on the terahertz frequency band for 6G, and Samsung announced plans to invest more than \$200 billion into areas such as chipset manufacturing to support the development of 6G's infrastructure and devices.

Within the U.S., there's a concerted effort to consider not just the technological challenges of sending data at very high frequencies, but to identify how a 6G network or networks can best serve a range of new data-intensive applications.

"What we're trying to do with 6G is to jump into this much sooner in the process and get industry, government, and academia together from North American/U.S. perspective, and think about it not just in terms of technology, but think about the applications, the societal drivers, the future spectrum needs, and all the big market [issues] that we think are going to take many years to solve," says Mike Nawrocki, vice president, technology and solutions, at ATIS. "We have to think across all these different dimensions beyond just the technology domain."

The Institute of Electrical and Electronic Engineers (IEEE)'s International Network Generations Roadmap, 2022 Edition, echoes this view, noting that "The evolution and deployment of network generations is influenced and impacted not only by emerging, evolving, and potential convergence of technologies, but also by local and world socioeconomic and health conditions (and politics)."

Commercialization of 6G is likely 8 to 10 years away, as significant work must be completed with respect to the development of suitable applications and use cases, the setting of 6G standards and metrics, and the building and testing of the network technology and infrastructure. The technological hurdles involved with delivering even more speed, reliability, resiliency, ca-

In the U.S., there's a concerted effort to identify how future 6G networks can best serve a new range of data-intensive applications.

capacity, and lower latencies than fully functional 5G networks represent a significant challenge.

High-Frequency Spectrum Propagation Limitations

A key challenge with the development of 6G technology is identifying the technological approach to transmitting faster data rates. Several approaches are under consideration, but it is likely signal multiplexing techniques that support improved spectral efficiency within the area they are deployed will be used, including techniques such as Non-Orthogonal Multiple Access (NOMA) and Massive Multiple-Input-Multiple-Output (mMIMO). While these approaches provide greater capacity (in terms of the number of users that can be served in an area), they do not improve spectral efficiency per device or user, meaning each device served would not see a higher data capacity. Further, these approaches can introduce greater system latency and can feature low energy utilization efficiency. That said, orthogonal frequency division multiplexing (OFDM)-based NOMA systems have been proposed and found to achieve reasonable gains in spectral efficiency. OFDM is used in 5G systems, and is also used in the Wi-Fi 802.11 wireless LAN standard.

Another approach to supporting higher device data throughput is to pair a conventional OFDM waveform with an additional modulation technique that can create another dimension for conveying extra data per OFDM symbol. Techniques such as spatial modu-

lation OFDM (SM-OFDM), subcarrier-index modulation OFDM (SIM-OFDM), OFDM with index modulation (OFDM-IM), and OFDM with subcarrier number modulation (OFDM-SNM) have been reported in the literature, and are likely being considered as potential techniques for delivering higher data throughput within a 6G system.

In addition to needing to support higher data rates, sending data via air interfaces generally requires higher frequencies than are used for 4G or 5G networks. Whereas today's 5G signals tend to operate in the 3.4Ghz to 3.8Ghz range, with future 5G implementations operating up to about 5Ghz, wireless 6G networks likely will use frequencies located in the terahertz or sub-terahertz range, roughly 95Ghz to 3Thz. The challenge is signal propagation; the distance radio signals are able to propagate or travel decreases as the transmission frequency rises. The shorter transmission range of projected 6G networks likely will require a denser network of base stations and repeaters to provide adequate coverage, given the relatively short (10-meter) radiation range of high-frequency 6G signals.

One potential solution to the challenges involved with using high-frequency spectrum to deliver radio waves without installing hundreds or thousands of power-hungry antennas or signal repeaters is to use reconfigurable intelligent surfaces, which can be made from materials with specialized properties that can be used to redirect 6G signals and serve as amplifiers without requiring a dedicated power source. One such material is graphene, a single-layer, hexagonal matrix-based material that can be configured to sense and reflect electromagnetic waves in a specific direction, boosting and reflecting wireless signals.

Progress is being made, according to reports on successful tests of potential 6G wireless approaches. LG Electronics and European research lab Fraunhofer-Gesellschaft used adaptive beamforming and high gain antenna switching technology to send a 15 -decibel-milliwatt (dBm) transmission in the 155GHz-175GHz band about 300 feet. China's Ministry of Industry and Information Technology is investing in and monitoring 6G research and devel-

opment in that country. A government-backed lab called Purple Mountain Laboratories announced in January 2022 that a research team had achieved wireless transmission speeds of up to 206.25 gigabits per second, albeit in a controlled environment.

6G: A Network of Networks?

Still, industry watchers say 6G is not just about air interfaces; a 6G standard likely will include multiple interface types to account for the “network of networks” approach that seems to make the most sense, given the wide variety of applications that will require network access, and the expected convergence of certain technologies.

“A lot of the [future] applications get tied via marketing to 5G or 6G, but in reality, they may run mostly over Wi-Fi, because with Wi-Fi 6, the [technology] went from OFDM to OFDMA, which is more like 6G,” says Phil Solis, research director for IDC’s Enabling Technologies team for wireless and mobile connectivity technologies and semiconductors. OFDMA (orthogonal frequency-division multiple access) is a technology in Wi-Fi 6 enabling concurrent uplink and downlink communication with multiple clients by assigning subsets of subcarriers called Resource Units (RUs) to the individual clients, supporting larger data transmission channels and greater security. “So the point is that Wi-Fi is getting better and better, too,” Solis adds.

Other experts also point to the hybrid nature of tomorrow’s applications. For example, data within a home may use the latest incarnation of Wi-Fi, which supports very high data rates. If data needs to be sent beyond the home, some type of fiber optic connection would be used to send that signal to a cell tower, because fiber optic technology may be less costly to install in densely populated areas, compared with building out a vast network of antennas and repeaters within a small geographic area, according to Paolo Gargini, chairman of the International Roadmap for Devices and Systems (IRDS) sponsored by IEEE.

“If you really want to do 6G 10 years from now, there is a lot of infrastructure that is missing and needs to be put in place,” Gargini says. “The reality is that if you really want to carry this higher frequency, like for 6G, you have to go

“A lot of the [future] applications get tied via marketing to 5G or 6G, when in reality, they may run mostly over Wi-Fi.”

to fiber,” noting the limitation of signal propagation when using very high sub-terahertz and terahertz spectrum to transmit data.

Another example of the convergence of formerly disparate networks is seen in Huawei’s approach to 6G. The Chinese tech giant announced plans to integrate terrestrial and non-terrestrial networks by launching several low- or very low Earth orbit (LEO/VLEO) satellites to form a mega satellite constellation, which will expand the coverage of the terrestrial cellular infrastructure and empower new low-latency solutions for ultra-long-haul transmission. Both networks are expected to be deeply integrated as one system where the terrestrial and non-terrestrial network nodes can be treated as base stations in a similar way, enabling users to leverage the advantages of each type in different service conditions.

Within North America and beyond, satellite 6G is projected to provide KPIs and QoS at an unprecedented level for Non-Terrestrial Networks (NTNs), according to the IEEE’s Satellite Working Group, which has identified use cases that combine several technologies, metrics, and approaches. A chapter in the association’s International Network Generations Roadmap, 2022 Edition, describes the technological hurdles and solutions to using satellite technology to support 5G and 6G networks, and “contains an enriched description of use cases combining direct satellite access and satellite backhaul, satellite IoT, mmWave for satellite networks, network management aspects, QoS/QoE, security, and recent standardization activities by 3GPP, ETSI, ITU, and IEEE.”

Another approach to improving data transfer speeds and reducing power consumption include Japan-based telecommunications company NTT’s 6G work, which includes the development and testing of a wireless network that uses an end-to-end optical communications infrastructure called Innovative Optical and Wireless Network (IOWN). The network uses photonics, or beams of light, to transmit data without converting the signals to electrical signals. Because such signal conversion is not required, NTT is targeting a 100-fold improvement to power consumption, end-to-end latency, and transmission capacity levels, compared to traditional networks.

However, it is important to note that the visions for 6G, as well as any potential standards approaches, are likely to change before its commercial rollout, given that there is a significant amount of technical and funding work that needs to be completed. From a technical standpoint, it is possible that the actual 6G architecture standard may not be that important in the future, so long as the interfaces between various networks are standardized to allow data to flow across networks.

“I would say 6G is the opportunity to really converge Wi-Fi and cellular together,” Spirent’s Douglas says. “Could we not just have sort of one universal wireless that’s connecting that can connect them to any type of backend network required? What is underneath the hood of the 6G network could be radically different, as long as the interoperability between them is standardized.” Douglas says. ■

Further Reading

ATIS’ Next G Alliance 6G Roadmap: https://nextgalliance.org/working_group/national-6g-roadmap/

IEEE International Network Generations Roadmap, <https://bit.ly/3wojXGO>

H. Tataria, M. Shafi, A. F. Molisch, M. Dohler, H. Sjöland and F. Tufvesson, “6G Wireless Systems: Vision, Requirements, Challenges, Insights, and Opportunities,” in *Proceedings of the IEEE*, vol. 109, no. 7, pp. 1166-1199, July 2021, doi: 10.1109/JPROC.2021.3061701.

Keith Kirkpatrick is principal of 4K Research & Consulting, LLC, based in New York, NY, USA.

© 2022 ACM 0001-0782/22/9 \$15.00

How AI Is Driving the Esports Boom

Artificial intelligence is helping the esports industry take the world by storm.

VIDEO GAMES ARE NOT just for fun anymore, thanks to esports (electronic sports), also known as live-streamed professional gaming.

In a typical esports competition, teams of expert game players face off against each other in a range of popular titles, like League of Legends and Dota 2. Their every move is watched, scrutinized, and analyzed by millions of viewers digitally logging into live streams, attending live events, or watching match recaps. The top players in the world, often known better by their on-screen handles than their real names, get paid a fortune in prize money. Esports teams play live events in the actual arenas used for traditional sports and rock concerts, like the Crypto.com Arena in Los Angeles.

If this all sounds a little serious for playing games, that is because it is. Esports is no longer a niche, informal pursuit; it is a behemoth entertainment category with revenues expected to reach \$1.8 billion this year, according to gaming analytics firm Newzoo.

The space is so popular that even traditional sports stars are unable to resist it. Basketball star Shaquille O'Neal is a prominent investor in multiple esports companies.

As this booming industry matures, esports increasingly relies on artificial intelligence (AI) to power many aspects of the player and fan experiences—which is having transformative effects on the industry.

“Up until a few years ago, AI was not commonly used in esports viewing, and a vast space for innovation was untapped,” says Florian Block, a Reader in Digital Creativity and esports researcher in the Digital Creativity Labs of the U.K.'s University of York.

That has all changed. Now, esports isn't just getting bigger; it's getting much, much smarter.



A Brand New (Digital) World

Esports is not simply a virtual version of traditional sports; it is an entirely new category of digitally native competition that utilizes smart machines to enhance the experience.

“Esports is deeply integrated with other tech trends such as live streaming,” says Guo Freeman, assistant professor of Human-Centered Computing in Clemson University's School of Computing, and head of the Clemson University Gaming and Mediated Experience Lab (CUGAME). “Many esports events now allow players to compete remotely, while also allowing spectators to watch across the world. The ability to be spectated live worldwide has made it a huge industry.”

The biggest difference between traditional sports and esports is the level of control that fans have, says Block. Many fans watch matches through in-game clients; that means they control their perspective of the match and the data flows they display while watching it unfold. This is not just a nice perk of

being an esports fan; it's a necessity to get the most out of the viewing experience. On-screen player actions often happen too quickly for the human eye to fully process. Action unfolds in many different places, and multiple important events can happen in different places simultaneously.

“Compared to traditional sports, many popular esports are extremely fast-paced and complex, making them hard to appreciate with the naked eye,” Block says. “The additional layers of information help fans better appreciate strategy and human performance.”

This makes AI essential at every stage of the esports experience, from the development of the games themselves to making matches more realistic, watchable, and profitable.

Faster Games, Smarter Fans

The esports industry's AI footprint is large, says Michael Naraine, an assistant professor of Sport Management on the Faculty of Applied Health Sciences at Brock University in Ontario, Canada. At the most fundamental level, AI is a critical feature of any esports game with simulated enemies or players, says Naraine. These AI-generated bots interact with or battle players, relying on sophisticated AI to make decisions and take actions. AI-generated bots are also used by esports teams to train and practice facing off against skilled avatars.

However, that is only the beginning of AI's contribution to the industry. AI also is being used to make esports more lifelike.

“Publishers like EA and Ubisoft have been harnessing AI to improve scale and motion for esports titles, and that has a direct impact on how esports athletes strategize and develop their acumen,” Naraine says. As a game more closely mirrors the physics

of reality, esports athletes have less opportunity to exploit gaps. In addition, publishers rely on AI to predict which maps and characters are most likely to yield specific outcomes. This data allows them to keep games balanced.

Block points out that AI is not used just to make games playable, but also to make them more watchable.

AI systems are used to help fans follow the action. The action is not just fast and complex; it also unfolds in multiple areas simultaneously. Piecing together everything going on in real time can be difficult for human commentators—and near-impossible for fans.

Block's own research team has created an AI-powered analytics engine for Dota 2, a popular esports. The engine learns from previous matches, then generates analyses, breakdowns, and storylines about live matches to help fans follow along.

AI techniques such as clustering and archetype analysis also are used in the industry to identify different playstyles, which in turn allows better context around a player's performance. Random forest algorithms link performance indicators to outcomes, which helps fans understand which parts of gameplay matter most.

And when the dust settles, AI also helps fans get more out of match recaps. Block's team also used natural language processing to train a model with vocabulary specific to Dota 2. "This generates dynamic match recaps that are tailored to different players, teams, and playstyles," he says.

AI-powered event prediction also helps enhance watchability. It is often harder for esports viewers to gauge a team's progress during a match than it is to, say, understand which team is winning in football. Conditions often change in the blink of an eye, and that can completely change a match's outcome. Deep neural networks and random forest algorithms are employed to help fans anticipate match outcomes, so they can more easily follow along.

"Today, win prediction algorithms are embedded in various esports natively," says Block. (In a twist, these varieties of AI are now also being used in traditional sports like the Australian Open.)

In the case of esports, AI prediction does not ruin the experience; it

"It is often harder for esports viewers to gauge a team's progress during a match than it is to, say, understand which team is winning in football."

enhances it. Understanding the likelihood of an outcome leads to a better fan experience when unexpected upsets happen. It also allows virtual camera operators to better home in on an upcoming player death or match upset, which is not always easy to track in real time manually. In fact, Block and his team trained a deep neural network for this purpose: it predicts the death of a virtual avatar 10 seconds before it happens—with over 80% accuracy.

Last, but not least, AI is even used on the industry's business side, to analyze which screens fans are looking at, as well as which ads are being watched and for how long.

"Esports naturally lends itself to technological advancement," says Naraine. "It is viewed as a less risk-averse sport property to trial new tech developments than traditional, analog sport."

The Weird, Wonderful Future of AI in Esports

As the games improve in speed and realism, AI is increasingly necessary to the industry's future.

"I believe AI is truly the next important marker in the evolution of esports," says Naraine. "It's going to be critical to making esports more challenging and fun simultaneously."

Using AI to fully analyze—and increase—the difficulty of games does not just keep the space fun, fresh, and interesting, but also supports a healthy betting ecosystem, he says. AI will be used increasingly to document gameplay, which increases gambler confidence that games are not rigged.


Not to mention, the ability of es-

ports AI to make ever more realistic visuals means we could see the traditional world of sports getting a makeover, too. "It lends itself to a future where we'll have virtual and live cars and horses racing side-by-side, or perhaps augmented worlds where fans are virtually ported into Yankee Stadium to watch the eYankees play the Red Sox," says Naraine.

This is where Block sees AI in esports entering "metaverse" territory. In fact, AI will be key to creating the types of augmented and virtual experiences that could fuel a world where we live, work, and play predominantly in simulated worlds.

"AI will be a key technology to enable the generation of rich entertainment worlds, facilitate the exchange of knowledge and entertainment within them, and enable fans to create and share meaningful versions of their experiences with each other," he says.

There's even research to suggest the technology behind esports as a whole could actually make us better humans. Instead of esports technology isolating us from reality or distancing us from others, it can actually produce positive behavior in people, says Freeman. Her body of research has documented how most esports players started out as strangers but, through in-game teamwork and collaboration, they actually come to exhibit these qualities more in person.

"Such in-game informational and instrumental support often lead to emotional and esteem support," she says. "These different types of support functions not only remain within the context of esports—they bleed out into in-person interactions and relationships." 

Further Reading

Dota 2: <https://www.dota2.com/home>

Newzoo's Global Esports & Live Streaming Market Report 2021, *Newzoo*, Mar. 9, 2021, <https://newzoo.com/insights/trend-reports/newzoos-global-esports-live-streaming-market-report-2021-free-version>

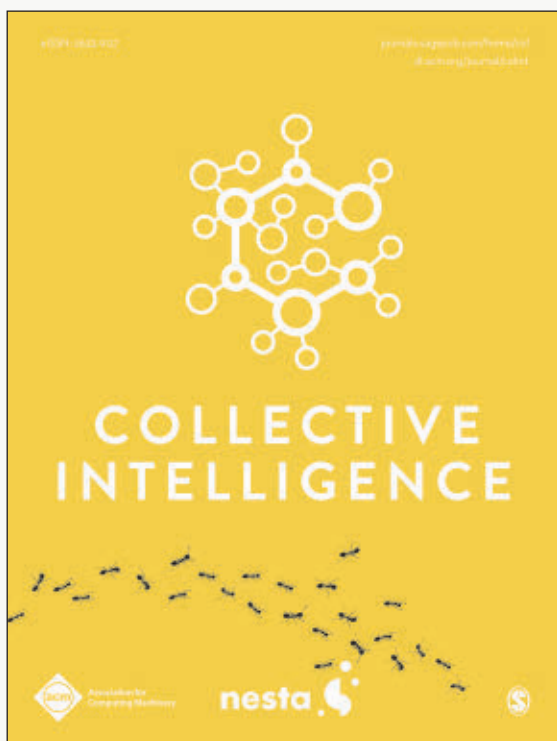
Top 100 Highest Overall Earnings, *Esports Earnings*, <https://www.esportsearnings.com/players>

Logan Kugler is a freelance technology writer based in Tampa, FL, USA. He has written for more than 60 major publications.

© 2022 ACM 0001-0782/22/9 \$15.00

A New Journal from ACM

Co-published with SAGE



Collective Intelligence, co-published by ACM and SAGE, with the collaboration of NESTA, is a global, peer-reviewed, open access journal devoted to advancing the theoretical and empirical understanding of collective performance in diverse systems, such as:

- human organizations
- hybrid AI-human teams
- computer networks
- adaptive matter
- cellular systems
- neural circuits
- animal societies
- nanobot swarms

The journal embraces a policy of creative rigor and encourages a broad-minded approach to collective performance. It welcomes perspectives that emphasize traditional views of intelligence as well as optimality, satisficing, robustness, adaptability, and wisdom.

Accepted articles will be available for free online under a Creative Commons license. Thanks to a generous sponsorship from NESTA, Article Processing Charges will be waived in the first year of publication.

For more information and to submit your work,
please visit <https://cola.acm.org>

► James Grimmelman, Column Editor

Law and Technology

These Are Not the Apes You Are Looking For

Considering copyright licensing issues involving non-fungible tokens to manage creative works.

IMAGINE YOU WANT to stream some music. On today's Web, you would sign up for a service such as Spotify or Apple Music. These platforms have obtained copyright licenses from record companies and artists, and they offer you that music for a monthly subscription. The music streaming services are centralized intermediaries. They exist to connect musicians and fans, and in exchange they take a substantial cut of the money.

But a growing number of technology enthusiasts have a different vision, which they call Web3. To them, it "represents the next phase of the Internet and, perhaps, of organizing society."^a One of the pillars of the Web3 vision is tokenization: using representing ownership of different assets using cryptographic tokens that can be exchanged on a blockchain or other decentralized system. Only the person who knows the private key associated

with a token can use or transfer it. A token can be used to represent anything, from frequent-flyer miles to hotel reservations. By transferring a token from user to user, it can record who owns an associated asset.

In a Web3 world, your music experience would be mediated not by Spotify but by tokens. Instead of signing up for a music service, you would buy a token directly from the artist. The token would represent your right to listen to the music. The token's cryptography would be tied directly into the digital rights management protecting the music, so that only token owners would be able to listen. In other words, the token living on a decentralized blockchain would let you and the artist automatically cut out the middlemen like Spotify, and maybe even record companies.

One of the sectors receiving particularly intense Web3 interest and investment is the creative industries. In this area, the tokenization push is being driven by non-fungible tokens

(NFTs), cryptographic tokens that represent a unique asset. One banana is pretty much like any other banana, but a Picasso portrait and an Ai Weiwei sculpture are radically different. The tokens representing them are not interchangeable, or fungible, hence the name.

The most famous NFT project is the Bored Ape Yacht Club, whose collection of "Bored Ape" NFTs have been selling for hundreds of thousands of dollars. Each of the 9,999 Bored Ape NFTs consists of a token on the Ethereum blockchain linked to a JPEG cartoon drawing of an ape. The JPEGs were procedurally generated with different combinations of traits, including jackets, hats, and facial expressions. While they all resemble each other, each individual Bored Ape is unique, a bit like the different cards in a trading-card set.

There is currently a push to move the economy in the direction of a wider use of tokens, and this is being driven mostly by a combination of Silicon

^a See <https://bit.ly/3NXRIV1>



Valley venture capitalists and cryptocurrency holders and investors. If the funders, developers, and artists pushing NFTs and Web3 get their way, the media landscape will look very different from what it looks like now.

This might sound like a great idea, but only until you start looking in detail at how it would actually work. As soon as you do, there are serious problems at every practical level.

A Stolen Ape

Actor Seth Green, who is most famous as the creator of the comedy series “Robot Chicken” and for playing Scott Evil in the *Austin Powers* movies, has been a vocal proponent of NFTs. Recently, he has been developing a new series called “White Horse Tavern,” which will feature a mixture of live-action actors and animated ones. As Green explained in an interview, “it doesn’t matter what you look like, what only matters is your attitude.”

Part of the gimmick of “White Horse Tavern,” is that the animated

characters will be based on NFTs. For example, the protagonist, Fred Simian, will be based on Bored Ape #8398—whose traits include a skeleton shirt and a halo—which Green bought last year. Green could do this because the Bored Ape NFTs come

If the funders, developers, and artists pushing NFTs and Web3 get their way, the media landscape will look very different from what it looks like now.

with a copyright license letting their owners use the artwork to create “derivative works” such as TV shows.

Unfortunately for Green, in May 2022 he fell victim to a phishing attack. A unknown hacker was able to obtain access to the private key securing Green’s Bored Ape NFT and sold it to a third party for approximately \$200,000. The buyer of the stolen ape NFT, who uses the Twitter handle DarkWing84, said they bought it in good faith and have no intention of parting with it.^b

For a traditional video series based on traditional copyright licensing, a theft like this would be no big deal. The animators who work on *Robot Chicken* and other series have contracts letting the producers use their creations in the show. If the producers want to include an existing character, they will get a license. These are written documents, drafted by lawyers, and in the names of all of

^b See <https://bit.ly/3OYBXir>

the parties. Even if the actual physical document is stolen, the contract is still valid. Everyone still knows what it said because they all keep copies, and in many countries important documents are notarized or recorded to provide extra security.

But in the Web3 world, licenses, registrars, notaries, and contracts are supposed to be replaced by the blockchain and tokens. The blockchain never lies, goes the theory: whatever the blockchain says is the absolute truth. Whoever possesses the NFT owns the rights that come with it. That is what it means to “tokenize” rights. Code is law.

In this world, when Seth Green loses his Bored Ape NFT to a hacker, he also loses all his rights to the artwork associated with it. Now, Fred Simian, the star of “White Horse Tavern,” cannot be played by Bored Ape #8398, because Seth Green no longer owns the rights. DarkWing84 does. Unless Green can buy back the rights for whatever DarkWing84 demands, the show will have to be reanimated with a different lead. Either way, a lot of his investment in developing the show and character was lost when his wallet was hacked.

In a Web3 NFT world, no production would ever be safe. Suppose Disney gets an NFT to license a pop song to play over a training montage in an upcoming Marvel movie. Filming and production go ahead, the movie is playing in theaters and on Disney+, and then the NFT is stolen. Does Disney have to pull the movie until it can recut the scene?

What Are NFTs, Really?

Situations such as described here arise because there is little clarity regarding what one actually gets when buying an NFT. At its most basic, an NFT is just a chunk of data stored on a blockchain. Even if that data includes a link to some other asset, such as a JPEG of an ape or a pop song, the NFT is not the asset itself.

When a person buys an NFT, they are buying the NFT, not the artwork or music. Ownership of the token does not guarantee in any way ownership of the asset. People use all sorts of analogies to explain this. Some say an NFT is like a receipt, others say it is

Since an NFT is just a chunk of data, the transfer of an NFT does not confer any rights other than control of that chunk of data itself.

more like a treasure map. Either way, the token is not the thing. A receipt is not pretty to look at; a treasure map is not a treasure. An NFT is a chunk of metadata, and that is the extent of it.

For some uses, this is enough. Some people give value to NFTs because owning one gives them bragging rights, like owning a rare stamp or other collectible. Other people value NFTs because they like being part of a community, or it helps them feel a connection to an artist. These are all legitimate reasons to hold NFTs.

But if the intention is to manage rights, then NFTs are a uniquely bad tool for the job. Seth Green’s story illustrates one problem. An NFT is what lawyers call a “bearer” instrument, like a check that can be cashed by whoever physically has it. Checks can be stolen and cashed by the “wrong” person, and they can also be lost. Similarly, the owner of an NFT could lose the private key associated with it, and end up unable to prove they own it. Imagine paying big money for a copyright license and then being sued for copyright infringement anyway.


Another problem is that copyright licensing law is not designed for NFTs. If you buy an oil painting from a gallery, or a book from a bookstore, you do not get the copyright too, just the specific painting or book. If you want to buy the copyright, you need something more. Copyright law requires “formalities” such as signed written contracts for this and many other common licensing transactions.

But since an NFT is just a chunk

of data, the transfer of an NFT does not confer any rights other than control of that chunk of data itself. While the blockchain transaction transferring an NFT to someone else must be “signed” using a private key, this is not likely to meet copyright law’s requirements to also transfer copyright ownership. The result of this is that there are thousands of people who may very well believe they own some art, video, or sports highlight, when in reality they just own a link to those digital items. If you actually want to buy a copyright, the best way to do it is still the traditional way, not with a smart contract but with a “dumb” paper contract, signed and in writing.

Finally, even though NFTs are supposed to simplify licensing by cutting out the middleman, they can often make it more complicated by adding their own intermediaries, platforms, and third parties. If an artist sells an NFT, she might list it on a gallery website with the terms and conditions of the license, and sell it to a buyer on a marketplace such as OpenSea with its own terms and conditions. Then that buyer might resell it through a different marketplace with a third set of terms and conditions. The downstream buyer may not have agreed to the terms on OpenSea or the gallery website, or even be aware they exist. Now think about transactions further down the distribution chain, and the author’s original intentions when licensing the work are lost.

Conclusion

There could be perfectly legitimate uses for NFTs. It seems inevitable that we will start seeing tokens in different sectors of our lives, particularly given the economic investment that is pushing us toward Web3 (not to mention the nebulous concept that is the metaverse, which is a subject for another time). But NFTs are not good at managing rights to digital assets. Copyright licensing is the point where grandiose claims about a Web3 future of using of NFTs to manage creative works crashes back into reality. 

Andres Guadamuz (a.guadamuz@sussex.ac.uk) is a Reader in Intellectual Property Law at the University of Sussex, Brighton, U.K.

Copyright held by author.

► Terry Benzel, Column Editor

Security

Security by Labeling

Protecting and empowering the digital consumer.

EMPOWERING CONSUMERS TO make risk-informed purchasing decisions when buying Internet-of-Things (IoT) devices or using digital services is a principal thrust to advance consumer cybersecurity. Simple yet effective labels convey relevant cybersecurity information to buyers at the point of sale and encourage IoT vendors to up their cybersecurity game as they now can recoup their security investments from risk-aware buyers. These dynamics benefit consumers and the industry alike, resulting in better, more resilient cybersecurity for all.

Consumers are insufficiently aware of risks emanating from IoT and are ill-equipped to manage them. For all the much-heralded benefits of consumer IoT to come true, the industry must ensure all the smart home appliances, connected thermostats, and digital services are secure and can be trusted. The industry has for long been criticized for not paying sufficient attention to the cybersecurity of its products. Concerns over security were pushed aside, yielding precedence to shorter time-to-market and higher corporate profits. Less time for testing translates into insecure products in residential homes.

The full cost of insecurity is on display when consumers, industry, and governments must respond to and clean up after cyber incidents. The toll of consumer cybercrime alone adds up to more than 100 billion USD per year globally.⁴ The industry, with support from government, must find ways to put IoT security front and center and make the necessary up-front invest-



ments that enhance consumer cybersecurity and lower cost to everyone.

Lack of Information Drives Cyber Insecurity

Consumer cybersecurity is suffering from information asymmetry, the skewed appraisal of the quality of a property that Nobel Laureate economist George Akerlof described in his seminal writing “The Market for Lemons: Quality Uncertainty and the Market Mechanism.”¹ In the secondhand car market, Akerlof observed, buyers of used cars could not tell good cars from bad ones and thus differentiated the product on price alone, rather than including the quality of the preowned vehicles in their

purchase decision-making. Sellers had no incentive to sell higher-quality cars since they could not find buyers willing to pay a higher price. Thus, the information asymmetry between the seller, who knows the quality of the car, and the buyer, who cannot assess the quality of the car, led to a market of lemons, a degraded market of subquality cars, which frequently break down and are in constant need of expensive repairs.

The consumer IoT marketplace faces a similar conundrum. Buyers cannot discern a secure Internet-connected camera from its insecure, cheaper alternative. With no market demand, IoT manufacturers have no incentive to invest in cybersecurity. All that is left

The challenges on the way to consumer IoT cybersecurity labeling are considerable but not insurmountable.

is to compete on price, further incentivizing the reduction of security to save on cost and hindering the much-needed consumer adoption of secure Internet-connected devices and services. Adding transparency by means of a recognized, trusted cybersecurity label can break this vicious cycle, empower buyers to make risk-informed purchases, and allow vendors to reap the rewards of their cybersecurity investments by marketing to security-aware customers. In fact, research shows that a sizable portion of consumers is willing to pay a 30% markup for secure IoT products.⁵

Making Obscure Certifications Consumer Friendly

Traditionally, certifications in the ICT industry have been used to attest to the conformance of products or services with standards. The number of standards and certifications relevant to the protection of the digital consumer is growing. The U.K.'s Internet of Toys Assurance Scheme is a case in point. It certifies a cybersecurity and data protection baseline for interconnected toys to protect children from digital harm. Another consumer-centered example is The Digital Standard, an initiative spearheaded by a U.S. non-profit collective, that designed a framework for evaluating cybersecurity, privacy, governance, and product ownership of consumer IoT. Technical standards and security baselines, including ETSI EN 303 645 consumer cybersecurity IoT baseline requirements and its test specification ETSI TS 103 701, the NIST IoT device cybersecurity capability core baseline NISTIR 8259A and the foundational cybersecurity activities for IoT device manufacturers NISTIR 8259 as well as the C2 consensus on IoT device

security baseline capabilities, undergird consumer IoT certification.

While these are important contributions toward advancing consumer cybersecurity, certification remains a rather obscure matter to most end users and even less provides clarity on what it implies for cybersecurity. When consumers buy a new refrigerator or a dishwasher at a retailer, they hardly inquire about technical industry certification. But what has successfully emerged for many decades now are labels that translate selected technical facts into easy-to-understand information that consumers can use to compare products.

To account for distinct product types, informational needs, and contexts of use, conventional labels come in different forms and styles. Binary labels or seals of approval denote the existence of a property, such as the USDA Organic Seal. Graded labels use a scale to indicate levels of quality, such as New York City's restaurant sanitation letter grades A, B, or C. Finally, descriptive labels offer the most information and highlight key properties. The FDA Nutrition Facts label falls in this category. The purpose and objective of a cybersecurity label will determine which type is most applicable to inform the digital consumer in an effective way.

Early Movers Leading the Way

Singapore's efforts to create a cybersecurity label have garnered much attention and have been widely cited as an example to use market forces to even out the information asymmetry and thus strengthen cybersecurity. Under the purview of the Cyber Security Agency of Singapore (CSA), the Cybersecurity Labeling Scheme for consumer smart devices was launched in October 2020 and offered certification and labeling for Wi-Fi routers and smart home hubs. Labels of the voluntary scheme fall into four categories with distinct security requirements; Levels 1 and 2 rely on vendor self-certification, whereas Levels 3 and 4 require independent assessment by approved test labs. CSA-issued labels are valid for a period of up to three years, during which the manufacturer will provide security updates to consumers. Seven months into the launch, eight devices from five vendors received a label, all of them at Level 1. One year later, the number rose to 138

devices that attained a label on all but Level 3, with many more products from a diverse set of manufacturers waiting in the certification pipeline.

Finland's Transport and Communications Agency was the first to award its own consumer cybersecurity label, Tietoturvamerkki. It has issued labels to 14 products that met Finland's National Cyber Security Centre information security requirements based on ETSI EN 303 645. Finland and Singapore signed a mutual recognition agreement, which allows manufacturers to receive certification for both markets in a single certification process.

Through the Executive Order "Improving the Nation's Cybersecurity," U.S. President Biden directed in 2021 the National Institute of Standards and Technology (NIST) to study IoT labeling, which resulted in recommendations for cybersecurity criteria for consumer IoT products and software.⁶ In contrast to Singapore and Finland, NIST is not establishing a government labeling program, but its recommendations and standards aim at enabling private actors to fill the gap.

IoT trust marks for consumer IoT devices were also discussed in the U.K. and Australia. Eventually, the U.K. favored a mandatory cybersecurity baseline for connected consumer devices instead. In Australia, a private provider piloted the IoT Security Trust mark in 2021. Certification providers with global reach play an important role in the adoption of labels by industry. Known for its safety and quality testing, the independent Underwriters Lab (UL) introduced a label-based IoT Security Rating. To that end, UL works with manufactur-

Buyers cannot discern a secure Internet-connected camera from its insecure, cheaper alternative.

ers and retailers to certify smart washers, HVAC systems, refrigerators, and other home appliances.

It should be noted these are not the first attempts for an IoT label. Starting in 2015, the Cyber Independent Testing Laboratory aimed at providing public product cybersecurity ratings but has since then pivoted to surveying firmware security at scale. The Trustable Tech mark, which closed down in 2020, was another effort in this category, that ran into difficulties to reach critical mass and find a sustainable business model. It is a cautionary note that the failure of a label can undermine trust and divert much-needed industry investments in consumer cybersecurity.

Interventions to Rejuvenate Cybersecurity Market Forces

One does not need to be an economist to determine there exists a market failure for cybersecurity. Simply put, market participants do not compete on security. This is a case for justifiable and, frankly, much-needed government intervention to strengthen cybersecurity. Regulators can set the conditions for labels to successfully leverage market mechanisms to overcome information asymmetry and prevent a market of cybersecurity lemons. Defraying manufacturers' expenses for labeling through government cost reimbursement is one way to accelerate label adoption. A recent British regulatory cost estimate for physical label implementation ranged from 3,000 GBP per company on the lower end to 500,000 GBP for the largest manufacturers.² But this is not where it stops. A series of root causes of cybersecurity externalities need to be addressed, think of it as security essentials.

To that end, governments have made important contributions to overcome information asymmetries and reduce externalities, in some cases through regulatory means. For instance, NIST developed an IoT device cybersecurity capability core baseline that helps boost consumer cybersecurity. Per California IoT Law (SB-327 Information privacy: connected devices), manufacturers are required to equip each connected device with a unique password and other reasonable security features when selling to

Consumer cybersecurity can no longer be ignored.

consumers in the Golden State. Building on the Code of Practice for Consumer IoT Security, the U.K.'s pending Product Security and Telecommunications Infrastructure bill would tighten the industry's responsibility for consumer cybersecurity. Non-compliance with relevant standards—such as the globally applicable ETSI standard EN 303 645: Cyber Security for Consumer Internet of Things: Baseline Requirements—would be punishable to the greater of 10 GBP million or 4% of the manufacturer's annual global revenue.

Multistakeholder Collaboration to Overcome Challenges

The challenges on the way to consumer IoT cybersecurity labeling are considerable but not insurmountable. Building upon existing IoT cybersecurity standards, the relevant labeling infrastructure, processes, and governance mechanisms need to be established. Governments should focus on setting the conditions upon which the private sector can jumpstart labels and move quickly to practical solutions that small and large IoT manufacturers can implement. The market must come with sustainable business models for labels. Leveraging sector-specific knowledge in trade associations and harnessing certification expertise and capacity in the private sector are key ingredients to scale IoT consumer labels successfully. The adoption and diffusion of labels by industry and consumers is another, perhaps the most critical, step in advancing consumer cybersecurity. Securing the cooperation of large national retailers in this step can help reach critical masses for IoT cybersecurity labeling in industry and among consumers, as they have the power to decide what secure IoT they put on their physical and digital shelves.

The nascent labeling regime also must address a range of policy design decisions. For instance, a label's static character poses a challenge to the ever-changing nature of cybersecurity. Should a label include an expiration date or be subject to reoccurring assessments? Should a competent authority be able to revoke a label or even stop the sale of a consumer IoT device that upon testing shows severe security vulnerabilities? Should expert consumers get access to testing and certification results to make security decisions in line with their risk profile? Experiences from other domains, such as food, energy, and health sanitation should help inform these design decisions, but they also provide hints on how labeling systems may evolve.³

Consumer cybersecurity can no longer be ignored. It is time for governments, manufacturers, trade associations, and consumer advocacy groups to take immediate steps to establish guidelines and identify best practices but also consider financial incentives for setting up IoT labeling systems, driving industry adoption, and ensuring consumer label recognition and education with the ultimate goal to advance consumer cybersecurity. Close multistakeholder collaboration among consumers, industry, and government is a must to secure IoT devices and for consumers to justifiably trust in the digital future. ■

References

1. Akerlof, G.A. The market for "lemons": Quality uncertainty and the market mechanism. *The Quarterly Journal of Economics* 84, 3 (1970), 488–500; <https://bit.ly/305NMLA>
2. Evidencing the cost of the U.K. government's proposed regulatory interventions for consumer IoT. Department for Digital, Culture, Media and Sport (2020); <https://bit.ly/3D5Njjk>
3. Garg, V. and Kuehn, A. Squeezing the Cybersecurity Lemons—A Labeling Regime for IoT Products. *login.* (2021); <https://bit.ly/3yvvhQB6>
4. 2016 Norton Cyber Security Insights Report: Understanding cybercrime and the consequences of constant connectivity. Symantec Corporation (2016); <https://bit.ly/300Se4V>
5. Product security: IoT and other Internet-enabled devices. Centre for International Governance Innovation, CIGI-Ipsos Global Survey on Internet Security and Trust (2019); <https://bit.ly/3IAK7Lm>
6. Report for the Assistant to the President for National Security Affairs (APNSA) on Cybersecurity Labeling for Consumers: Internet of Things (IoT) Devices and Software. National Institute of Standards and Technology (2022); <https://bit.ly/3uJRrOS>

Andreas Kuehn (akuehn@orfamerica.org) is Senior Fellow at the Observer Research Foundation America, Washington, D.C., USA.

Copyright held by author.

The Profession of IT

The Atlas Milestone

Celebrating virtual memory, which has made such a difference in how we approach programming, memory management, and secure computing.

VIRTUAL MEMORY IS a technology of computer systems architecture that is as old as academic computer science and has affected the careers of many computing professionals. We take this opportunity to celebrate it as a milestone of computing, recognized by the recent IEEE Milestone award to the University of Manchester, where it was invented in 1958.

First, some background. The IEEE is the world's largest technical society with over 430,000 members in 160 countries. The IEEE Milestones program was established in 1983 to recognize the achievements of giants who advanced the electrical and electronics profession around the world. Each IEEE Milestone is recognized by a bronze plaque mounted at the location of the achievement. The IEEE website lists 224 milestones awarded since 1977, of which 35 milestones are associated with computing.^a

In June 2022 two Milestone plaques were dedicated, one for the “Manchester University ‘Baby’ computer and its Derivatives 1948–1951” and one for the “Atlas Computer and the Invention of Virtual Memory 1957–1962.” An image of the latter plaque appears here.

The Atlas Computer

The Atlas architecture (see Figure 1) incorporated a multitude of what were then novel features: asynchronous pipelined operation, parallel arithmetic,



tic, 128 index registers, double address modification by index registers, extracodes (software sequences simulating additional hardware instructions), interrupts, an interleaved main core store, multiprogramming, and, most importantly, a one-level storage system² that later became known as virtual memory. Virtual memory required novel software and hardware, leading to the creation of an operating system known as the Atlas Supervisor.³ The supervisor also included a compiler for Atlas Autocode, a high-level language similar to Algol 60.

Atlas incorporated multiple kinds of store, including main memory (magnetic core), secondary memory (rotating drum), Fixed Store (precursor to today's firmware holding extra instructions), and V-Store (precursor

to today's memory-mapped peripherals). These different kinds of storage were needed to optimize the different storage-access tasks the CPU had to do. The Atlas Processor (CPU) consisted of the Accumulator (called A-register) and its associated floating-point arithmetic unit, the index registers (called B-registers), and the Control section.

Originally called one-level storage,² the Atlas virtual memory system gave each user the illusion of having a very large main memory by automating the transfer of code and data between a small fast main core store and a large, much slower, magnetic drum. Prior to this, on earlier Manchester machines, programmers spent vast amounts of time augmenting basic algorithms with “overlay sequences”—calls on the secondary memory to transfer

^a See <https://bit.ly/3RFg35c>

pages (standard size data blocks) into the limited main memory. Kilburn believed the one-level storage mechanism would eliminate manual overlays and estimated that programmer productivity would be improved by a factor of up to 3.

The Atlas allowed every program to address up to 1M words via a 20-bit virtual address. However, this created a problem. Kilburn wrote² “In a universal high-speed digital computer it is necessary to have a large-capacity fast-access main store. While more efficient operation of the computer can be achieved by making this store all of one type, this step is scarcely practical for the storage capacities now being considered. For example, on Atlas it is possible to address 10^6 words in the main store. In practice on the first installation at Manchester University a total of 10^5 words are provided, but though it is just technically feasible to make this in one level it is much more economical to provide a core store (16,000 words) and drum (96,000 words) combination.”

There was more to it than just this, however. In previous machines, page transfers took place under direct program control as directed by the programmer. In Atlas the ratio of drum to processor access time would be approximately 2,700:1,^b so to avoid having the processor idle for long periods during page transfers, multiple programs were co-resident in the core store at locations hidden from users. When one program stopped for a page transfer, the processor was switched to another resident program. Kilburn’s solution to these problems was to make user program addresses virtual addresses and to have the computer itself determine the mapping between virtual and real addresses. The system for implementing this concept was a combination of operating system software and hardware known as paging.

It was clear that translating from virtual to real addresses would have to be done in hardware, otherwise there would be a huge time penalty. Also, it would only be feasible to move blocks or pages of information, rather than

individual words, between the drum and core stores. So a set of associative (content addressable) registers, the Page Address Registers, was used (see Figure 2). A PAR held the page number of the page loaded in the associated page-frame of memory. The lock-out bit was set for PARs containing pages of suspended jobs. With the chosen page size of 512 words, the 16K words of core store spanned 32 pages, so 32 PARs were needed. The 20-bit virtual address was therefore split into 11 bits of page address and 9 bits of line address. The page address was presented to all the Page Address Registers simultaneously and in most cases one of them would indicate a match with

the presented address. The outputs of the PARs were then encoded to form a 5-bit page-frame address which was concatenated with the 9-bit line address to form the real address to be sent to the core store. All this address mapping was done in a small fraction of a memory cycle and was invisible to programmers.

If there was no PAR match, a page-fault interrupt was generated. The page-fault handler of the operating system intervened, found the missing page on the drum, moved it into a blank page-frame of the main store, and updated that frame’s PAR with the page number it now contained. When all this was done, the operating

Figure 1. The ATLAS architecture.

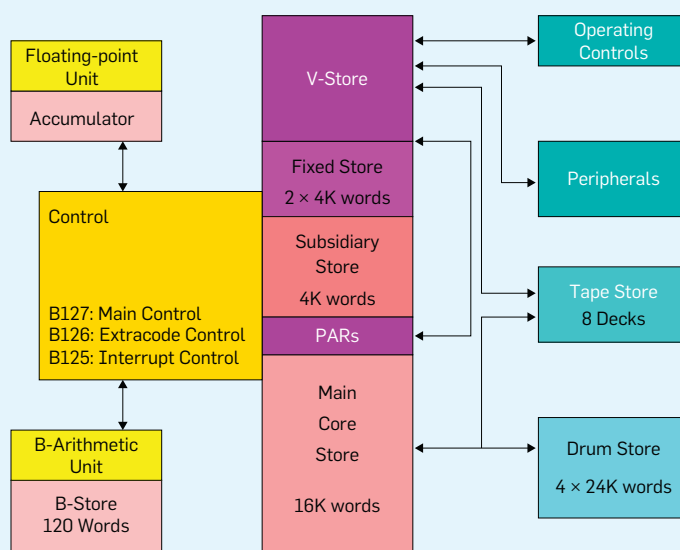
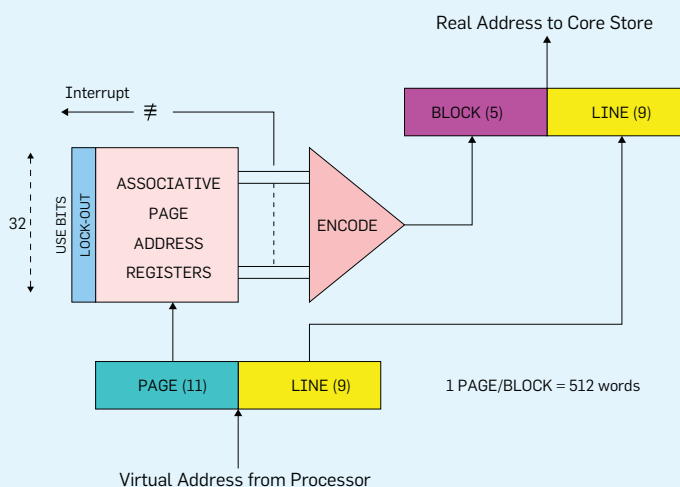


Figure 2. Page address registers.



^b In today’s virtual memories, this ratio is much worse, closer to $10^6:1$

Overcoming the Inherent Challenges in Creating a Revolutionary Academic Program

Should Young Computer Scientists Stop Collaborating with Their Doctoral Advisors?

Linear Address Spaces

The Dangers of Participation Bias in Educational Studies

Assessing the Quantum Computing Landscape

Storytelling and Science

AuraRing: Precise Electromagnetic Finger Tracking

Traffic Classification in an Increasingly Encrypted Web

Plus, the latest news about neuro-symbolic logic, finding hidden malware, and how applied AI teaches writing skills.

The Atlas one-level store was hailed as a major breakthrough in computer systems architecture and was quickly taken up by engineers building other systems.

system later resumed the interrupted program, which could now continue because its last memory access would now map to main memory. In later virtual memories, not enough PARs could be provided to cover the whole of main memory; the PAR array was replaced with a translation lookaside buffer and a page table.

Now there is one other problem to deal with: maintaining a blank page-frame in memory so that the next page fault had a frame available to receive the missing page. This was done by a replacement policy called the “learning algorithm.” As part of processing a page fault, the operating system would use the learning algorithm to select one of the other 31 pages for replacement and initiate a swap to copy that page back to the drum. The learning algorithm was the world’s first replacement policy.

The learning algorithm assumed all pages of a program were involved in loops. By monitoring use bits, it measured intervals of use and non-use and calculated a period for each page’s loop. It then selected for replacement the page that would not be reused for the longest time into the future. This principle, known today as the “MIN principle,” is optimal if indeed all pages are in fixed loops. This assumption is not always met and caused performance problems in virtual memories built after 1962. We will discuss this next.

Performance of Virtual Memory

The Atlas one-level store was hailed as a major breakthrough in computer systems architecture and was

quickly taken up by engineers building other systems. These systems soon encountered two significant performance problems. One was that the best replacement algorithms tended to require heavy overhead—the Atlas “learning algorithm” was of this kind—and the low overhead ones caused too much paging. The other problem was thrashing, an unexpected collapse of throughput in a multiprogrammed system when the number of loaded jobs exceeded an unpredictable threshold. These issues put the entire project for virtual memory under a cloud. What good was a multimillion-dollar computer that was bogged down with paging and whose performance is likely to collapse unpredictably?

In a classic study (1966), Les Belady of IBM put a large number of possible replacement algorithms to the test under a variety of workloads. He concluded the near-zero-overhead FIFO (first in first out) policy generated more paging than most of the others, and that the high-overhead LRU (least recently used) generally outperformed most of the others. He also tested an optimal algorithm, MIN, which gave the least possible amount of paging but was not real-time implementable because it required knowledge of the future. He was disappointed that most of the policies including LRU were significantly poorer performers than MIN. There seemed to be no hope that a paging policy with near optimal performance was feasible.

When these basic algorithms were extended to multiprogramming, the operating system needed to assign a memory region to each job—for example, N jobs would each get $1/N$ of the memory. If N got too high, all the jobs would be pushed into a state of high paging, which meant every job was unable to use the CPU very much and overall throughput collapsed as the jobs “paged to death.” There was no way to determine where the threshold N was because it depended on the details of each job.

A breakthrough came in 1966 with the concept of working set.¹ A working set is the intrinsic memory demand of a program—the set of pages that if resident would generate a very low level of paging. The working set was

measured by observing which pages a job accessed in a backward looking window of the last T memory accesses. The working set policy for multi-programming gave each job enough pages for its working set. This meant a small portion of memory, called FREE, was unused. At a page fault a working set would increase by one page, taking it from FREE. When a page was no longer in a job's T -window, it would be evicted and returned to FREE. In this way, no page fault could steal a page from another working set and thereby interfere with its performance. The scheduler would not load a new job if its working set was bigger than FREE. It was impossible for a working set policy to thrash.

The final piece of the performance puzzle—optimal throughput—was provided by the principle of locality. This principle holds that programs access small subsets of their address spaces over relatively long phases, and the entire execution of a program can be described as a series of phases where in each one a locality set of pages was used continuously. If the operating system could detect locality sets and load them, most jobs would generate almost no paging during phases. Most of the paging was caused by the relatively infrequent phase transitions to new locality sets. It is not difficult to prove that such a policy is near-optimal—no other policy, including those with lookahead, can generate significantly higher throughput.¹

The working set policy does just this because the working set measurement sees exactly the locality set when its T -window is contained in a phase. The same locality sets and phases are observed over a wide range of T -values. Programs with locality managed by working sets typically operate within 1%–3% of optimal.

It is now easy to see that virtual memory and working-set management are a perfect team to attain best possible, thrashing-free performance from a virtual memory.

The Secure Kernel Problem

Some people believe virtual memory has become obsolete because memory has become so cheap we can allocate all the real memory a job needs. There

Some people believe virtual memory has become obsolete because memory has become so cheap we can allocate all the real memory a job needs.

is then no paging and the mechanism becomes superfluous.

While it may be true that most jobs can be fully loaded into main memory, that hardly spells the demise of virtual memory. There are always jobs that are just too big for the available memory. Virtual memory makes it possible to run such jobs.

Even more important, however, is that the address mapping of virtual memory guarantees complete isolation of jobs. A job can access only the page frames linked to its page table, and no frame can be shared between two jobs. Therefore, no job can access the memory held by another job. This default isolation is the basis for security kernels in operating systems. Even if there is no need for an automatic solution to the overlay problem, virtual address mapping provides the logical partitioning that is the basis for secure computing.

What a legacy for Kilburn's invention. ■

References

1. Denning, P.J. Working set analytics. *ACM Computing Surveys* (Jan. 2021).
2. Kilburn, T. et al. One-level storage system. *IRE Trans. EC-11* (Apr. 1962).
3. Kilburn, T. et al. *The Atlas Supervisor*. *AFIPS Proc. European Joint Computer Conference (EJCC)* (Dec. 1961).

Peter J. Denning (pjd@nps.edu) is Distinguished Professor of Computer Science at the Naval Postgraduate School in Monterey, CA, is Editor of *ACM Ubiquity*, and is a past president of ACM. His most recent book is *Computational Thinking* (with Matti Tedre, MIT Press, 2019).

Roland Ibbett (roland.ibbett@bcs.org.uk) is Emeritus Professor of Computer Science at the University of Edinburgh; he was previously Reader in Computer Science at the University of Manchester, where he was a major contributor to the MU5 project.

Copyright held by authors.



Advertise with ACM!

Reach the innovators and thought leaders working at the cutting edge of computing and information technology through ACM's magazines, websites and newsletters.



Request a media kit with specifications and pricing:

Ilia Rodriguez
+1 212-626-0686
acmm mediasales@acm.org



Point/Counterpoint On the Model of Computation

DOI:10.1145/3548783

Point: We Must Extend Our Model of Computation to Account for Cost and Location

William Dally

FOR DECADES WE have used the RAM (random-access memory)² and PRAM (parallel RAM) models⁵ along with asymptotic analysis to measure the complexity of algorithms. The RAM and PRAM models treat all operations, from an integer add to a global memory load as unit cost. This approximation was appropriate during the early days of computing when the costs of arithmetic and communication were somewhat comparable. Over time, however advancing semiconductor technology has caused the cost of arithmetic and logic to shrink by orders of magnitude while the cost of communication has reduced at a much slower rate. As a result, today fetching two 32-bit words from main memory expends 1.3nJ of energy while performing a 32-bit add operation (which requires two 32-bit words as input) takes only 20fJ of energy 64,000x less. A model of computation like RAM or PRAM that treats these two operations as equivalent does a poor job of estimating the cost of a computation, and hence does a poor job of comparing alternative algorithms.

Communication Is the Dominant Cost in Computing. Whether we consider cost to be energy or time, commu-



nication dominates modern computation. A 32-bit add operation takes only 20fJ and 150ps. Moving the two 32-bit words to feed this operation 1mm takes 1.9pJ and 400ps. Moving the 64 bits 40mm from corner to corner on a 400mm² chip takes 77pJ and 16ns. Going off chip takes 320pJ with a delay of 6ns per meter.

While memory accesses are costly operations, almost all of the cost of accessing memory is communication cost. The time and energy needed to read or write a single bit cell is negligible. The vast majority of the time and

energy of a memory access is due to communication: moving the address to the bit cell and moving the data from the bit cell to the consumer. Because communication dominates, accessing a small, local memory is far less expensive than a global memory access. On-chip SRAM memories, for example, are built from small 8KB (64Kb) sub-arrays. Accessing 64b from a sub-array costs 0.64pJ and takes 300ps. Accessing a 256MB memory (approximately 100mm²) constructed from these sub-arrays costs 58pJ and 12.3ns—of which 57.4pJ and 12ns are due to communi-

cation—15mm each way, 30mm total. Because communication completely dominates the cost of computation, our model of computation must consider communication, and to do so, it must have a model of location.

Consider the problem of summing a table that fits in the on-chip memory of a 4x4 array of 256-core processor chips that are each 16mm on a side. Each core is located at the center of a 2MB SRAM array and can access the local array with 1mm (round trip) of communication cost (1.9pJ and 400ps for a 64b access). A random access incurs 21.3mm of on-chip communication cost for the 1/16 of accesses to the local chip and an additional 640pJ + 21.3mm of cost for the 15/16 of accesses that go off chip for an average random access energy of 680pJ. With a model of location, we can have each core sum the 256K words in its local memory and then forward the result up a tree to compute the final sum. We can perform the same computation by placing the threads and data randomly. The cost of the randomly placed computation is 354x higher than the local computation. The PRAM model considers the local and random computations to be of equal cost despite the orders of magnitude difference in cost.

Store or Recompute. The large difference in energy between arithmetic and communication often drives the decision whether to store or recompute an intermediate value. Training a multi-layer perceptron (MLP) requires the activation values for each layer during the back-propagation step. If there is insufficient on-chip memory, the activations can be stored to off-chip memory at a cost of 640pJ per 16b activation (write + read), an $O(n)$ operation—where n is the number of activations in this layer. Alternatively the activations can be recomputed on the fly by performing a matrix-vector multiplication, an $O(n^2)$ computation, at a cost of 160fJ/MAC. The PRAM model would suggest storing the intermediate result—preferring the $O(n)$ store over the $O(n^2)$ $M \times V$. However, for vectors smaller than $n = 4,000$, recomputation is less expensive. For a typical MLP with $n = 256$, recomputation is 16x less expensive.

Constant Factors Matter. Under the PRAM model, the asymptotic complexity of the summation example here is

With the end of Moore's Law, domain-specific architectures are emerging as a leading candidate to continue scaling computing performance.

$O(n)$ for both the local and the random cases. The 230x difference in energy is just a constant factor, as is the 64,000x difference in cost between an add operation and a global memory access.

Constant factors do matter, however, particularly when the asymptotic complexity is the same. The local computation is 230x cheaper. Just as you would not settle for paying 230x too much for your groceries, you should not pay 230x too much for a computation because you are using an incomplete model of computation. Large constant factors can even overcome a difference in asymptotic complexity (this is why Strassen's $O(n^{2.8})$ matrix multiply algorithm⁶ is rarely used). This is the case in our store $O(n)$ vs. recompute $O(n^2)$ example here, where for typical values the higher asymptotic complexity is less expensive.

Caches Are Not Enough. Modern computers use cache memories to improve locality. Caches work great in cases where there is temporal locality (that is, reuse). However, for computations such as the summation example here they do not help. Every word is visited exactly once, there is no reuse. Many HPC codes and neural-network models with a batch size of one have the same issue.

Domain-Specific Architectures Highlight the Issue. With the end of Moore's Law, domain-specific architectures (DSAs) are emerging as a leading candidate to continue scaling computing performance.^{3,4} By removing most of the overhead associ-

ated with general-purpose CPUs, DSAs make the large gap between arithmetic and communication more apparent. The overhead of a CPU turns the 20fJ of a 32b add operation into an 80pJ add instruction. In comparison the 1.2nJ memory access is only 15x more expensive rather than 64,000x. With a DSA this overhead is eliminated revealing the full size of the gap between arithmetic and communication.

Computational Models are for Performance Programming. In some situations, such as user-interface code, programmers concentrate on functionality without concern for cost. In these situations a naive code is *fast enough* and there is no need for a model of computation (RAM, PRAM, or other) to estimate cost and compare alternatives. In other cases, however, such as training neural network models, performing large scientific simulations, and running optimizations, cost is of paramount importance. Huge amounts of energy and expensive computation time are wasted by an inefficient algorithm. It is in these cases of *performance programming* where we need a model of computation to estimate cost and compare alternatives, and that model of computation needs to reflect the communication-dominated nature of today's computing hardware.

The Planet Demands We Be More Efficient. Datacenter computers alone used 1% of global electricity in 2018, and estimates indicate this number will grow to 3%–13% by 2030.¹ With such a large fraction of the world's energy going to computing, it is incumbent on us to design efficient computations to minimize the carbon footprint of computation. The first step in being more efficient is having an accurate model to analyze algorithms, so we can pick the most efficient algorithm for a given problem. We can no longer afford to ignore the constant factor difference between arithmetic and communication or the increase in energy with distance.

PECM: A Simple, Accurate Model of Computation. Two simple changes to the PRAM model can fix its two main problems. To account for the large difference in cost between arithmetic operations (like add) and memory access, we assign them different costs. Arith-

metic operations remain unit cost, but memory operations have a higher cost, proportional to distance. Each processing element and each memory is assigned a location $l \in L$ and a distance function $D: L \times L \Rightarrow R$ is defined to determine the cost of sending a message or making a memory access between two locations.

We can use different distance functions to model different computational targets. Two-dimensional Manhattan distance models on-chip communication. Locations are x, y coordinates and distance is the Manhattan distance be-

tween two coordinates. Off-chip communication can be modeled by adding additional distance when either coordinate spans a chip boundary (as in the example here).

By allowing us to reason about the true costs of computations, this parallel explicit communication model (PECM) will allow us to design more efficient computations and in doing so reduce the carbon footprint of computing. **C**

References

1. Anders, A. and Edler, T. On global electricity usage of communication technology trends to 2030. *Challenges* 6, 1 (2015), 117–157.

2. Cook, S.A. and Reckhow, R.A. Time-bounded random access machines. *Journal of Computer Systems Science* 7, 4 (Apr. 1973), 354–375.
3. Dally, W.J., Yatish, T., and Han, S. Domain-specific hardware accelerators. *Commun. ACM* 63, 7 (July 2020), 48–57.
4. Hennessy, J.L. and Patterson, D.A. A new golden age for computer architecture. *Commun. ACM* 62, 2 (Feb. 2019), 48–60.
5. Karpand, R.M. et al. A survey of parallel algorithms for shared-memory machines. In *Handbook of Theoretical Computer Science*. North-Holland, 1988.
6. Strassen, V. Gaussian elimination is not optimal. *Numer. Math.* 13, 4 (Apr. 1969), 354–356.

William Dally (dally@stanford.edu) is an adjunct professor of computer science at Stanford University and chief scientist and senior vice president of research at NVIDIA, Incline Village, NV, USA.

Copyright held by author.

DOI:10.1145/3548784

Counterpoint: Parallel Programming Wall and Multicore Software Spiral: Denial Hence Crisis

Uzi Vishkin

BACKGROUND. THE SOFTWARE SPIRAL (SWS) for single CPU cores and the RAM algorithmic model. Andy Grove (Intel’s business leader until 2004) termed “software spiral” the exceptionally resilient business model behind general-purpose CPUs. *Application software* is the defining component of SWS: Code written once could yet benefit from performance scaling of later CPU generations. SWS is comprised of several abstraction levels. The random access machine, or model (RAM) is most relevant for the current Counterpoint Viewpoint (CPV): each serial step of an algorithm features a basic operation taking unit time (“uniform cost” criterion). The RAM has long been the gold standard for algorithms and data structures. Salient aspects of the RAM included: its simplicity; importing from mathematics its own gold standard: mathematical induction for describing algorithms (or their imperative programming code) and proving their correctness; and good enough support by computer systems based on the von Neumann architecture. Other abstraction levels and means included a variety of benchmark suits guiding balanced performance over a range of tasks, software compatibil-

ity, object or binary code compatibility, operating systems, and a variety of standards, for example, Figure 1.8 on functional requirements in Hennessy and Patterson.⁵ The single core CPU business became synonym with making every effort to advancing architectures and optimizing compilers for keeping this SWS on track, making it, as well as the RAM, so resilient, and clear role models for the road ahead.

There Is More to a Lead Model of Computation Than Specialized Efficiencies. The Point Viewpoint (PV) makes a strong case for optimizations based on quantifiable costs at the hardware level. This CPV concurs with applying the PECM model of computation the PV proposes to specialized routines whose use in workloads merits it, as well as to accelerators. However, the foremost problem of today’s multicore parallelism is dearth of programmers, since programming such systems is simply too difficult. Imposing the PECM implied optimizations on programmers is unlikely to bring programmers back, thus qualifying its applicability. I am also not aware of demonstrated success of PECM for general-purpose programming. Using computer architecture lingo, the upshot of the current paragraph is that architects of manycore platforms must recognize not only the so-called “memory wall” and “energy wall,” but also a “parallel programming wall.”

A Broader Perspective. This CPV demonstrates how to approach the debate on a computation model using a different premise. Learn from the traditional business model of general-

purpose single core CPUs, the aforementioned SWS. SWS enabled the remarkable success of such CPUs. These CPUs are arguably the biggest success story of the founders’ generation of the whole information technology sector. SWS provided superb sustainability and resilience. It allowed CPU applications to grow from a handful of scientific applications to today’s ubiquity, as can be seen in desktop, laptop, server and smartphone apps. However, current exploitation of parallelism for general-purpose application performance falls far behind the exploitation of performance of single core CPUs. It is time to recognize the source of this crisis: after nearly two decades of multicore CPUs domination, SWS stagnated, failing to extend to general-purpose multicore CPUs. CPU vendors have simply gone AWOL on this matter. Lacking cost-effective means to exploit parallelism is at the heart of the problem: most application programmers and their employers simply stay away from even trying to program for parallelism today’s multicores, or domain-specific-driven accelerators such as GPUs. This CPV aspires to rectify this failure. Namely: seek to instate a multicore SWS (MSWS) for general-purpose CPUs.

Cost-Effective Programming Is Critical Even for Performance Programming. A product must accommodate its customers. Multicore CPUs are no different. An MSWS will need to appeal to a wide range of programmers, whose software will then make it appealing to application users at large. For sustainability and resilience, a new

MSWS will need to provide: cost-effective programming and as an MSWS takes hold, backward compatibility on prior MSWS application software. These features are necessary to drive investment in new and current applications. The RAM model of computation played a leading role in enabling easy programming for performance for serial computing, so it is only natural to turn to the PRAM model, its parallel counterpart for such programming. In comparing the PRAM to PECM, the primacy of cost-effectiveness appeal to diverse programmers will tilt the scales toward the PRAM. The PECM over-reliance on specialized efficiencies will make current systems even more brittle, exacerbating current predicaments rather than curing them. This CPV will explain how the PRAM coupled with properly designed architecture and optimizing compilers could sustain backward compatibility in a future MSWS. For instance, the type of optimization for locality used for matrix multiplication on GPUs, which is currently associated with unavoidable performance algorithms and programming, would instead be characterized as “compiler algorithms.” The point being that with few exceptions application programmers must not be responsible for such optimizations for a manycore to reach broad use. Instead, once developed these optimizations as well as PECM-type optimizations belong in a compiler.

Multicore Software Spiral: Industry Aspiration and Current Crisis. Parallelism has become the dominant source for performance improvement for application developers and innovators, circa 2005, once CPU clock rate acceleration started slowing down. But, what happened to SWS? In a 2008 interview,³ Patrick Gelsinger (then CTO of Intel Corporation and its current CEO) was asked whether the SWS concept still holds, with multicore chips already available but without a lot of SW that can run on more than one processor at a time. He answered that he sees the move to multicore as a new turn in the spiral. Unfortunately, the premise of the question remains as valid today: not much SW exploits parallel processing on multicores. Also, SW written for yesterday’s multicore may need to be performance tuned for

The foremost problem of today's multicore parallelism is dearth of programmers, since programming such systems is simply too difficult.

today’s multicore. Thus, the turn in the spiral is yet to happen. This widening MSWS crisis is at the very core of computer science and engineering. It must be widely recognized and addressed. Multicores are on every desk and laptop. Yet, the programming and business barriers for multicore software utilization preempt bringing enough programmers on board.

The Original Rationale for PRAM.

Since the transition to multicore CPUs, the question remains: which computational model should replace RAM as the lead model? The PV proposes the parallel explicit communication model (PECM), though without reference to MSWS. In contrast, this CPV suggests that the PRAM model leads overcoming the MSWS crisis. In the PRAM abstraction, each serial step of an algorithm comprises an unlimited number of basic operations, all are assumed to execute concurrently in unit time. Realizing that serial algorithm knowledge coupled with compilers alone will not be sufficient for effective use of parallel processing, the original motivation for the PRAM at the time was to address that while retaining as much of the simplicity of the RAM as possible. The idea was that programmers express all the parallelism they see, but nothing else must be added to the RAM. The synchronous lockstep abstraction retained also tight reliance on mathematical induction. In contrast, many threaded models of parallelism abandoned the latter, leading⁷ to his stern warning concerning their utility.

PRAM Has What It Takes to Over-

come the MSWS Crisis. Envisioning a manycore CPU on chip within a decade from the late 1990s, a vertically integrated computer framework, called XMT or PRAM-On-Chip,¹¹ was developed and thoroughly prototyped at the University of Maryland (UMD). Prior well-cited papers such as Culler² claimed impossibility of ever supporting the PRAM due to technology limitations. XMT aimed to directly refute them by providing effective support to the PRAM model. XMT started with enhancing serial and parallel hardware designs with several low overhead hardware primitives and system software. The main effort was to bridge the considerable tension between: the need for multithreaded operation of a fixed number of hardware contexts, limiting synchrony among them for limiting power and for taking advantages of locality; and the PRAM ideal (or illusion) comprising lock-step unit time access to shared memory, including highly irregular pointer-based access, and (possibly drastically) changing amounts of parallelism from step to step. The XMT solution included hardware supported prefix-sum and dynamic load balancing, high bandwidth low latency on-chip interconnection networks, integrated parallel and serial memories and serial/parallel mode transitions, as well as refined notions of locality- and predictability-of-reference,^{9,12} dodging the need for system enforced cache coherence, scalable architecture (toward future backward compatibility of the new MSWS) and gradual development of optimizing compiler. XMT incorporated a variety of prefetching techniques as well as loop unrolling, relying on both hardware and software methods to hide latencies and improve locality. Each compiler “generation” required manual tuning of PRAM algorithm followed by having the tuning automated in the next generation of the compiler. Optimizing a measure called LSRTM, for the length of the sequence of round trips to memory, has been a prime target of the design. Hardware (commitment to silicon using FPGA) and software prototyping demonstrated competitive speedups over contemporary off-the-shelf multicores and GPUs. In view of the PV, the power consumption demonstration Keceli et al.⁶ is specifically noted. The main target for competitive speedups were irregular problems, where

memory access is data dependent and cannot be fully predicted at runtime. (This is in contrast to when memory access is data independent, as in standard matrix multiplication, where optimizing a parallel algorithm per the PECM model is more likely to make a difference, in line with the PV.) The XMT multidecade effort culminated in Ghanim et al.,⁴ which demonstrated supporting PRAM algorithms (and the PRAM illusion, or virtual model) as-is with no additional programming effort. It is also worth noting that recent applications of locality sensitive hashing for performance improvements of some salient deep learning tasks on multicore CPUs over GPUs¹ suggest a greater arsenal of potential techniques than pursued by XMT for general-purpose multicore CPUs.

Ease of Programming Superlatives.

1. *The PRAM parallel algorithmic gravitas.* The PRAM model has also been challenged for around four decades by numerous competing parallel algorithmic models. Still, none of its competitors ended up reaching the magnitude and breadth of its parallel algorithmic knowledge base and techniques, on which the PRAM is second only to the serial RAM model. 2. *The XMT/PRAM broadening of access to parallel programming education.* First, an example directly benefitting from PRAM support. Nearly a thousand students at a single high school, the Thomas Jefferson High School, Alexandria, VA, programmed XMT over the last 12 years, freeing them to pursue problems beyond “embarrassingly parallel” ones in contrast to the programming of off-the-shelf systems.⁸ Cost-effective broadening of parallel programming to more people including students from middle school to graduate school as well as for more challenging problem have been also demonstrated. Such broadening of participation can help overcoming the MSWS crisis once industry-grade PRAM-supporting platforms become available.

A Lesson from the Commercial Success of GPUs. Prior to the rise of deep learning, the performance of GPUs for multiplication of full matrices was already ahead of much of the competition. The serendipitous application of matrix multiplication to deep learning (via a stochastic gradient descent primitive for backpropagation of neural nets) changed the fortune of GPUs,


A computation model may be less negotiable than a physical technology one. It is time for vendors to come to terms with that and adjust.

leading to market caps of GPU vendors exceeding CPU ones. If our generation of IT professionals will finally deliver the needed MSWS, such parallel platform will immediately become an open invitation to an unlimited range of applications, many are yet to be invented. The outcome could far exceed the impact of deep learning on GPUs, continuing the exponential SWS trajectory that flattened once the MSWS crisis commenced approximately 2005.

Exceptionalism of Computation Models. It is intuitive to regard physical technology modeling as non-negotiable as the PV does, but treat computational model as fully negotiable manmade artifacts. Indeed, over the four decades of my involvement in parallel computing research claims that the RAM and PRAM models are “unrealistic” coupled with “more accurate” alternative models flourished in papers, producing, in my estimate, many more publications than strictly PRAM ones. These alternative models offered publication opportunity to publication hungry academics specifically because their modeling forced complex solutions even for simple problem, supporting the type of “depth” and “originality” claims needed to merit publications. However, these models ran into applicability challenges once significant applications and more complex problems had to be solved. Still, 40 years later, the PRAM remains the primary target for similar attacks. I believe withstanding so many attacks over so many years provides powerful attestation to unique robustness. This may support a perhaps surprising conclu-

sion. Namely, that a computation model may be less negotiable than a physical technology one. After 40 years, the PRAM with (the centuries-old) mathematical induction by its side, remain unchallenged. It is time for architects and vendors to come to terms with that, as the term parallel programming wall introduced earlier in this CPV suggests, and adjust.

MSWS Merits Government Funding.

The U.S government is in the process of awarding \$52 billion to support U.S.-based semiconductor research and production. A small fraction of such funding level should allow kickstarting MSWS. Even targeting industry-grade hardware and software would be much less expensive. It will also be a better deal. The resulting gold mine of application opportunities could prompt vendors and investors to then also fund semiconductors on their own. Overall, a better outcome for the U.S. economy with lower cost to government. 

References

- Chen, B. et al. SLIDE: In defense of smart algorithms over hardware acceleration for large scale deep learning systems. In *Proceedings of the Conference on Machine Learning and Systems (MLSys)* (2020).
- Culler, D. et al. LogP: Towards a Realistic Model of Parallel Computation. In *Proceedings of the Fourth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 1993, 1–12.
- Gelsinger, P. Interview. *Network World* (2008); <https://bit.ly/3AOsy8N>
- Ghanim, F., Barua, R., and Vishkin, U. Easy PRAM-based high-performance parallel programming with ICE. *IEEE Transactions on Parallel and Distributed Systems* 29:2, 2018
- Hennessy, J.L. and Patterson, D.A. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, 2019.
- Keceli, F., Moreshet, T. and Vishkin, U. Power-performance comparison of single-task driven many-cores. In *Proceedings of the 17th IEEE International Conference on Parallel and Distributed Systems (IEEE ICPADS)*, (Tainan, Taiwan, Dec. 7–9, 2011).
- Lee, E.A. The problem with threads. *IEEE Computer* 39, 5 (May 2006), 33–42.
- Torbert, S. Is teaching parallel algorithmic thinking to high-school students possible? One teacher's experience. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education (SIGCSE)*, (Milwaukee, WI, Mar. 10–13, 2010).
- Vishkin, U. Can parallel algorithms enhance serial implementation? *Commun. ACM* 39, 9 (Sept. 1996), 88–91.
- Vishkin, U. Algorithmic approach to designing an easy-to-program system: Can it lead to a HW-enhanced programmer's workflow add-on? In *Proceedings of the International Conference on Computer Design (ICCD)*, Lake Tahoe, CA, October 4-7, 2009.
- Vishkin, U. Using simple abstraction to reinvent computing for parallelism. *Commun* 57, 1 (Jan. 2011), 75–85.
- Vishkin, U. Is multicore hardware for general-purpose parallel processing broken? *Commun* 57, 4 (Apr. 2014), 35–39.

Uzi Vishkin (vishkin@umiacs.umd.edu) is a professor at the University of Maryland Institute for Advanced Computer Studies (UMIACS) and Electrical and Computer Engineering Department, A. James Clark School of Engineering, College Park, MD, USA.

Copyright held by author.

Viewpoint

Let Us Not Put All Our Eggs in One Basket

Toward new research directions in computer science.

OUR COLLEAGUES AT the Intergovernmental Panel on Climate Change (IPCC^a) and the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES^b) have been telling us for years the situation is serious. Last year saw both the publication of the sixth IPCC report, and dramatic illustrations of the impacts of climate change. Researchers and teachers in all disciplines face the question: What can you do in your professional life? If you search the Internet for occurrences of “carbon-neutral university,” you will find a long list of declarations by universities worldwide, claiming they will be carbon neutral by 2030 or 2040. I will not discuss here whether carbon-neutrality objectives are feasible or even make sense at all (see Dyke³). I take this series of declarations as a symptom that the academic world is hopefully starting to take scientific results seriously, at least concerning the impact of our work organizations.

In computer science, several personalities have started questioning our peculiar organization that gives an important role to conferences,⁸ advocating for a massive change in how research is made and disseminated. Funders also have a significant impact.² As far as I am concerned, I stopped airline travel completely, and that is the least I can do, having done



quite a lot in the past 30 years. But when I ask myself “what should I do?”, when my students ask “are we part of the solution, or part of the problem?”, I also look at my research and teaching topics, and I feel compelled to question the contributions of these topics to the development and impacts of the digital world as a whole. It is tempting to look at the positive impacts only. The public discourses tend to present the “digital transition” as a necessary and non-questionable solution to the needed “ecological transition.” Our research

community has the responsibility to consider several hypotheses, including one in which the digital world is part of the problem.

A Tale of Three Futures

Let me tell you a short tale meant to let our imagination escape the very pregnant determinism of tech discourses, at least for a few minutes. In 2005, I had a very simple mobile phone allowing me to place and receive calls (almost) everywhere, and which needed to be charged once a week. Telephone

a See <https://www.ipcc.ch>

b See <https://ipbes.ne>



Peer-reviewed Resources for Engaging Students

EngageCSEdu provides faculty-contributed, peer-reviewed course materials (Open Educational Resources) for all levels of introductory computer science instruction.



engage-csedu.org



Association for
Computing Machinery

booths were still available in urban or rural areas. I am now one of at least one billion people carrying an always-connected always-on portable computer in our pocket, and if we really use all of its functions, we need to charge it twice a day. Telephone booths have disappeared completely. Cafes all over the world advertise the availability of electric plugs and free Wi-Fi to attract a crowd of connection-hungry customers. You can charge your phone by practicing on a static bike while waiting at airports, and you can carry a solar panel on your backpack for a two-day hike. GPS and maps are an example of functions already available on dedicated devices prior to smartphones that have migrated to smartphones thanks to the versatility of this type of platform. Entirely new functions have appeared thanks to 24/7 connectivity—for instance, platforms such as Uber.

What happened between 2005 and 2021? There is absolutely no doubt that huge progress has been made on several key points: the technology of batteries has improved; the hardware architecture and the operating systems have been enriched with sophisticated mechanisms to optimize energy consumption; the capacity of memories has increased; new underwater cables and optical fibers have been installed, 4G and 5G have been deployed; and significant other improvements. But what about the overall environmental impacts of this growing infrastructure and the huge number of short-lived devices connected to it, or the indirect impacts on other sectors?

Let us imagine for a moment that we are back in 2005, doing our job of computer scientists, optimizing hardware and software. What futures did we envision? Future 1, in which our simple phones, functionally unchanged, would need to be charged once a month only, thanks to the improvements of batteries, software and hardware? Or Future 2, in which the one-week charging period would be preserved, and as many new functions packed in the device as made possible by those improvements? Could we have imagined Future 3, that is, what we have now? The huge improvements of all aspects of the digital world have been accompanied by massive rebound effects,⁷ both direct and indirect. The fact that

Futures 1 and 2 were very unlikely to emerge because there would have been no economic incentives for such massive improvements, without an expected market increase, that is, a bet on the rebound effects, makes the path followed between 2005 and 2021 a quite slippery slope. This phenomenon cannot be explained by technological arguments only. When we are working on optimizations of digital systems now, are we not in the position we were in 15 years ago, believing we were working for Futures 1 or 2, but allowing Future 3 instead?

Should We Try to Avoid a New Future 3 and if Yes, How?

Evaluating the total environmental impacts of the digital world is a complex task. According to the meta study,⁴ the greenhouse gases emissions of the digital world account for 1.8% to 3.9% of total emissions and are likely to increase. Arguably, compensating those impacts by corresponding cuts in the emissions of other—non-digital—sectors, would require such profound and quick transformations that it might not be feasible.

The moral of the story, put in a provocative form, could be: If there is a single example in the history of computing, where a particular optimization has not been accompanied by massive direct and indirect rebound effects, then we should study it extensively, from various points of view: technological, economical, sociological, and so forth, in order to try and reproduce it. If there is no such example, then we should stop believing that optimizations always help reducing environmental impacts.

When we start thinking of what it would take to avoid rebound effects and keep the impacts of the digital world within certain limits, at least two types of arguments are common: individual ethics and self-limitations, or regulations designed collectively. Both imply choices and priorities.

I personally think that, in front of climate change dramatic consequences, 8K videos, connected refrigerators, cloud-dependent home automation, cashierless retail stores, autonomous vehicles, smart shoes, the metaverse, Web3, and NFTs are at best helpless and misdirected innovations, at worst

and most probably, harmful. Other technologies, such as high-tech medicine, may be useful, but concern the happy few only.

Whatever our personal opinion, as computer scientists we can start exploring the notion of limits even if we do not agree on the moral judgments related to the choice of those limits. We can even explore the notion of limits without being convinced there should be limits in the first place, just because this is a fascinating territory of undone science.⁵ How to stay within limits has become a scientific and technical problem that is little addressed.

Toward New Research Directions in CS: Limits as First-Class Citizens

Aside green-IT, which deals with optimizations of digital systems, and green-by-IT, in which IT is used to reduce the impact of some non-digital sectors, avoiding the slippery slope of future 3 requires that we also work on an entirely new topic: limited-by-construction IT. The recently created series of conferences LIMITS^c or the notion of Collapse Informatics,⁶ advocate for a digital world that deals with planetary limits, or may survive collapse scenarios.

When it comes to designing and developing computer systems, thinking in terms of limits requires a paradigm shift. We can start by highlighting the implicit anti-limits most of the digital systems of our everyday life are based on. An anti-limit is both a promise and a deliberate hypothesis that resources will grow as needed. For instance, there are obvious anti-limits if a digital system:

- ▶ Requires an increasing amount of resource globally (unlimited number of cryptocurrencies relying on proof-of-work, space, or bandwidth, ...);
- ▶ Promises immediate service delivery, whatever the number of clients and usages (most of the cloud services);
- ▶ Promises unlimited storage in both space and time (Gmail);
- ▶ Assumes availability of some hardware, software, and vendor cloud forever (some home automation devices);
- ▶ Is designed to allow for unlimited functional extensions;

c See Computing within Limits workshop series: <https://bit.ly/3IGWANv>

Our discipline may need a radical approach, redesigning the digital world from scratch with specifications based on explicit hardware and software limits.

▶ Bets on the availability of a more efficient machine, soon; and

▶ Needs more users or an increased usage per user to be profitable

Most of these examples are clearly rooted in economical choices, but thinking without limits has become so tightly knitted with the very principles of technical solutions, that in some cases it could be difficult to continue delivering solutions, should environmental, (geo-) political or social constraints impose restrictions on the development of the digital world.

So what can we do? Having spent most of my 30-year career working on critical embedded real-time systems, I am used to languages and tools meant to determine the worst-case execution time, and the amount of memory needed by a program, before deployment: limits are part of the specification, and a very stringent constraint for the implementation. Other sources of inspiration include Gemini,^d which is designed to be difficult to extend in the future, while in any software engineering course, “extensibility” is presented as a desirable property. According to Wikipedia, it is “a software engineering and systems design principle that provides for future growth.” Designing systems that are not scalable, on purpose, is one way to keep limits in mind. Designing for intermittent resources or user quotas is another: A solar-powered website, which means it sometimes goes offline, is presented in Abbing.¹ The ultimate

d See Project Gemini: <https://bit.ly/3Ba6VjB>

limit, as addressed by collapse informatics, is: What if we stopped manufacturing new hardware?

Our discipline may need a radical approach, redesigning the digital world from scratch with specifications based on explicit hardware and software limits. If something is not feasible without assuming some resource will grow as needed, then it should be considered as infeasible.

Let Us Take Some Eggs Out of the Good Old Optimizing Basket

Even if we are not all convinced that optimizations cannot win over rebound effects and that we should therefore impose limits, even if we do not agree on where the limits should be, it would be a good idea not to put all our eggs in one basket. We should devote some research to the selection and preservation of a somewhat minimal, robust, limited-by-construction digital world, and we should teach it. Asking which computer systems can still be designed and maintained, if we cannot count on the unlimited growth of the hardware and the infrastructure, leads to very intellectually challenging research topics. Moreover, it is our responsibility to provide the necessary scientific background to the legitimate questions on technological choices that should be possible in democratic contexts. **□**

References

1. Abbing, R.R. This is a solar-powered website, which means it sometimes goes offline: A design inquiry into degrowth and ICT. In *Proceedings of the Workshop on Computing within Limits*, 6 2021. <https://bit.ly/3cl8cJS>
2. Bousema, T. et al. The critical role of funders in shrinking the carbon footprint of research. *The Lancet Planetary Health* 6, 1 (2022).
3. Dyke, J. et al. Climate scientists: Concept of net zero is a dangerous trap. *The Conversation*, 2021.
4. Freitag, C. et al. The real climate and transformative impact of ICT: A critique of estimates, trends, and regulations. *Patterns* 2, 9 (2021); <https://bit.ly/3aJyuFu>
5. Hess, D.J. *Undone Science: Social Movements, Mobilized Publics, and Industrial Transitions*. MIT Press, 2016.
6. Tomlinson, B. et al. Collapse informatics and practice: Theory, method, and design. *ACM Transactions on Computer-Human Interaction (TOCHI)* 20, 4 (2013).
7. Wikipedia contributors. Rebound effect (conservation). Wikipedia; <https://bit.ly/3z9wuQb>
8. Vardi, M.Y. Publish and perish. *Commun. ACM*, 63, 1 (Jan. 2020), 7.

Florence Maraninchi (florence.maraninchi@univ-grenoble-alpes.fr) is a professor of computer science at University Grenoble Alpes, Grenoble, France.

The author thanks N. Halbawachs, B. Tourancheau, F. Berthoud, J. Combaz, S. Bouveret, and the anonymous reviewers for their thoughtful comments and suggestions.

Copyright held by author.

Article development led by [acmqueue](https://queue.acm.org)
queue.acm.org

What to know now and for the future.

BY ALEXANDROS GAZIS AND ELEFThERIA KATSIRI

Middleware 101

IN COMPUTER SCIENCE, systems are typically divided into two categories: software and hardware. However, there is an additional layer in between, referred to as *middleware*, which is a software pipeline—an operation, a process, or an application between the operating system and the end user. This article aims to define middleware and reflect on its necessity, as well as address controversies about when and where it applies. It also explores the application of middleware in emerging technologies such as cloud computing and the Internet of Things (IoT), as well as future middleware developments.

The term was introduced in the early 1980s. It encompasses complex software solutions that modernize legacy systems—typically mainframes—through new features such as software and application components. Initially, it was solely used to expand the layer separating the network and application layers. Subsequently, its use expanded to serve as the layer above the operating system and network layer, and below the application layer. This means that middleware could now facilitate the generic

communication between the application component and the distributed network.

Through middleware, a programmer has the option to implement a decentralized solution instead of having to interact and analyze different components.¹⁸

In recent literature^{3,12,14,16} multiple definitions have been used, depending on the field of research. On the one hand, both a software and a DevOps engineer would describe middleware as the layer that “glues” together software by different system components; on the other hand, a network engineer would state that middleware is the fault-tolerant and error-checking integration of network connections. In other words, they would define middleware as communication management software. A data engineer, meanwhile, would view middleware as the technology responsible for coordinating, triggering, and orchestrating





actions to process and publish data from various sources, harnessing big data and the IoT. Given that there is no uniform definition of middleware, it is best to adopt a field-specific approach.

The main categories of middleware are as follows:¹¹

► **Transactional.** Processing of multiple synchronous/asynchronous transactions, serving as a cluster of associated requests from distributed systems such as bank transactions or credit card payments.

► **Message-oriented.** *Message queue* and *message passing* architectures, which support synchronous/asynchronous communication. The first operates based on the principle that a queue is used to process information, whereas the second typically operates on a publish/subscribe pattern where an intermediate broker facilitates the communication.

► **Procedural.** *Remote* and *local* architectures to connect, pass, and retrieve software responses of asynchronous communications such as a call operation. Specifically, the first architecture calls a predetermined service of another computer in a network, while the second interacts solely with a local software component.

► **Object-oriented.** Similar to procedural middleware, however, this type of middleware incorporates object-oriented programming design principles. Analytically, its software component encompasses object references, exceptions, and inheritance of properties via distributed object requests. It is typically used synchronously, because it needs to receive a response from a server object to address a client action. Importantly, this type of middleware can also support asynchronous communication via the use of (multi) threads and generally

concurrent programming.

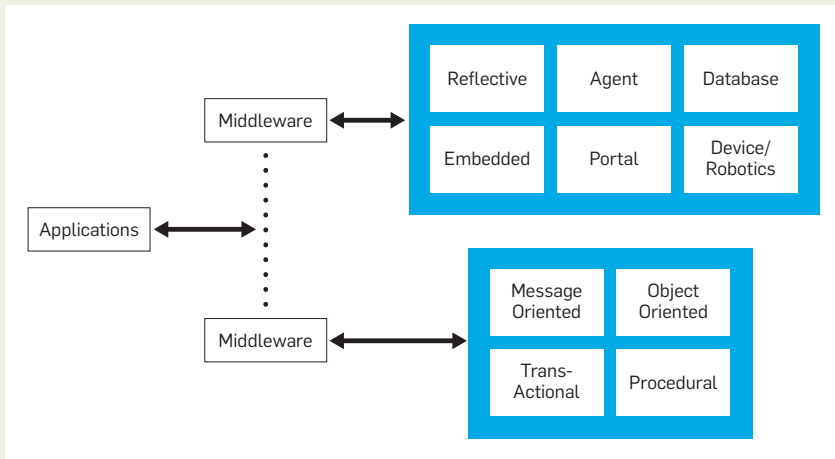
Academics have further segregated middleware depending on the application module it serves, such as database and Web server. There are several types of middleware, falling into these key categories: reflective, agent, database, embedded, portal, and device (or robotics).^{4,15}

First, *reflective* middleware constitutes components that are specifically designed to “easily operate with other components and applications,” while *agent* middleware has multiple components that operate on complex domain-specific languages and laws.

Second, *database* middleware focuses on DB-to-DB or DB-to-apps communication—either natively or via call-level interfaces (CLIs)—while *embedded* middleware acts as the intermediary for embedded integration apps and operating-system communication.

Third, *portal* middleware creates a

Figure 1. Types of middleware.



context-management tool in a composite, single-screen application, while *device* (or *robotics*) middleware simplifies the integration of specific device operating systems or robotic hardware and firmware.

The first categorization is broader, emphasizing architecture operating principles, while the second categorization is application-driven. For this reason, the first segregation is preferable to define middleware accurately per architecture integration instead of its application properties. All types of middleware are presented in Figure 1.

The Use of Middleware

In developing an application, the three necessary elements to consider are scalability, maintenance, and automation. First, developers avoid horizontal scaling, which is just adding resources to expand the capabilities of the main system. They strive for workload partitioning—optimally distributing job scheduling over the overall network. As for maintenance, the separation of concerns principle is very important for developers, both to make each entity reusable (modularity) and to bundle its properties (encapsulation). (Typical modular examples include the Linux kernel, since the code base can be altered—added/removed—and a LEGO set where different elements can be used multiple times to build a system.) Moreover, developers focus on automating operations to reduce errors and make an application available 24/7 or ad hoc (that is, on-demand provisioning).

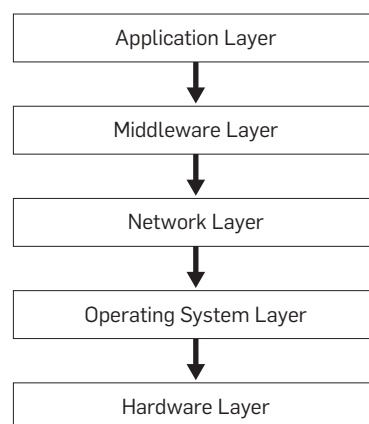
Middleware can act as a facilitator to achieve scalability, maintenance, and automation. Specifically, it adds a layer that simplifies complex systems into small integrations, allowing for their association with the network of distributed resources.² This means that middleware provides agility during software development while simultaneously decreasing the time for a full software cycle; it also provides developers with easier future scaling.

Moreover, middleware can support rapid prototyping by incorporating the “fail fast, succeed faster” principle. It allows developers to apply, adopt, and evaluate business changes instantly. Middleware can also reduce project cost and generally promote entrepreneurship and innovation.⁹

Middleware's Capabilities

Before the widespread use of the Inter-

Figure 2. IoT middleware layers.



net and the adaptation of high-speed connections, most applications were developed as single-tiered, independent software solutions. This software was “monolithic,” built to serve a specified purpose and activity, and thus not designed to connect and interact with other applications and software components. Single-tiered software needed a complex middleware solution either to share information with different modules, such as client/server, or to deliver a request from host/resource-management software.

After the Internet revolutionized the way developers operate, an increasing number of transactions driven by multiple computing devices connected to a large, distributed computer network (also referred to as IoT). *Distributed computing* introduced SOA (service-oriented architecture) instead of monolithic applications. Specifically, SOA consists of multitiered software solutions that implement the separation of entities and services, thus breaking down each component into microservices. This is achieved by decreasing the system's complexity and further increasing its modularity. In this scenario, middleware considers each entity as unique and autonomous. Thus, future modifications are addressed to a specific service (module) and not to the overall system's components.

This middleware “is an approach for developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API.”¹³

Middleware is closely connected to APIs (application protocol interfaces), serving as the tier or a software bundle for different APIs used by a programmer. This means that middleware can simplify sophisticated applications so that the developer focuses on not only the communication of components but also the business logic and the systems' interaction. This is an important aspect in the era of IoT, since APIs are the main gateway to connect devices and send information without errors.

IoT Middleware

The term IoT describes a large network of interconnected devices that

gather real-time data fused by multiple smart sensory devices.¹⁵ To achieve that, electronic devices (mobile/tablet/computer) send data to an external service hosted in a cloud or edge-computing infrastructure. Recent research has focused on developing an IoT network that will not only interact with its surroundings, but also act autonomously without requiring a user's intervention. Under this scope, IoT researches pervasive/ubiquitous computing as the future of computing applications.¹⁰ Computers are no longer associated with single devices or a network of devices. *Pervasive computing* is defined as “the entirety of situative services originating in a digital world, which are perceived through the physical world.”⁸

Moreover, the architectural principles to develop an IoT application include a review of security, energy consumption and monitoring, reliability, interpretability, and communication.¹⁹ As mentioned, middleware provides an abstract tier for all these functions. Based on its software licensing, it can be categorized as corporate-maintained, open sourced, or device-specific (for microcomputers/actuators such as Raspberry Pi or Arduino).

Figure 2 depicts how IoT middleware typically handles its operations based on the following separation of concerns:


- ▶ The hardware layer (also known as the edge layer) includes all the sensory devices, as well as the sensor network in which they operate. This tier is responsible for gathering and processing the available data.

- ▶ The operating system's layer (that is, the access gateway tier) performs the necessary data-transformation operations for information to be extracted and loaded accordingly.


- ▶ The network layer (that is, the Internet layer) focuses on sending data to the next layer by securing a continuous, safe, and nondisruptive communication stream.

- ▶ The middleware layer handles the message communication protocols and services. Specifically, this layer checks systems for operation and data-transmission failures, in addition to providing access protocols for the application.

- ▶ Finally, the application layer is solely responsible for providing ser-



In developing an application, the three necessary elements to consider are scalability, maintenance, and automation.



vices (typically by API) to the end user by enabling the broadcast of services to various application endpoints (for example, different developers and departments).

A review of some of the most promising open-source projects regarding IoT middleware⁶ highlights the following: OpenIoT for sensor systems in the cloud; FIWARE for translating protocols of communication between devices; LinkSmart (formerly known as Hydra) for fast deployment and high scalability of data storage and machine learning; DeviceHive for IoT abstraction of automation layers regarding communication, control, and management; and ThingSpeak for industrial IoT frameworks regarding smart applications.

Similarly, IBM, Amazon Web Services (AWS), Microsoft Azure, Google, and Oracle have developed corporate middleware.¹ Based on the highlighted projects, several middleware frameworks focus on automating either a specific task or a core business activity process.

The Future: Cloud Containers and Microservices

While developers use virtualization (layering of resources into infrastructure), hypervisor (interpreters of the operating system), guest operating systems (with their own kernels), and applications, middleware promoted a decentralized deployment in a single multipurpose environment. This became evident following the exponential increase of *containers*—software environments that can be rapidly and easily deployed multiple times via the same server (host) in an isolated environment, also referred to as a *sandbox*. Like Java's motto, “Write once, run anywhere,” containers are an independent software environment with unique code, libraries, runtime, and dependencies. Middleware tiers have also shifted from virtualization to containerization for the same purpose of optimizing communication and abstracting the communication protocols to develop a software pipeline.


From the scope of a developer,⁷ the shift to cloud-computing solutions means that less coding is required, since most of the work in the cloud infrastructure is performed “under the

hood.” In other words, several aspects of distributed programming and enterprise development previously handled by a local middle tier can now be handled remotely. More specifically, common issues to be tackled include scaling, resilience observability, resource management, and continuous integration and delivery. This means that enterprises will limit the number of middleware developers. Instead of deployment, they will focus on architecture and application development.

Conclusion


Middleware can be used during several phases of the software cycle—from its architecture and development to its deployment. The perpetual need for the digital transformation of businesses (from monolithic to microservice implementations) has showcased that middleware is here to stay. Whether segregating a sophisticated software component into smaller services, transferring data between computers, or creating a general gateway for seamless communication, you can rely on middleware to achieve communication between different devices, applications, and software layers. Moreover, there is a need to educate new developers about middleware and highlight its importance through modern education techniques and learning systems.^{5,17}

Following the increasing agile movement, the tech industry has adopted the use of fast waterfall models to create stacks of layers for each structural need, including integration, communication, data, and security. Given this scope, it is no longer important to study the potential expansion of cloud or data services. Emphasis must now be on endpoint connection and agile development.


This means that middleware should not serve solely as an object-oriented solution to execute simple request-response commands. Middleware can incorporate pull-push events and streams via multiple gateways by combining microservices architectures to develop a holistic decentralized ecosystem. 

References

1. Agarwal, P., Alam, M. Investigating IoT middleware platforms for smart application development. *Smart Cities—Opportunities and Challenges 58* (2020), 231–244. Springer; https://link.springer.com/chapter/10.1007/978-981-15-2545-2_21.



Like Java’s motto, “Write once, run anywhere,” containers are an independent software environment with unique code, libraries, runtime, and dependencies.



2. Al-Jaroodi, J., Nader, M. Middleware is STILL everywhere!!! *Concurrency and Computation: Practice and Experience* 24, 16 (2012), 1919–1926; <https://dl.acm.org/doi/abs/10.1002/cpe.2817>.
3. Becker, C., Julien, C., Lalande, P., Zambonelli, F. Pervasive computing middleware: current trends and emerging challenges. *CCF Transactions on Pervasive Computing and Interaction* 1, 1 (2019), 10–23. Springer; <https://link.springer.com/article/10.1007/s42486-019-00005-2>.
4. Bishop, T.A., Ramesh, K.K. A survey of middleware. In *Proceedings of 18th Intern. Conf. Computers and Their Applications*, 2003, 254–258; https://www.researchgate.net/publication/221205414_A_Survey_of_Middleware.
5. Ciufudean, C., Buzduga, C. Digital engineering education applications. *Trans. Adanves in Engineering Education* 17 (2020), 10–14; <http://www.doi.org/10.37394/232010.2020.17.2>.
6. da Cruz, M.A.A., Rodrigues, J.J.P.C., Sangaiah, A.K., Muhtadi, J.A., Korotaev, V. Performance evaluation of IoT middleware. *J. Network and Computer Applications* 109, C (2018), 53–65; <https://dl.acm.org/doi/abs/10.1016/j.jnca.2018.02.013>.
7. Farahzadi, A., Pooyan, S., Rezazadeh, J., Farahbakhsh, R. Middleware technologies for cloud of things: a survey. *Digital Commun. and Networks* 4, 3 (2018), 176–188; <https://www.sciencedirect.com/science/article/pii/S2352864817301268>.
8. Ferscha, A. Pervasive computing. *Hagenberg Research*. B. Buchberger et. al. Eds. Springer, 2009, 379–431; https://link.springer.com/chapter/10.1007/978-3-642-02127-5_9.
9. Helland, P. Fail-fast is failing... fast! *acmqueue* 19, 1 (2021); <https://dl.acm.org/doi/10.1145/3454122.3458812>.
10. Hong, J. The privacy landscape of pervasive computing. *IEEE Pervasive Computing* 16, 3 (2017), 40–48; <https://ieeexplore.ieee.org/document/7994573>.
11. IBM Cloud Education. What is middleware, 2021; <https://www.ibm.com/cloud/learn/middleware>.
12. Zhang, J., Ma, M., Wang, P., Sun, X.D. Middleware for the Internet of Things: a survey on requirements, enabling technologies, and solutions. *J. of Systems Architecture: The EUROMICRO Journal* 117, C (2021); <https://dl.acm.org/doi/abs/10.1016/j.sysarc.2021.102098>.
13. Lewis, J., Fowler, M. Microservices: a definition of this new architectural term, 2014; <https://martinfowler.com/articles/microservices.html>.
14. Medeiros, R., Fernandes, S., Queiroz, P.G.G. Middleware for the Internet of Things: a systematic literature review. *J. Universal Computer Science* 28, 1 (2022), 54–79; <https://lib.jucs.org/article/71693/>.
15. Pinus, H. Middleware: Past and present a comparison. Seminar paper, 2004; <https://bit.ly/3wdIttw>.
16. Salazar, G.D.S., Venegas, C., Baca, M., Rodríguez, I., Marrone, L. Open middleware proposal for IoT focused on Industry 4.0. In *Proceedings of the IEEE 2nd Colombian Conf. Robotics and Automation*, 2018, 1–6; <https://ieeexplore.ieee.org/document/8588117>.
17. Salem, A.B.M., Mikhalkina, E.V., Nikitaeva, A.Y. Exploration of knowledge engineering paradigms for smart education: techniques, tools, benefits and challenges. *Trans. Advances in Engineering Education*, (2020) 1–9; <http://www.doi.org/10.37394/232010.2020.17.1>.
18. Schantz, R.E., Schmidt, D.C. *Middleware*. *Encyclopedia of Software Engineering*. J.J. Marciniak, Ed. Wiley and Sons, 2002; <https://onlinelibrary.wiley.com/doi/10.1002/0471028959.sof205>.
19. Singh, K.J., Kapoor, D.S. Create your own Internet of Things: a survey of IoT platforms. *IEEE Consumer Electronics Mag.* 6, 2 (2017), 57–68; <https://ieeexplore.ieee.org/document/7879392>.

Alexandros Gazis is a researcher and Ph.D. candidate in the department of electrical and computer engineering, Democritus University of Thrace in Xanthi. He is a member of the Technical Chamber of Greece and works as a software engineer at Eurobank S.A., specializing in core banking systems and mainframe development.

Eleftheria Katsiri is a tenured assistant professor in the department of electrical and computer engineering at Democritus University of Thrace in Xanthi, where she teaches operating systems, structured programming, human-computer interaction, and distributed and parallel systems. Since 2014, she has been an adjunct researcher of the Athena Research and Innovation Center.

Copyright © 2022 held by owner/author.
Publication rights licensed to ACM.

Are we doing this right?

BY ARCHIE L. COBBS

Persistence Programming

MOST SOFTWARE APPLICATIONS require persistence programming of some kind—but what exactly is it, and more importantly, are we doing it right?

A few years ago, my team was working on a commercial Java development project for Enhanced 911 (E911) emergency call centers. We were frustrated

by trying to meet the data-storage requirements of this project using the traditional model of Java over an SQL database. After some reflection about the particular requirements (and non-requirements) of the project, we took a deep breath and decided to create our own custom persistence layer from scratch. This ended up being a lot of work, but it also gave us a chance to rethink persistence programming in Java.

Along the way we uncovered some new, and possibly better, ways to do things. In summary, by reducing the “database” to its core functions and reimplementing everything else in Java, we found that managing persistence became more natural and more powerful. Although this was a Java project, the lessons learned are not specific to Java.

Persistence Programming

First of all, what exactly is *persistence programming*?

One simple definition is your program stores data outside of the context

of the running program itself. Simply saying `int foo = 42` doesn't qualify, but saving data to a file or writing data into a database does. Data is *persistent* if it can be read by a completely different invocation of your program or by a completely different program.

By this definition, you are doing persistence programming when you create an XML (Extensible Markup Language) document and send it over the network. In some sense, an XML document is a little database sent to the receiver, who then opens and reads from it. In fact, one insight from our project is that network communication and database storage involve several common issues, and these issues can be addressed with common tools (more about that later).

Although most software requires some form of persistence programming, programming languages typically provide limited support for it (for example, serialization of basic data types). In the Java programming language, accessing an `int` is idealized: It's atomic, instantaneous, it never

fails, and it never requires a schema migration. If you want to access an `int` that *persists* across program invocations, however, you face a host of new issues, with zero guidance from the language itself. It's then your job to assemble (or homebrew) the required additional components.


This may be the correct choice from a language design perspective, but it has caused persistence programming to evolve unnaturally—from the outside in, so to speak. First, databases were created; then query languages such as SQL were designed to communicate with them (originally for use by *humans*); and finally, libraries were written to allow programs to send SQL queries over the network and retrieve results.

It was left to programmers to figure out how to bridge between the pure, idealized world of programming languages and the practical world of SQL tables, query design, performance, transactions, and network failures. The net result is that programmers have ended up catering to the demands and requirements of the database technology, instead of the other way around.


This can be seen in the many problems with JPA (Java Persistence API), the current state-of-the-art tool for persistence programming in Java. (In 2019, JPA was renamed Jakarta Persistence: https://en.wikipedia.org/wiki/Jakarta_Persistence.) JPA goes to heroic lengths to make querying and retrieving data from an SQL database as painless as possible, but the result is hardly elegant or intuitive.

Several problems with JPA will be discussed, but to be clear, these problems are not JPA's fault. They are inherent in trying to bridge between the abstract and idealized world of high-level, object-oriented programming languages such as Java and the very different world of SQL databases. This is often referred to as the “object relational impedance mismatch.” As one example of this mismatch, consider that JPA defines more than 100 Java enum and annotation classes to cover all the different ways of mapping between the two domains.

This provokes a basic question: Are we doing this right? To answer, let's imagine what persistence program-



Validation checks should be deferred until transaction commit time, unless explicitly requested earlier.



ming might look like if we could start over and design it to address the needs of programmers first.

What Is a Database?

The need to persist data is not going away, so let's consider what is essential about using a database. First, there must be a way to encode programming language data values into raw bits. Ideally these encodings should sort in the same order as the corresponding values in the programming language, which allows the database to respect that ordering.

Once those bits are put into a database, there must be a way to retrieve them. Since you often want to retrieve them in a different order from the way they were put in, the data must be keyed somehow. Then you can give the database an arbitrary key and get the corresponding value back. You may also want the database to keep the keys sorted, so you can iterate them efficiently and query by upper or lower bound. For general-purpose use, the ability to sort is expected, so we'll assume that.

Obviously, key lookup must be efficient, which implies some kind of lookup-optimized data structure. Practically speaking, virtually all databases store data using one of two data structures: hash tables (for unsorted data) or balanced trees (for sorted data).

Finally, any nontrivial database must allow concurrent access, which means that it must define notions of transactions and corresponding semantics for atomicity, isolation, consistency, and durability, among others (see <https://jepsen.io/consistency> for a thorough discussion of consistency levels). There is a wide and interesting variety of models here, so we won't make any specific assumptions.

A modern SQL database provides all of this, plus a bunch of “other stuff.” If you are starting from scratch, however, then a transactional, sorted binary key/value store is a reasonable lowest common denominator to assume. The simplicity of this definition also makes it a great place to define an API, allowing you to easily port existing databases and add new ones.

Everything else that databases typically provide—schemas, indexes, foreign key constraints, and com-

mand line interfaces—can be considered “other stuff” because nothing requires them to be implemented by the database itself. Omitting them from the definition of database leaves room to reimplement these features in a new way that better serves the programmer.

Bringing Persistence into the Language

Now that we have a notion of what a database provides underneath, let’s jump to the top layer and look at what persistence programming might look like from the programming-language level. One way to understand what we really want is to revisit our frustrations with JPA and for each one ask: What would be a better way to do things?

Basic types. In JPA, the first frustration is that the Java basic types don’t match their corresponding SQL types. For example, floating point NaN values are often completely unsupported, SQL’s DATETIME is not the same as `java.util.Date`, and SQL databases sometimes silently truncate or space-pad character string values. In short, if you’re not careful, you won’t reliably read back what you write.

Moreover, the set of supported types is limited and fixed. For example, the only supported array type is `byte[]`. If you need COMPLEX or LAT-LONG type but your database doesn’t natively support it, you’re out of luck.

Instead, we want exact support for any primitive, wrapper, or array type, and the common types such as `String` and `Date`. Equally important, including any new Java type should be possible simply by providing its bit encoding, and these custom types should be first class, in that they are just as sortable and indexable as the built-in types.

Types, classes, and interfaces. JPA must map Java class hierarchies onto rectangular tables, resulting in unused columns or extra joins. Properties may be inherited only from superclasses, not from interfaces (that is, JPA properties must be concrete), and JPA is incompatible with certain uses of Java generics. Instead, we want efficient storage of class hierarchies and full compatibility with Java’s interface inheritance and generics.

Change notifications. JPA supports

basic entity life-cycle notifications via `@PrePersist`, `@PreRemove`, `@PreUpdate`, among others. These apply with per-object granularity (that is, multiple individual property changes will coalesce into a single notification), however, and notifications are generated asynchronously on cache flush, not immediately when they occur.

Instead, precise, synchronous notifications for both object life cycle and individual property change events are desired, as is the ability to detect nonlocal changes in objects reachable through an arbitrary number of references (ideally, either forward or inverse). For example, nodes in a tree with a parent property may want to get notified when the color property in any child (or grandchild) node changes, or vice versa.

Aside from being useful in their own right, having notifications with per-property granularity that can be synchronous and nonlocal is a key enabling technology for other useful new features.

Indexes. When you define an SQL table `USER` with columns `ID` and `USERNAME`, the database creates an internal balanced tree with `ID` keys and `USERNAME` values. If you then index the `USERNAME` column, the database creates a secondary balanced tree underneath the covers with `USERNAME` keys and `ID` values. The database automatically keeps the two trees consistent by updating the secondary tree whenever any `USERNAME` in the primary tree is added, modified, or removed.

That’s a traditional index, which JPA supports. Thinking more generally, however, an index can be any combination of a *secondary data structure* that is entirely derived from primary data; *change notifications* that notify about changes in the primary data; and an *update algorithm* that updates the secondary data structure when notified. Why shouldn’t you be able to create any kind of index simply by defining `a`, `b`, and `c`?

Suppose you have a table of home values and you frequently want to access the median home value. This is not something you can index (or even directly query) in SQL. If the database could notify you whenever any home value is added, removed, or changed,

however, a simple secondary data structure and update algorithm could complete the picture: Create a traditional index on home values so you can quickly find neighboring values, and then store the current median, the number of lower values, and the number of higher values in a new secondary data structure.

This example shows why change updates must be synchronous and per-property rather than per-object. Support for nonlocal change notifications is also important, because sometimes the information you want to index is not limited to a single object. Suppose you want nodes in a tree to index how many “blue” child nodes they have. Nodes could store that number in a private field `numBlueChildNodes` and register for change notifications on the `parent` and `color` properties of child nodes to keep that field up to date.

Of course, any such index could be implemented manually by the programmer with more work, but it’s less messy and more robust when the database provides the change notifications. After all, the database is in the perfect position to do so because it sees every change to every data value. In summary, we want the database to provide tools that make it easy to implement arbitrary custom indexes. Synchronous, nonlocal, object-, and property-based change notifications are such a tool.

Validation and invariants. Key to any software that manages data is validation—that is, the maintenance and verification of required invariants (aka validation constraints). When a transaction starts, your code assumes the invariants hold, and your goal is to do whatever needs to be done, even if possibly violating some invariants temporarily, while ultimately ensuring the invariants are reestablished by the time the transaction commits. It’s the same principle that applies within Java synchronized blocks.

Enforcing invariants efficiently requires: code that can check the constraint; and a notification that fires when the constraint needs to be rechecked as a result of a change in the associated data. Again, precise change notifications are a key ingredient.

In JPA, the database itself provides

a few validation constraints, such as foreign-key integrity and column uniqueness. At the Java level, JSR 303 (Java Specification Request) validation provides additional per-object validation. Both these implementations are imperfect: JPA can trigger phantom foreign-key violations caused by cache flush ordering (for example, when persisting two new objects that indirectly refer to each other); and both types of constraints apply on cache flush, not on transaction commit, which means they can happen in the middle of a transaction, before the invariants have been reestablished.

Instead, validation checks should be deferred until transaction commit time, unless explicitly requested earlier. The ability to enqueue objects for validation manually is important, as this makes implementing arbitrary custom (and precise) validation constraints easy: Simply register for change notifications on the fields involved and then enqueue for validation when notified.

With nonlocal change notifications, validation constraints can span multiple objects. For example, imagine implementing a constraint that no two child nodes may both be blue. You would register a change notification on child node `parent` and `color` properties, and when notified, enqueue the parent node for validation.

In practice, indexing and validation often work together. In the previous example, you could index the number of blue child nodes using a private `numBlueChildNodes` property, and then simply enqueue for validation whenever `numBlueChildNodes` changes.

How do you query? Is the following SQL query efficient? `SELECT * FROM User WHERE lastName = 'Smith'`.

It's a trick question, because the answer depends on whether `lastName` is indexed, and this can't be determined by looking at the query. This violates the basic principle that software should be understandable by looking at it.

In JPA, programmers are required to learn and use a new query language, but even when they do, there's a lack of performance transparency. (JPA has three ways to query: SQL, JPQL, and Criteria.) To understand

whether your queries are efficient, your skill set has to contain the union of computer programmer and database administrator.

In an ideal world, inefficient queries shouldn't be able to hide like this. If you're about to iterate through every `User` in the database, that should be obvious when looking at the code.

What would be better is if queries were written in normal Java, using existing concepts such as `Set`, `List`, and `Map`. Then, a query's efficiency would always be obvious, or at least visible. The sorted key/value pairs that the database provides can be modeled in Java as a `NavigableMap`, and data extracted can be represented by a `Stream`. An indexed property is then just a `Map` from property value to the set of objects (such as key prefixes) with that value in that property. In fact, the Java language already has all the tools you need to query, using existing concepts that programmers already understand.

Schemas and migrations. Code evolves over time, and that means database schemas do as well. Therefore, the database structure must sometimes be updated via schema migrations. These migrations have two aspects: *structural changes* to the actual data format or layout; and *semantic changes* that are the corresponding "fixups" to the data.

For example, if you replace the `lastName` and `firstName` columns with `fullName`, the structural change is the `ALTER TABLE` stuff to add and remove columns, while the semantic change is initializing each row's new `fullName` column to be the concatenation of `firstName` and `lastName`. Note that performing the semantic change requires access to both the old (`lastName`, `firstName`) and new (`fullName`) columns.

JPA provides no tools for schema migration, nor does it verify that the schema being used is correct. Helper libraries exist for tracking schema migrations, but they require "stop the world" operations such as `ALTER TABLE`, which are incompatible with rolling (zero downtime) updates, where multiple schemas can exist in the database at the same time. Moreover, these tools typically require both structural and semantic changes to be

written manually, and in SQL rather than Java.

We would like to improve this in several ways: First, the database should be able to support more than one schema at a time, allowing rolling schema migrations where objects can be upgraded over time (for example, on demand), so you never need "stop the world" operations. Second, the schema(s) being used in the database should be tracked automatically in the database itself and then verified against what the code expects at the start of each transaction (obviously, this check should be efficient). Third, there's no reason *structural* changes can't be fully automated, which would eliminate bugs caused by inconsistent migrations. Finally, *semantic* changes would be written more naturally in Java than in SQL.

Offline data. JPA has a somewhat awkward model for offline data, that is, data read from a transaction but used after that transaction has closed. For example, touching a collection that was not loaded during the transaction throws an exception, but touching a collection during the transaction loads the entire thing, which can get unwieldy. In any case, it's not always clear what offline data is available, because JPA conflates offline data with its online cache: Offline data is whatever happened to be in the online cache when the transaction closed. Moreover, once the transaction closes, you lose the ability to query into your offline data, and even if you could, the query would be slow because no associated index information is retained.

JPA includes support for *fetch joins* and *load graphs*, but these can only partially address the problem, because you don't always know what data you'll need next until you've seen some of the data first. The core problem is that SQL is often not expressive or precise enough to define the exact data you need, even if you happen to know it ahead of time, without omitting some data or causing a "join explosion."

It would be nice to have a more precise way to define what data to copy and retain as offline data after the transaction has closed. In addition, you should then be able to query offline data in all the usual ways, including index queries. This implies that

secondary index data be retained as well. In short, it should be possible to define a subset of the whole database precisely, query it and pull it into memory, and then treat that like a normal in-memory mini-database even after the original transaction has closed.

For database designs that support snapshots efficiently (for example, log-structured databases), this process can even be a zero-copy operation, where the in-memory mini-database is really just a read-only, memory-mapped view into a snapshot.

Network communication. As mentioned earlier, network communication and persistence programming have many issues in common. Assuming a database is just a sorted key/value store, then network communication can be redefined as a simple form of persistence programming: First, create and initialize an in-memory database, including the usual schema-tracking information; second, populate that database with Java objects to transmit; finally, serialize the database (just a bunch of key/value pairs) and send them over the network. On the receiving end, open and read the database just as normal. Because the database also contains schema information, any required schema migrations happen automatically.

Many of the database issues that normally reappear with network programming are thus taken care of automatically: how to serialize an arbitrary graph of objects, define and document the data format (that is, schema), migrate automatically when different versions of the code are running on either end of the connection (for example, during rolling upgrades), and query into the data efficiently on the receiving end.

Bringing Code and Data Together


We developed Permazen, an open source project (<https://github.com/permazen/permazen>) to investigate and prototype the concepts described in this article, and now use it in our commercial solution. All of these ideas were implemented and deployed in some form, and programming with our new persistence layer was a truly refreshing experience. It also enabled us to implement some required custom functionality that would other-

Network communication and persistence programming have many issues in common.

wise have been difficult or impossible—for example, a clustered key/value database based on the Raft Consensus Algorithm (<https://raft.github.io/>) with support for “standalone” mode. The project demonstrates that these ideas are actually feasible and perhaps worth further exploration.

However, a key caveat allowed this project to succeed: Each node was required to keep a complete, up-to-date local copy of the database (in case it needs to revert to standalone mode). As a result, individual database accesses within each transaction were low latency, because mostly, they required no network traffic. This was crucial to many of the new features mentioned here, such as nonlocal change notifications and custom indexes, which require frequent—but low-volume—access to the data in each transaction.

Put another way, the usual method of persistence programming using SQL over a network connection is kind of like grocery shopping via walkie-talkie with an intermediary who speaks Latin. If you can cut out the intermediary and go there yourself, the experience can be much more productive. In other words, distance itself (that is, latency) is a barrier to innovation in persistence programming. Sometimes distance is unavoidable, but in situations where you can give the code low-latency access to the data, new possibilities arise. In situations where the database and the application are separated by a network, this argues for sending the code to the data instead of the data to the code.

Historically, persistence programming has been driven from the *database* side, and this has limited programmers’ options. Redefining the database as just a sorted key/value store creates more room for innovation from the programming-language side. Our experience shows that, at least in some scenarios, this allows reimagining persistence programming to make it more natural and less frustrating, so we can spend less time wondering: Are we doing this right? 

Archie L. Cobbs is a software developer and entrepreneur who has worked with software startups his entire career. He has also created and contributed to many open source projects.

Copyright held by owner/author.
Publication rights licensed to ACM.

CARMELA TRONCOSO
EPFL, SWITZERLAND

DAN BOGDANOV
CYBERNETICA AS, ESTONIA

EDOUARD BUGNION
EPFL, SWITZERLAND

SYLVAIN CHATEL
EPFL, SWITZERLAND

CAS CREMERS
CISPA HELMHOLTZ CENTER FOR
INFORMATION SECURITY, GERMANY

SEDA GÜRSES
DELFT UNIVERSITY OF TECHNOLOGY,
NETHERLANDS

JEAN-PIERRE HUBAUX
EPFL, SWITZERLAND

DENNIS JACKSON
MOZILLA, U.K.

JAMES R. LARUS
EPFL, SWITZERLAND

WOUTER LUEKS
EPFL, SWITZERLAND

RUI OLIVEIRA
UNIVERSITY OF MINHO
AND INESC TEC, PORTUGAL

MATHIAS PAYER
EPFL, SWITZERLAND

BART PRENEEL
KU LEUVEN AND IMEC, BELGIUM

APOSTOLOS PYRGELIS
EPFL, SWITZERLAND

MARCEL SALATHÉ
EPFL, SWITZERLAND

THERESA STADLER
EPFL, SWITZERLAND

MICHAEL VEALE
UNIVERSITY COLLEGE OF LONDON, U.K.

Lessons from a pandemic.

Deploying Decentralized, Privacy- Preserving Proximity Tracing

CONTACT TRACING IS a time-proven technique for breaking infection chains in epidemics. Public health officials interview those who come in contact with an infectious agent, such as a virus, to identify exposed, potentially infected people. These contacts are notified that they are at risk and should take efforts to avoid infecting others—for example, by going into quarantine, taking a test, wearing a mask continuously, or taking other precautionary measures.

In March 2020, as the first wave of the COVID-19 pandemic was peaking, traditional manual contact tracing efforts in many countries were overwhelmed by the sheer volume of cases; by the rapid speed at which SARS-CoV-2 spread; and by the large fraction of asymptomatic, yet infectious, individuals.



Many people quickly and independently proposed using ubiquitous smartphones to implement *digital contact tracing* (DCT). In this new approach, an app on a user’s phone could record contacts (encounters with other people) of sufficient time duration. If a physically close contact was diagnosed as infected, the app could inform the phone’s potentially infected user. The envisioned technology would complement manual contact tracing by notifying people faster; reducing the burden on trained contact tracers; increasing scalability; and finding anonymous contacts, such as those in public spaces like shops and transportation, who would be otherwise unreachable through traditional systems.

Due to the fast-moving pandemic, the need for DCT was urgent, and had to be designed, developed, and deployed in a highly compressed timeline. This pressure limited the design scope and constrained many decisions. For example, manufacturing and distributing new hardware to the public would have incurred substantial delays, so viable solutions could only make use of sensor technology already widely deployed on consumer mobile phones and existing communication infrastructure.

A further challenge was to ensure the infrastructural components deployed for DCT could not be used to invade individual privacy or facilitate human rights abuses. For example, DCT applications that collect and share time-stamped and geo-located records of people’s physical contacts can be easily repurposed for illegitimate, oppressive uses beyond public health. This happened with contact-tracing information collected in paper form^{19,39} and has led to increased surveillance² and stigmatization.²⁵ Moreover, databases recording peoples’ locations are susceptible to being leaked, intentionally or unintentionally.³⁴ During an event requiring an internationally coordinated response, the potential for abuse of a new technology could not be ignored at the design stage, especially with respect to the varying political and governmental systems and rule of law (particularly during states of emergency).

Furthermore, trustworthy and transparent technology is essential to achieve the high voluntary adoption

» key insights

- **It is possible to build privacy-preserving systems that not only collect and process little information but ensure information can only be used for a single purpose. Our contact-tracing system can only be used to notify contacts.**
- **Successful deployment of privacy-preserving solutions requires consideration of the broader context in which these solutions must operate. For example, integrating contact-tracing apps with public health systems is essential.**
- **Reliance on third-party technologies, in particular mobile platforms, severely constrains the deployment of privacy-preserving systems.**

necessary for maximal public health impact, particularly in countries with a history of poor data management. Achieving and retaining trust is an international effort—data breaches or misuse in one country can resonate around the world. Thus, limiting abuse is fundamental to achieving public health goals.

The authors represent a major portion of the group that designed the Decentralized Privacy-Preserving Proximity Tracing (DP-3T) protocol, which heavily influenced the Google and Apple Exposure Notification (GAEN) framework used by most DCT apps, and we helped deploy five apps: SwissCOVID^a (Switzerland), Corona Warn App^b (Germany), STAYAWAY COVID^c (Portugal), Coronalert^d (Belgium), HOIA^e (Estonia), and the European Federated Gateway Server.^f In this article, we describe the lessons learned from our efforts to design, develop, and deploy digital contact tracing. We detail the hurdles and challenges, including the design of underlying cryptographic protocols, the development of mechanisms to ensure end-to-end privacy for users, and the integration of the apps within public health systems. We also discuss some issues raised in the media concerning the contact-tracing apps’ effectiveness, security, and independence from device manufacturers. We conclude with recommen-

a <https://foph-coronavirus.ch/swisscovid-app/>

b <https://www.coronawarn.app/en/>

c <https://stayawaycovid.pt/landing-page/>

d <https://coronalert.be/en/>

e <https://hoia.me/en/>

f https://ec.europa.eu/commission/presscorner/detail/en/ip_20_1904

dations to help prepare technologically for the next emergency.

Digital Privacy-Preserving, Proximity-Tracing Protocols

In response to a pressing need for DCT to combat the COVID-19 pandemic, a substantial number of proposals were put forward, using a diverse range of technologies including Bluetooth, Ultrasound, and the Global Navigation Satellite System (GNSS). Of these, Bluetooth was most widely adopted, in large part due to privacy concerns around access to microphone and location information. In this article, we focus exclusively on proposals that use Bluetooth Low Energy (BLE) beacons to detect proximity without knowledge of a contact’s location or identity.

Common to these proposals, a smartphone runs a contact-tracing app that executes a Bluetooth contact-tracing protocol, also known as a proximity-tracing protocol. As such, devices broadcast ephemeral identifiers using BLE beacons. Broadcasting, instead of point-to-point connections, ensures that the number of devices able to receive the identifiers is not limited by a phone’s Bluetooth connection rate. A phone that receives such an identifier records it, alongside information about the signal’s power, which can be used to estimate the proximity of the transmitting device. Later, these records provide a basis for a risk calculation based on the estimated proximity to contagious individuals.

The proposals differ on how the risk calculation is carried out, and on the capabilities they require from mobile phones and the communication infrastructure. A few dozen non-deployed academic proposals explore different privacy/security trade-offs—for example by using more complex cryptography^{7,9,37} or requiring currently nonexistent or non-scalable infrastructure as a basis for their strong privacy guarantees.^{1,3,9}

We only discuss the BLE-based protocols whose design allows immediate deployment. We categorize them as either centralized or decentralized according to their risk calculation. In the former, a central server carries out the risk calculation on behalf of all users and notifies those it considers to be at risk. In the latter, each user’s device carries out an individual risk calculation to


decide whether to notify the user. For both, we provide a high-level description of their operation and their security and privacy properties. For more, we refer the reader to our detailed analysis and comparison.¹⁴

Centralized proximity-tracing protocols. This class of contact-tracing protocols,^{4,13,28,32} pioneered by Singapore's BlueTrace app,⁴ uses a central server to generate ephemeral identifiers that a phone downloads and periodically broadcasts. When a user receives a positive COVID-19 diagnosis, that user's phone app uploads all the identifiers it received to the server. The server performs a matching process on these identifiers to determine who was in prolonged contact with the COVID-19-positive person and notifies those people of a potential exposure.


Security and privacy. In a *centralized* design, the server generates the ephemeral identifiers and associates them with the long-term identities that are necessary to send notifications. Therefore, an adversary with access to the server can de-anonymize any observed BLE beacons. Moreover, an adversary who can influence the server can falsely notify a user of infection. In addition, the server's information allows inference of relationships among users (who they met, when, and for how long). This information can be inferred even for users who do not test positive, so long as they come in contact with a positive user. This can lead to scope creep, where the system is explicitly or implicitly repurposed, such as in Singapore, where the police were given access to the app-related databases for law enforcement purposes.²³

Decentralized proximity-tracing protocols. To avoid the security and privacy shortcomings of the centralized approach, a number of *decentralized* protocols^{8,10,31,36,38} moved the generation of the ephemeral identifiers broadcast in BLE beacons and the matching process to run entirely on an individual's smartphone. This design reduces the power of the central server by limiting its role to checking that a user has been diagnosed by a healthcare provider and to distributing public information.

We now present our design, the DP-3T protocol.³⁸ Other protocols, which appeared concurrently, are very simi-



Due to the fast-moving pandemic, the need for DCT was urgent, and had to be designed, developed, and deployed in a highly compressed timeline.



lar.^{10,13,36} In a setup phase, a phone creates a secret seed SK_t . After that, the DP-3T protocol operates in three steps:

1. Local ephemeral identifier creation. The phone app creates its ephemeral identifiers (EphIDs) using the following procedure:

- ▶ Each day, the secret seed is rotated using a simple, non-reversible transformation: $SK_{t+1} = H(SK_t)$, where H is a hash function.

- ▶ Phones derive $n = (24 * 60)/L$ ephemeral identifiers (EphIDs) from the daily seed: $\text{EphID}_1 || \dots || \text{EphID}_n = \text{PRG}(\text{PRF}(SK_t, \text{"broadcast key"}))$, where PRF is a pseudo-random function (for example, HMAC-SHA256), "broadcast key" is a fixed public string, and PRG is a pseudo-random generator (for instance, AES in counter mode).

Each EphID is broadcast for L minutes. The value L is public. Smartphones pick a random order in which to broadcast these EphIDs. One cannot link two such identifiers without knowing the key SK_t .

2. Operation: Storage of beacons and seeds. For each received beacon, a phone stores:

- ▶ The received ephemeral Bluetooth identifier EphID.
- ▶ The exposure measurement (for example, signal attenuation).
- ▶ The day on which this beacon was received (for instance, "April 2").

Phones store beacons indexed by EphID. In addition, each device stores the seeds SK_t it generated for as long as recommended by health authorities (for example, 14 days).

3. Local notification procedure. Users who receive a positive COVID-19 test are authorized by the health authority to upload the seed SK_t to the central server, corresponding to the first day when they are likely to have been contagious. After the upload, user devices randomly generate a new secret seed SK_t to prevent linkability with respect to previous secret keys.

All phones periodically download the seeds of COVID-19-positive users from this server. With each seed, a smartphone can locally reconstruct the list of EphIDs broadcast by a diagnosed person for one day. The app matches these EphIDs to check two things: If the phone observed a beacon with one of these EphIDs and if the observation occurred before the corresponding seed

SK_i was published (to avoid replay attacks in which EphIDs are re-derived from a published SK_i and retransmitted). Using the signal strength of the set of observed beacons, the smartphone locally computes how long and at what distance its user was exposed to COVID-19-positive people. If the time frame is long and the distance close enough, the phone notifies its user of a high-risk contact indicating a possible contagion.

Security and privacy. In this decentralized design, the server has no information to link ephemeral identifiers. The server also cannot influence the generation of identifiers or arbitrarily mark users as at-risk. Most implementations of these protocols try to reduce the amount of information transmitted by the server to save bandwidth. Unfortunately, more bandwidth-efficient implementations (such as the one described earlier) enable linking of beacons broadcast by positive users on the days they were infectious. The protocol fully protects non-positive users. Slight changes in the cryptographic protocols can combat linkability, at the expense of increasing bandwidth (see unlinkable scheme in Troncoso et al.³⁸).

From DP3T to GAEN. Google and Apple subsequently implemented a decentralized DCT framework, very similar to DP-3T,³⁸ in their GAEN framework.¹⁸ The main difference is the creation of a fresh new key every day and the use of a different derivation to create the EphIDs.

This framework is currently the basis of more than 40 DCT apps in Europe and North and South America; the number of downloads is estimated to be at least 90 million. Almost all European countries and U.S. states adopted the decentralized approach because of its strong privacy benefits and support from mobile operating-system vendors. Currently, apps from 14 countries in the EU are connected through the European Federated Gateway System.¹⁷ This gateway enables exchanges between apps using the GAEN decentralized framework (see section titled Integration Across Health Systems). National applications were used throughout the pandemic, as part of national test-and-trace strategies. In most European countries, DCT apps were suspended together with test-and-trace during the first half of 2022.



Trustworthy and transparent technology is essential to achieve the high voluntary adoption necessary for maximal public health impact.



From Protocol to System: Integration Challenges

While instrumental to the operation and the security and privacy guarantees of a DCT app, a proximity-tracing protocol is just a small piece in the larger challenge of reducing COVID-19 transmission. This protocol must be implemented in mobile apps that run on a large and diverse collection of phones that differ in firmware and hardware. In turn, the apps and server must be integrated into a public health system that acts as an interface between health services and users. Each step of integration brings new operational challenges and difficulties in maintaining end-to-end security and privacy.

Integration with existing hardware. Bluetooth's ubiquity offers a solid basis for building widely deployed privacy-preserving systems. However, Bluetooth also imposes numerous constraints. For example, support in hardware and operating systems varies widely, often exposing differing APIs with limited functionality to an application. Apple's *CoreLocation* API allows BLE to function much more extensively in the background than its generic *CoreBluetooth* API yet functions only with proprietary "iBeacons," thereby prohibiting interaction with non-Apple devices. Similarly, capabilities concerning transmission power, tag options, or permissions vary between Android versions. A further complication is ensuring a system has minimal impact on battery life while maintaining reliable message reception and transmission.

Many proposals opted for a connectionless broadcast system that requires each phone to produce a constant stream of broadcast messages. While a connection-based approach could in theory be used, it would require substantially more battery power to maintain a connection with each nearby phone and would encounter interference problems in crowded environments. Moreover, the most widely used Bluetooth broadcast standard supports only relatively small beacon payloads, which imposes another design constraint. Another privacy problem is the highly varied support for Bluetooth privacy extensions, such as rotating MAC addresses. Although address rotation is recommended to

mitigate user tracking, many devices do not support it. Without it, an adversary can track users (as the MAC address remains the same with an effect comparable to broadcasting a single static ID), despite the cryptographic precautions included in DCT proposals.¹⁴

Integration in the mobile operating system. When designing DCT protocols, designers assume these protocols will be part of a DCT app and will operate independently of the mobile operating system. However, the highly integrated design of mobile-phone platforms means that a DCT protocol must be integrated into a phone's operating system. This is necessary to guarantee that beacons will be reliably sent and received, to limit battery consumption, and to ensure that protection at the hardware level (for example, MAC rotation) is applied. Next, we describe how the consequences of this integration strongly impact the way apps operate and can be deployed.

The GAEN API¹⁸ went beyond the necessary integration. It presented an app with an API with a heavily constrained set of parameters. These constraints strongly limited the design choices of app developers in making tradeoffs among privacy, security, and epidemiological utility of the applications.

For example, the first version of the GAEN API provided apps with only heavily summarized information about observed beacons, with the operating system performing the exposure computation and providing a result through API calls. Initial API versions allowed only limited forms of aggregation. Specifically, it did not permit computation of daily viral exposure accumulation. As a result, early versions of SwissCovid, Radar COVID (Spain), STAYAWAY COVID, and HOIA, whose epidemiology experts opted for day-based computation, had to work around the limitations by performing multiple API queries, thereby delaying notifications for up to eight hours. This was eventually resolved in GAEN version 1.6.

As a further example, the early version of the GAEN API did not permit the release of daily keys until they expired. This security mechanism, aimed at reducing the likelihood of a replay attack, affected how apps could upload keys to the central server. An authorized user could still be infectious; thus, that

user should upload keys up to and including the day of upload. As a result, most apps had to change their original authorization schemes to perform a second authorization to upload on the subsequent day without user intervention. This not only changed the app's functional flows, but also changed the security analysis: A second upload behind a user's back carries the risk that its authorization can be misused to upload unauthorized cryptographic material to the server.

Finally, how and when the GAEN API was integrated in the operating system strongly affected the availability of DCT apps. For example, old versions of iOS, such as those for the iPhone 6, were only supported six months after the initial release of the framework. This affected a non-negligible number of users, who lost interest in using the app. Other older iOS versions, even those originally supported by the framework, do not have good background task management, which hinders the apps by not permitting them to wake up periodically to download new information about infected users.

Integration into a health system. In all DCT applications, a key step in the process is when a COVID-19-positive person uploads ephemeral identifier

seeds to a server. To ensure that only infected users upload their identifier seeds, minimizing the likelihood of fake alerts, apps require an authorization key from the health system. However, only a small number of proposals specified how this key could be securely provided.^{9,15}

While this process may seem simple, it has proven to be a major challenge because most countries' health systems lack a comprehensive digitized framework to manage aspects of the pandemic response, including test results and interactions with people. Often these systems are not even computerized and consist of a few disconnected databases and personnel who cannot communicate digitally with patients. To deal with this situation, most apps use a very simple authorization mechanism that is communicated to users by phone or SMS. Moreover, the DCT infrastructure needed to be developed, maintained, and secured on an ongoing basis, often by governmental departments that lack the experience or competence to directly run such services.

Integration across health systems. Finally, the pandemic is a global problem. Around the world, people frequently travel and commute between



states or countries. In such situations, apps must be able to trigger notifications across borders.

When designing DCT protocols, ease of interoperability was a consideration. Exchange of information across borders is in principle facilitated by the privacy guarantees of the decentralized protocols. In these protocols, only the keys from infected users must be exchanged. The keys are not sensitive since they carry no information about individuals, their location, or their interactions with others.

In practice, legal experts have categorized the uploaded seeds as pseudonymous personal data under GDPR, which means that legal rules influence how they can be shared and with whom. Such legal considerations hindered the fast deployment of the European Federated Gateway Server (EFGS)¹⁷ used by the decentralized apps in most countries in Europe to exchange keys.

Interoperability also becomes complex when countries configure the GAEN API in different ways to estimate exposure risk. Even if a country uses a simple set of the parameters for its own risk function—for example, SwissCovid³⁵—its server may need to collect extra information to support the more complex risk functions of other countries—for instance, CoronaWarnApp.¹¹ This necessitates the creation of common standards to exchange metadata associated with keys and complicates the logic that interprets and supports other risk functions.

Increasing complexity can also affect the privacy promised from a country's app to its citizens. As extra information is published by other countries to enable their risk estimation, this information becomes available to an adversary, who can use it to reduce the anonymity sets of users. Mitigating this leakage requires developers to carefully select the information that is shared to minimize inferences while still enabling meaningful risk estimation.

Deploying Large-Scale DCT: Lessons Learned

The challenges noted in the previous section hindered the deployment of these apps at many steps, requiring extra engineering to build and deploy the apps at a large scale. During deployment

in several countries, we learned many valuable lessons which highlight how often research and academic work are not aligned with real-world demands. Academic research sometimes aims toward a level of perfection beyond that which is demanded in real situations. Academic work, moreover, typically focuses on one aspect of a system and requires additional mechanisms to ensure that the security and privacy properties encompass all the elements of a deployed system, from phone to cloud.

Privacy must be guaranteed at all layers. Long discussions in academia and public forums focused on competing DCT protocols and their security and privacy properties. However, as we previously noted, the cryptographic protocol is just a small part of a DCT system. Information flows that complement the low-level protocol can result in privacy leaks that must be prevented with additional mechanisms.

Privacy at the network layer. Information uploaded to the server cannot be linked to the identity of users reporting their positive status. However, the mere existence of the connection to upload this information reveals the health status (SARS-CoV-2-positive) of the user to any adversary who can see a user's IP address (for example, an eavesdropper on a Wi-Fi network or the Internet service provider).

In academia, the typical mitigation is for the app to generate dummy traffic, in addition to its real traffic, to help obscure when real actions occur. Even though well-known and frequently proposed, dummy traffic is non-trivial to properly implement. For instance, configuring traffic requires an understanding of the actual usage patterns that must be mimicked. In reality, this pattern is often unknown, particularly for a new and unprecedented service such as DCT. One simple option is to over-provision the dummy traffic, but this could affect the user experience by reducing battery life and consuming data bandwidth. To balance these requirements, instead of aiming to make dummy and real traffic indistinguishable, plausible deniability is a more attainable goal. This enables the system to deny that some actions are real (in this case to deny that actual uploads happened by claiming uploads are dummies). It is typically

considered weak in an academic publication but often suffices in practice.

Privacy of the authentication scheme. Besides hiding the traffic patterns associated with an upload, it is also important that the authentication mechanism does not provide additional information about (1) the identity of the user or (2) the link between a user and the information the user uploads.¹⁵

While powerful cryptographic tools exist to achieve security and unlinkability at the same time, such as anonymous credentials or cryptographic commitments, using them in practice requires a highly digitalized health system not available in most countries. Further, even with systems using anonymous credentials, many orthogonal means of inference are available to an adversary, such as timing or IP-address metadata. As this latter class of metadata is very difficult to conceal,⁸ most apps use a simple code-based authentication scheme and trust the servers to not log information that would enable the linkage of users' IPs and their ephemeral identifiers.

Contact-tracing applications are most valuable when widely used, so some countries opted to host their servers in public clouds to support high loads. Other countries hosted their servers locally in infrastructure owned by the government or local companies. Whether hosting is public or private, a large number of users requires technologies, such as load balancers and firewalls, that can log information outside of the control of the app designers. Careful design of the app server's logging policy is vital to ensure that none of the information logged by the cloud infrastructure can be used to breach the app user's privacy.

Exposure estimation goes beyond distance measurement. Another point of contention in academic circles and public discussions is the accuracy of BLE when measuring distance and its suitability as the underlying technology for DCT.^{27,41} In practice, while accuracy matters and improvements in distance measurement at the Bluetooth layer would be valuable,^{22,29} it is important to remember that the goal


^g Contemporary metadata hiding systems such as Tor do not scale to hundreds of millions of users.

of a contact-tracing app is not to measure a precise distance at one point in time but instead to estimate a person's exposure to other COVID-19-positive people over a period of time.


It is also important to keep in mind that the epidemiological basis for computing exposure is not an exact science. The technological solution mimics a contact-tracing interview in which patients are asked to recall close contacts, typically defined as those occurring longer than 15 min. within 2 m (6 ft.). In such a situation, a patient's estimation of distance is naturally limited in precision and accuracy by human perception and memory. Moreover, the 2-m criterion is itself an approximation. There is no specific distance at which the virus stops traveling, and the high-risk zone depends very heavily on environmental conditions, such as air circulation. Later in the article, we discuss complementary protocols for notifying about contamination in poorly ventilated spaces in which contagion can occur well beyond 2 m.

Third, the computation of exposure with the GAEN framework is constrained by the frequency of measurement and the information exposed to apps via the framework's API, which limits how information can be combined. As a result, existing contact-tracing apps follow diverse strategies, ranging from very simple approaches³⁵ to complex formulae.^{6,11} These constraints on the exposure computation also reduce the importance of distance measurement accuracy, as it gets diluted in the aggregation function and degraded by the measurement frequency.

Even the best technology underperforms if not used. Researchers and developers have focused on optimizing the technology by improving measurement accuracy and proposing many variations to the protocol. These alternatives offer different tradeoffs among security, privacy, and device capabilities (for example, battery consumption, sensor usage, and use of devices beyond the phone). However, in the end, no improvement can increase the value of a technology that lacks broad adoption. Achieving this end requires good integration, not only in a technical sense but also with the processes and



Google and Apple subsequently implemented a decentralized DCT framework, very similar to DP-3T, in their Exposure Notification framework.



individuals in the broad environment.

The environment of contact-tracing apps is complex, with many stakeholders: governments, public health services and employees, mobile operators, mobile operating systems developers, and, of course, users. In this article, we have discussed the technical integration difficulties arising from the strong dependence of these apps on the operating system and changes by mobile-system developers. However, throughout the world, the principal difficulties confronting these apps arise in procedural and social integration:

Rollout by public entities. Rollout of this technology by health authorities in numerous countries was a complicated process involving numerous facets of the public health system as well as the unfamiliar deployment of technical infrastructure and a publicly available app. To start, each health authority needed to procure or otherwise build a DCT system for its local market. Open source, such as that produced by DP-3T and other projects, helped development in countries with fewer resources. In addition, Apple and Google support development efforts in many countries and incorporated an Express app into their later OS releases. Even with this support, not all countries could promptly roll out an application-supported DCT system or maintain it properly.

External dependencies. The operation cycle of contact-tracing apps is not completely technical.⁵ Two crucial steps are outside the protocol: delivering the authorization code to users and contacting the health system after a notification. In most countries, these processes require human intervention and have proven to be the least reliable part of the system. The difficulty of incorporating these steps into existing medical practice introduces delays and communication failures that decrease the health impact and may eventually cause users to abandon the application, leading to a public perception that the app is not useful or functional.

Bottlenecks also appeared in systems that did not use authorization codes and instead integrated with national e-health records holding test results. In Estonia, requiring strong authentication for infection confirma-

tion proved to be the limiting factor. Even though multiple authentication methods were offered, not all patients with positive COVID-19 test results had access to at least one method for confirming their infection, preventing them from notifying others.

Communication strategy. The adoption of contact-tracing apps depends on multiple factors.^{21,24} Among them is a user's perception of the utility and risks stemming from using an app. Both studies, and results in practice, show that adoption is greatly hindered by doubts about the app's accuracy due to its use of Bluetooth, a technology not designed for distance measurements, as well as concerns about its privacy properties.

The importance of privacy concerns came as a surprise considering our efforts to minimize the data used by the applications. Unfortunately, the privacy properties of these apps are understood only by experts. Moreover, the early public and heated debate about the advantages and disadvantages of centralized and decentralized apps may have exacerbated this confusion. Most users have no means to verify an app and find it difficult to believe that it will not collect data (given that this is not true for almost any other app on their phone). Digital literacy is increasingly essential to enable non-experts to actively participate in this type of public discussion.

Concerns about the accuracy and effectiveness of the app are complex to explain. Moreover, the strong privacy protection of the apps does not permit immediate collection of statistics to demonstrate its value. It is possible, however, to gather data about DCT outside of the app.⁵ Using these other means, researchers have demonstrated the effectiveness of the apps in at least three countries.^{12,16,33,40} Among others, these studies show that apps have similar second-attack rates to manual tracing, provide faster notifications than manual contact tracing for users who do not live in the same household, and reach wider circles than contacts manually reported by index cases. These findings, which could build confidence in the value and effectiveness of these apps, are also difficult to communicate clearly and understandably to the general



The highly integrated design of mobile-phone platforms means that a DCT protocol must be integrated into a phone's operating system.



population. In some countries, communications so far have mainly been confined to researchers and not authorities, limiting its value in increasing DCT usage.

Looking Ahead: Paving the Way to Respectful Technology for the Next Emergency

The DP3T project is proof that it is possible to build and deploy practical, scalable, and useful privacy-preserving applications without collecting data that could be abused. At the same time, our deployment experience demonstrated the difficulty of achieving a high level of end-to-end privacy. Delivering a high assurance in the design required months of iterative effort to overcome the many practical obstacles to privacy that have their roots in today's service-oriented software engineering practices.²⁶

A large part of this difficulty stemmed from our reliance on the smartphone ecosystem, which is tightly controlled by Apple and Google. The involvement of these two giants came with notable advantages: It quickly established a de facto international standard and rapidly brought together resources from the two companies to build and deploy efficient GAEN implementations. However, their involvement meant that these two companies decided which DCT applications were permitted and how they could operate.

Two changes could make future public health deployments faster and less dependent on big tech. On the public sector side, there is a pressing need for improved independent infrastructure and software development capability. On the platform side, it is imperative that mobile application development patterns emerge that are architecturally separate from the core operating system provided by Google and Apple, so that the key control points (app delivery, update, notification, and more) are not solely under the control of operating system providers.²⁰ This does not mean removing all control points entirely, which might lead to security vulnerabilities. But having regard for security in software development and distribution does not necessarily entail giving a small number of firms the magnitude of decision-making power we currently do. It is impor-


tant for the research community, and for society, that alternatives to these proprietary platforms emerge so that future public health software can be effective, fully accountable, and auditable. The need to rethink the current landscape is increasingly subject to political attention, such as through third-party app store provisions in the EU's *Digital Markets Act*.

Despite the strong protections embedded in decentralized DCT protocols, the limited adoption of these applications in some countries hampered their efficacy. However, other countries saw reasonably high adoption levels (U.K., Finland, Netherlands, Ireland, and Germany), and there is evidence that the apps warned millions of users, in many cases faster than manual contact tracing.^{33,40} In the future, it is important to increase adoption by accelerating the collection of evidence of effectiveness with integrated, privacy-preserving metrics from the onset. However, mechanisms to compute such metrics are hard to integrate in practice.²⁶

The design principles behind DCT apps can be harnessed to build other applications to help with pandemic containment. For instance, there is growing evidence that SARS-CoV-2 can be transmitted beyond close-proximity contacts. Decentralized technologies can also be used to efficiently notify visitors of venues and events with SARS-CoV-2-positive attendees.³⁰

The same design philosophy, centered on limiting the purpose of applications, can be applied to other new technologies, especially if their effects are uncertain as in the case of pandemic-mitigation technical solutions. By following this path, we can harness the potential benefits of technology, without endangering the fundamental societal values of liberty, freedom, and the right to privacy.


Acknowledgments

The authors would like to thank all the individual researchers, companies, organizations, and public health officials that participated in the deployment and evaluation of COVID-19 DCT apps. This project has been partially funded by Fondation Botnar, Basel, Switzerland. **Carmela Troncoso** (camela.troncoso@epfl.ch) served as contact author for this article. 

References

1. Avitabile, G., Botta, V., Iovino, V., and Visconti, I. Towards defeating mass surveillance and SARS-CoV-2: The Pronto-C2 fully decentralized automatic contact tracing system. *IACR Cryptology ePrint Archive* (2020), 493. <https://eprint.iacr.org/2020/493>.
2. Bahrain, Kuwait and Norway contact tracing apps among most dangerous for privacy. Amnesty International (2020), <https://www.amnesty.org/en/latest/news/2020/06/bahrain-kuwait-norway-contact-tracing-apps-danger-for-privacy/>.
3. Barthe, G. et al. PanCast: Listening to Bluetooth beacons for epidemic risk mitigation. *CoRR abs/2011.08069* (November 2020), <https://arxiv.org/abs/2011.08069>.
4. Bay, J. et al. BlueTrace: A privacy-preserving protocol for community-driven contact tracing across borders. (2020), https://bluetrace.io/static/bluetrace_whitepaper-938063656596c104632def383eb33b3c.pdf.
5. Benzler, J. et al. Towards a common performance and effectiveness terminology for digital proximity tracing applications. *CoRR abs/2012.12927* (December 2020), <https://arxiv.org/abs/2012.12927>.
6. Briers, M., Charalambides, M., and Holmes, C. Risk scoring calculation for the current NHS contact tracing app. *CoRR abs/2005.11057* (May 2020), <https://arxiv.org/abs/2005.11057>.
7. Canetti, R. et al. Privacy-preserving automated exposure notification. *IACR Cryptology ePrint Archive* (2020).
8. Canetti, R., Trachtenberg, A., and Varia, M. Anonymous collocation discovery: Harnessing privacy to tame the coronavirus (March 2020), <https://arxiv.org/abs/2003.13670>.
9. Castelluccia, C. et al. DESIRE: A third way for a European exposure notification system leveraging the best of centralized and decentralized systems. *CoRR abs/2008.01621* (August 2020), <https://arxiv.org/abs/2008.01621>.
10. Chan, J. et al. PACT: Privacy sensitive protocols and mechanisms for mobile contact tracing. *CoRR abs/2004.03544* (April 2020), <https://arxiv.org/abs/2004.03544>.
11. CWA Team. Epidemiological motivation of the transmission risk level. (October 2020), https://github.com/corona-warn-app/cwa-documentation/blob/main/transmission_risk.pdf.
12. Daniere, P., Ballouz, T., Menges, D., and von Wyl, V. The SwissCovid digital proximity tracing app after one year: Were expectations fulfilled? *Swiss Medical Weekly* (September 2021).
13. Data protection and information security architecture. PEPP-PT (Apr. 2020), <https://github.com/pepp-pt/pepp-pt-documentation/blob/master/10-data-protection/PEPP-PT-data-protection-information-security-architecture-Germany.pdf>.
14. DP-3T Team. Privacy and security risk evaluation of digital proximity tracing systems (April 2020), <https://github.com/DP-3T/documents/blob/master/Security%20Analysis/Privacy%20and%20Security%20Attacks%20on%20Digital%20Proximity%20Tracing%20Systems.pdf>.
15. DP-3T Team. Secure upload authorisation for digital proximity tracing (April 2020), <https://github.com/DP-3T/documents/blob/master/DP3T%20-%20Upload%20Authorisation%20Analysis%20and%20Guidelines.pdf>.
16. Ebbens, W., Hoof, L., van der Laan, N., and Metting, E. Evaluation CoronaMelder: An overview after 9 months. https://www.coronamelder.nl/media/Evaluatie_CoronaMelder_na_9_maanden_english.pdf.
17. European Interoperability Certificate Governance. A security architecture for contact tracing and warning apps. *eHealth Network* (April 2020), https://health.ec.europa.eu/publications/european-interoperability-certificate-governance-security-architecture-contact-tracing-and-warning_en.
18. Exposure notification—Cryptography specification. Google LLC and Apple Inc. (April 2020), https://blog.google/documents/69/Exposure_Notification_-_Cryptography_Specification_v1.2.1.pdf.
19. German restaurants object after police use COVID data for crime-fighting. *Reuters* (July 2020), <https://www.reuters.com/article/us-health-coronavirus-germany-privacy-idUSKCN24W2K6>.
20. Groschupp, F. et al. Sovereign smartphone: To enjoy freedom we have to control our phones. *arXiv:2102.02743* (Feb. 2021).
21. Hargittai, E., Redmiles, E.M., Vitak, J., and Zimmer, M. Americans' willingness to adopt a COVID-19 tracking app. *First Monday* 25, 11 (Oct. 2020).
22. Hatke, G.F., et al. Using Bluetooth Low Energy (BLE)

- signal strength estimation to facilitate contact tracing for COVID-19. *Pact Technical Report* (2020).
23. Illmer, A. Singapore reveals Covid privacy data available to police. *BBC* (Jan. 2021), <https://www.bbc.com/news/world-asia-55541001>.
24. Kaptchuk, G. et al. How good is good enough for COVID19 apps? The influence of benefits, accuracy, and privacy on willingness to adopt. *arXiv:2005.04343* (May 2020).
25. Kim, N. 'More scary than coronavirus': South Korea's health alerts expose private lives. *The Guardian* (March 2020), <https://www.theguardian.com/world/2020/mar/06/more-scary-than-coronavirus-south-koreas-health-alerts-expose-private-lives>.
26. Kostova, B., Gürses, S., and Troncoso, C. Privacy engineering meets software engineering. On the challenges of engineering Privacy By Design. (July 2020), *arXiv:2007.08613*.
27. Leith, D.J. and Farrell, S. Coronavirus contact tracing: Evaluating the potential of using Bluetooth received signal strength for proximity detection. *Computer Communication Review* 50, 4 (October 2020), 66-74.
28. Levy, I. High level privacy and security design for NHS COVID-19 contact tracing app. National Cyber Security Centre, U.K. (May 2020), <https://www.ncsc.gov.uk/files/NHS-app-security-paper%20V0.1.pdf>.
29. Lovett, T. et al. Inferring proximity from Bluetooth Low Energy RSSI with unscented Kalman smoothers. *CoRR abs/2007.05057* (July 2020), <https://arxiv.org/abs/2007.05057>.
30. Lueks, W. et al. CrowdNotifier: Decentralized privacy-preserving presence tracing. In *Proceedings on Privacy Enhancing Techniques* 2021, 4 (July 2021), 350-368.
31. Rivest, R.L. et al. The PACT Protocol technical specification. PACT (April 2020), <https://pact.mit.edu/wp-content/uploads/2020/11/The-PACT-protocol-specification-2020.pdf>.
32. ROBERT: ROBUst and privacy-presERVing proximity Tracing. Inria PRIVATICUS team and Fraunhofer AISEC (April 2020), https://github.com/ROBERT-proximity-tracing/documents/blob/master/previous_versions/ROBERT-specification-EN-v1_0.pdf.
33. Salathé, M. et al. Early evidence of effectiveness of digital contact tracing for SARS-CoV-2 in Switzerland. *Swiss Medical Weekly* 150, 2020/5153 (December 2020).
34. Sterling, T. Personal data stolen from Dutch coronavirus track-and-trace programme. *Reuters* (2021), <https://www.reuters.com/article/us-health-coronavirus-netherlands-datapr-idUSKBN29Y1H3>.
35. SwissCovid Exposure Score Calculation. <https://github.com/admin-ch/PT-System-Documents/blob/master/SwissCovid-ExposureScore.pdf>.
36. TCN Coalition. TCN Protocol (2020), <https://github.com/TCNCoalition/TCN>.
37. Trieu, N. et al. Epione: Lightweight contact tracing with strong privacy. *IEEE Data Engineering Bulletin* 43, 2 (2020), 95-107. <http://sites.computer.org/debull/A20june/p95.pdf>.
38. Troncoso, C., et al. Decentralized privacy-preserving proximity tracing. *CoRR abs/2005.12273* (2020), <https://arxiv.org/abs/2005.12273>.
39. White, N. Creepy bartender uses coronavirus contact tracing data to ask out a girl he gave a free drink to—as Australians are warned their personal information could be misused or stolen. *Daily Mail Australia* (July 2020), <https://www.dailymail.co.uk/news/article-8516533/Creepy-bartender-uses-coronavirus-contact-tracing-data-ask-girl.html>.
40. Wymant, C. et al. The epidemiological impact of the NHS COVID-19 App. *Nature* 594, 7863 (2021), 408-412.
41. Zhao, Q., Wen, H., Lin, Z., Xuan, D., and Shroff, N.B. On the accuracy of measured proximity of Bluetooth-based contact tracing apps. In *Security and Privacy in Communication Networks* 335, Springer (December 2020), 49-60.

 This work is licensed under a <http://creativecommons.org/licenses/by/4.0/>



Watch the authors discuss this work in the exclusive *Communications* video. <https://caem.acm.org/videos/proximity-tracing>

DOI:10.1145/3513259

A rigorous journey from the bakery algorithm to a distributed state machine.

BY LESLIE LAMPORT

Deconstructing the Bakery to Build a Distributed State Machine

IN THIS ARTICLE, the reader and I will journey between two concurrent algorithms of the 1970s that are still studied today. The journey begins at the bakery algorithm⁹ and ends at an algorithm for implementing a distributed state machine.¹² I hope we enjoy the voyage and perhaps even learn something.

The bakery algorithm ensures processes execute a critical section of code one at a time. A process trying to execute that code chooses a number it believes to be higher than the numbers chosen by other such processes. The process with the lowest number goes first, with ties broken by process name. In the distributed state-machine algorithm, each process maintains a logical clock, with the clocks being synchronized by having a process

include its clock value in the messages it sends. Commands to the state machine are ordered according to the value of a process's clock when it issues a command, with ties broken by process name.

The similarity between the bakery algorithm's numbers and the state-machine algorithm's clocks has been noticed, but I know of no previous rigorous connection between them. Our trip makes this connection, going from the bakery algorithm to the state-machine algorithm through a sequence of algorithms, each (except the first) derived from the preceding one.

The first algorithm on the journey is a straightforward generalization of the bakery algorithm, mainly by allowing a process to read other processes' numbers in an arbitrary order. We then deconstruct this algorithm by having each process maintain multiple copies of its number, one for each other process. Next is a distributed version of the deconstructed algorithm obtained by having each copy of a process i 's number kept by the process that reads it, where i writes the value stored at another process by sending a message to that process. We then modify this distributed algorithm to ensure that numbers increase with each execution of the critical section. Finally, we arrive at the distributed state-machine algorithm by forgetting about critical sections and just using the numbers as logical clocks.

Not only do our algorithms date from the 1970s, but the path between them is one that could have been followed at that time. The large amount of related work done since then has neither influenced nor obviated any part of the route. At the end of our journey, a concluding section discusses that related work and why the algorithms that begin and end our path are still studied today. The correctness proofs in our journey are informal, much as they would have been in the 1970s. More modern, rigorous proofs are discussed in the concluding section.

A62



A61



A60



A59

A57



A56

The Original Bakery Algorithm


The bakery algorithm solves the mutual-exclusion problem introduced and solved by Edsger Dijkstra.³ The problem assumes a set of processes that alternate between executing a noncritical and a critical section of code. A process must eventually exit the critical section, but it may stay forever in the noncritical section. The basic requirement is that, at most, one process can be executing the critical section at any time. A solution to the mutual-exclusion problem lies at the heart of almost all multiprocess programming.

The bakery algorithm assumes processes are named by numbers from 1 through N . Figure 1 contains the code for process number i , almost exactly as it appeared in the original paper. The values of the variables *number* and *choosing* are arrays indexed by process number, with $number[i]$ and $choosing[i]$ initially equal to 0 for every process i . The relation \ll is lexicographical ordering on pairs of numbers, so $(1, 3) \ll (2, 2) \ll (2, 4)$; it is an irreflexive total ordering on the set of all pairs of integers.


Mutual exclusion can be achieved very simply by not allowing any process to ever enter the critical section. A mutual-exclusion algorithm needs to also satisfy some progress condition. The condition Dijkstra's algorithm satisfies is *deadlock freedom*, meaning that if one or more processes try to enter the critical section, one of them must succeed. Most later algorithms satisfy the stronger requirement of *starvation freedom*, meaning that every process that tries to enter the critical section eventually does so. Before discussing mutual exclusion, we show that the bakery algorithm is starvation free. But first, some terminology.

We say that a process is *in the doorway* when it is executing statement M . After it finishes executing M until it exits its critical section, we say that it is *inside the bakery*. When it is at any other place in its code, we say that it is *outside the bakery*.

We first show that the algorithm is deadlock free. If it weren't, it would eventually reach a state in which every process is either forever in its noncritical section or forever inside the bakery. Eventually, $choosing[i]$ would



The bakery algorithm ensures processes execute a critical section of code one at a time.



equal 0 for all i , so every process inside the bakery would be waiting forever at statement $L3$. But this is impossible because the waiting process i with the smallest value of $(number[i], i)$ would eventually enter the critical section. Hence, the algorithm is deadlock free.

To show that the algorithm is starvation free, it suffices to obtain a contradiction by assuming that a process i remains forever inside the bakery and outside the critical section. By deadlock freedom, other processes must continually enter and leave the critical section, since they cannot halt there.

However, once a process j is outside the bakery, to enter the bakery again it must execute statement M and set $number[j]$ to be greater than $number[i]$. At that point, process j must remain forever inside the bakery because it will loop forever if it reaches $L3$ with $k = i$. Eventually, $number[i]$ will be less than $number[j]$ for every process j in the bakery, so i will enter its critical section. This is the contradiction that proves starvation freedom.

Essentially, the same proof shows that the other mutual-exclusion algorithms we derive from the bakery algorithm also satisfy starvation freedom. So, we will say little more about starvation freedom. We now explain why the bakery algorithm satisfies mutual exclusion. For brevity, we abbreviate $(number[i], i) \ll (number[j], j)$ as $i \ll j$.

Here is a naive proof that i and j cannot both be in their critical sections at the same time. For i to enter the critical section, it must find $number[j] = 0$ or $i \ll j$ when executing $L3$ for $k = j$. Similarly, for j to enter the critical section, it must find $number[i] = 0$ or $j \ll i$ when executing $L3$ for $k = i$. Since a process's number is non-zero when it executes $L3$, this means that for i and j both to be in their critical sections, $i \ll j$ and $j \ll i$ must be true, which is impossible.

This argument is flawed because it assumes that both i and j were inside the bakery when the other process executed $L3$ for the appropriate value of k . Suppose process i read $number[j]$ while j was in the doorway (executing M) but had not yet set $number[j]$. It is possible for j to have read $number[i] = 0$ in $L3$ and entered the critical section, and for i then to have chosen $number[i]$ to make $i \ll j$ and entered the critical section.

The flaw in the argument is correct-

ed by statement $L2$. Since $choosing[j]$ equals 1 when j is in the doorway, process i executed $L3$ after $L2$ found that j was not in the doorway; similarly, j executed $L3$ after finding i not in the doorway. If, in both cases, the two processes were inside the bakery when $L2$ was executed, then the naive argument is correct. If one of them, say j , was not inside the bakery, it must have been outside the bakery. Since i was then inside the bakery, with its current value of $number[i]$, process j must have chosen $number[j]$ to be greater than the current value of $number[i]$, making $i \ll j$ true. Hence, j could not have exited the $L3$ loop for $k = i$ and entered the critical section while i was still in the bakery. Therefore, i and j cannot both be in the critical section.

Observe that the $choosing$ variable serves only to ensure that, when process i executes $L3$ for $k = j$, there had been an instant when i was already inside the bakery and j was not in the doorway. This will be important later.

The most surprising property of the bakery algorithm is that it does not require reading or writing a memory register to be an atomic action. Carefully examining the proof of mutual exclusion shows that it just requires that $number[i]$ and $choosing[i]$ are what were later called safe registers,¹³ ensuring only that a read not overlapping a write obtains the current register value. A read that does overlap a write can obtain any value the register might contain.

It is most convenient to describe a safe register in terms of atomic actions. We represent writing a value v to the register as two actions: the first sets its value to a special constant ζ and the second sets it to v . We represent a read as a single atomic action that obtains the value of the register if that value does not equal ζ . A read of $number[i]$ when it equals ζ can return any natural number, and a read of $choosing[i]$ when it equals ζ can return 0 or 1.

Generalization of the Original Algorithm

Two generalizations of the bakery algorithm were obvious when it was published. The first is that, in statement M , it is not necessary to set $number[i]$ to $1 + maximum(\dots)$. It could be set to any number greater than that maximum. It can also be set to the maximum if that

makes $(number[j], j) \ll (number[i], i)$ for all j , but we will not bother with that generalization. We rewrite statement M using \succ to mean “is assigned a value greater than.”

The second obvious generalization is that statements $L2$ and $L3$ for different values of k do not have to be executed in the order specified by the **for** statement. Since the proof of mutual exclusion considers each pair of processes by themselves, the only requirement is that, for any value of k , statement $L2$ must be executed before $L3$. For different values of k , those statements can be executed concurrently by different subprocesses. Also, there is no reason to execute them for $k = i$ because their **if** tests always equal false.

These two generalizations have appeared elsewhere.^{5,10} There is another, less obvious generalization that seems to be new: The assignment of 0 to $number[i]$ after the process leaves

the critical section does not need to be completed before the process enters the noncritical section. In fact, that assignment need not even be completed if the process leaves the noncritical section to enter its critical section again. As long as that assignment is completed or aborted (leaving the register equal to ζ) before $number[i]$ is assigned a new value in statement M , it just appears to other processes as if process i is still in the critical section or is executing the assignment statement immediately after the critical section. Therefore, mutual exclusion is still satisfied. To maintain starvation freedom, the write of 0 must eventually be completed if i remains forever in the noncritical section. There seems to be no simple way to describe in pseudo-code these requirements for setting $number[i]$ to 0 upon completing the critical section. We simply add the mysterious keyword **asynchro-**

Figure 1. Process i of the original bakery algorithm.

```

begin integer k ;
L1: noncritical section ;
    choosing[i] := 1 ;
M: number[i] := 1 + maximum(number[1], ..., number[N]) ;
    choosing[i] := 0 ;
    for k = 1 step 1 until N do
        begin
            L2: if choosing[k] ≠ 0 then goto L2 ;
            L3: if number[k] ≠ 0 and (number[k], k) ≪ (number[i], i)
                then goto L3 ;
        end ;
    critical section ;
    number[i] := 0 ;
    goto L1
end
    
```

Figure 2. A generalization of the original bakery algorithm.

```

process i in {1, ..., N}
    variables number[i] = 0, choosing[i] = 0 ;
    while true do
        noncritical section ;
        choosing[i] := 1 ;
M: number[i] ≻ maximum(number[1], ..., number[N]) ;
        choosing[i] := 0 ;
        process j ≠ i in {1, ..., N}
            L2: await choosing[j] = 0 ;
            L3: await (number[j] = 0) ∨ ((number[i], i) ≪ (number[j], j))
        end process ;
        critical section ;
        asynchronously number[i] := 0 ; see explanation in text
    end while
end process
    
```

nously and refer to this discussion for its explanation.

The generalized algorithm is in Figure 2. Processes are explicitly declared; the outer **process** statement indicates that there are processes numbered from 1 through N and shows the code for process number i . Variables are declared with their initial values. The inner **process** statement declares that process i has $N - 1$ subprocesses j with numbers from 1 through N , with none numbered i , and gives the code for subprocess j . That statement is executed by forking the subprocesses and continuing to the next statement (the critical section) when all subprocesses have terminated. Harmful or not, **gotos** have been eliminated. The outer loop is described as a **while** statement. The loops at $L2$ and $L3$ have been described with **await** statements, each of which repeatedly evaluates its predicate and terminates when it is true. The $>$ in statement M and the **asynchronously** statement are explained above.

The Deconstructed Bakery Algorithm

We have assumed that $number[i]$ and $choosing[i]$ are safe registers, written only by i and read by multiple readers. Such a register is easily implemented with safe registers having a single reader by keeping a copy of the register's value in a separate register for each reader.

We deconstruct the generalized bakery algorithm by implementing the safe registers $choosing[i]$ and $number[i]$ with single-reader registers $localCh[j][i]$ and $localNum[j][i]$, for each $j \neq i$. Note the counterintuitive subscript order, with $localCh[j][i]$ and $localNum[j][i]$ containing the copies of $choosing[i]$ and $number[i]$ read by process j .

The pseudo-code of the deconstructed algorithm is in Figure 3. The reads of $choosing[j]$ and $number[j]$ by process i in the generalized algorithm are replaced by reads of $localCh[j][i]$ and $localNum[j][i]$. The variable $number[i]$ is now read only by process i , and we have eliminated $choosing[i]$ because process i never reads it. Ad hoc notation is used in statement M to indicate that $number[i]$ is set to be greater than the values of all $localNum[j][i]$.

Figure 3. The deconstructed bakery algorithm.

```

process  $i$  in  $\{1, \dots, N\}$ 
  variables  $number[i] = 0, localNum[*][i] = 0, localCh[*][i] = 0$  ;
  while true do
    noncritical section ;
    process  $j \neq i$  in  $\{1, \dots, N\}$ 
       $localCh[j][i] := 1$ 
    end process ;
     $M: number[i] := \mathbf{any} \ v > 0 \ \mathbf{with} \ \forall j \neq i : v > localNum[i][j]$  ;
     $localNum[*][i] := i$  ;
    process  $j \neq i$  in  $\{1, \dots, N\}$ 
       $localNum[j][i] := number[i]$  ;
       $localCh[j][i] := 0$  ;
       $L2: \mathbf{await} \ localCh[i][j] = 0$  ;
       $L3: \mathbf{await} \ (localNum[i][j] = 0) \vee ((number[i], i) \ll (localNum[i][j], j))$ 
    end process ;
    critical section ;
     $number[i] := 0$  ;
     $localNum[*][i] := i$  ;
    asynchronously process  $j \neq i$  in  $\{1, \dots, N\}$  see explanation in text
       $localNum[j][i] := 0$ 
    end process
  end while
end process
    
```

We have explicitly indicated the two atomic actions that represent writing a value v to the safe register $localNum[j][i]$, first setting its value to i and then to v . We have not bothered to do that for the writes to $localCh[j][i]$. The $localCh[j][i]$ and $localNum[j][i]$ writes are performed by subprocesses of process i , except that the $N - 1$ separate writes of i to all the registers $localNum[j][i]$ are represented by an assignment statement

$$localNum[*][i] := i$$

of the main process i . (This will be more convenient for our next version of the bakery algorithm.) To set $number[i]$ to 0 after i exits the critical section, all the registers $localNum[j][i]$ are set to i by the main process, and each is set to 0 by a separate process. We require that the setting of $localNum[j][i]$ to 0 has been either completed or aborted when $localNum[j][i]$ is set to $number[i]$ by subprocess (i, j) . Again, this is not made explicit in the pseudo-code.

A proof of correctness for the deconstructed algorithm can be obtained by simple modifications to the proof for the original algorithm. For the original algorithm, we defined process i to be in the doorway while executing statement M , which ended with assigning the value of $number[i]$.

Since $number[i]$ has been replaced by the registers $localNum[j][i]$, process i now has a separate doorway for each other process j . We say that i is in the doorway with respect to j from when it begins executing statement M until its subprocess j assigns $number[i]$ to $localNum[j][i]$. We say that i is inside the bakery with respect to j from when it leaves the doorway with respect to j until it exits the critical section. The definition of i outside the bakery is the same as before.

To transform the proof of correctness of the original bakery algorithm to a proof of correctness of the deconstructed algorithm, we replace every statement that i or j is in the doorway or inside the bakery with the statement that it is there with respect to the other process. The modified proof shows that the function of statement $L2$ is to ensure some time between i coming inside the bakery with respect to j and executing $L3$ for j , process j was not in the doorway with respect to i .

The Distributed Bakery Algorithm

We now implement the deconstructed bakery algorithm with a distributed algorithm. Each main process i is executed at a separate node, which we call node i , in a network of processes

that communicate by message passing. The variable $localNum[j][i]$, which is process j 's copy of $number[i]$, is kept at node j . It is set by process i to the value v by sending the message v to j . The setting of $localNum[j][i]$ to i in the deconstructed bakery algorithm is implemented by the action of sending that message, and $localNum[j][i]$ is set to v by process j when it receives the message. Thus, we are implementing the deconstructed algorithm by having process j obtain a previous value of $localNum[j][i]$ on a read when $localNum[j][i]$ equals i . Since the deconstructed algorithm allows such a read to obtain any value, this is a correct implementation.

For now, we assume that process i can write the value of $localCh[j][i]$ atomically by a magical action at a distance. We will remove this magic later.

We assume that messages sent from a process i to any other process j are received in the order that they are sent. We represent the messages in transit from i to j by a first-in, first-out (FIFO) queue $q[i][j]$. We let \emptyset be the empty queue, and we define the following operations on a queue Q :

- ▶ $Append(Q, val)$ appends the element val to the end of Q .
- ▶ $Head(Q)$ is the value at the beginning of Q .
- ▶ $Behead(Q)$ removes the element at the beginning of Q .
- ▶ $Head(Q)$ and $Behead(Q)$ are undefined if Q equals \emptyset .

The complete algorithm is in Figure 4. The shading highlights uses of $localCh$, whose magical properties need to be dealt with. Along with the main process i , there are concurrently executed processes (i, j) at node i , for each $j \neq i$. Process (i, j) receives and acts upon the messages sent to i by j .

The main process i of the distributed algorithm is obtained directly from the deconstructed algorithm by replacing the assignments of i to each $localNum[j][i]$ with the sending of a message to j , except for two changes. The first is that statement M and the following sending of messages to other processes (represented by appending $number[i]$ to all the message queues $q[i][j]$) have been made a single atomic action. We can do this because we can view the end of each message queue $q[i][j]$, onto which messages are

appended, to be part of process i 's local state. A folk theorem⁴ says that, for reasoning about a multiprocess algorithm, we can combine any number of actions that access only a process's local state into a single atomic action. That folk theorem has been formalized in a number of results starting with one by Lipton,¹⁵ and perhaps the most directly applicable being Lamport.¹⁴ In our algorithm, making this action appear atomic just requires preventing other processes at node i from acting on any incoming messages while the action is being executed.

The other significant change to the deconstructed algorithm is that the **asynchronously** statement has disappeared. The setting of $localNum[j][i]$ is performed by the receipt of messages sent by i to j . FIFO message de-

livery ensures that it is set to 0 before its subsequent setting to a non-zero value. Also, since $localNum[j][i]$ is now set by process (j, i) upon receipt of the message, the assignment to it in subprocess j of i has been removed.

Correctness of the deconstructed algorithm also depends on the assignment to $localNum[j][i]$ being performed before process i sets $localCh[j][i]$ to 0. Since the assignment to $localNum[j][i]$ is now performed at node j , the ordering of those two operations is no longer trivially implied by the code. To maintain that ordering, subprocess j of i must learn that process (j, i) has set $localNum[j][i]$ to $number[i]$ before it can set $localCh[j][i]$ to 0. This is done by having (j, i) send a message to i with some value ack that is not a natural number. Process (j, i) sets the value of

Figure 4. The Distributed Bakery Algorithm, with magic.

```

process  $i$  in  $\{1, \dots, N\}$ 
variables  $number[i] = 0, localNum[*][i] = 0, localCh[*][i] = 0,$ 
           $ackRcvd[i][*] = 0, q[*][i] = \phi;$ 
while true do
    noncritical section ;
    process  $j \neq i$  in  $\{1, \dots, N\}$ 
         $localCh[j][i] := 1$ 
    end process ;
    atomic  $M: number[i] := \mathbf{any} \ v > 0 \ \mathbf{with} \ \forall j \neq i : v > localNum[i][j];$ 
           $Append(q[i][*], number[i])$ 
    end atomic ;
    process  $j \neq i$  in  $\{1, \dots, N\}$ 
         $L0: \mathbf{await} \ ackRcvd[i][j] = 1;$ 
           $localCh[j][i] := 0;$ 
         $L2: \mathbf{await} \ localCh[i][j] = 0;$ 
         $L3: \mathbf{await} \ (localNum[i][j] = 0) \vee (number[i], i) \ll (localNum[i][j], j)$ 
    end process ;
    critical section ;
     $ackRcvd[i][*] := 0;$ 
     $number[i] := 0;$ 
     $Append(q[i][*], 0)$ 
    end while
end process

process  $i, j \neq i$  in  $\{1, \dots, N\}$ 
while true do
    atomic  $\mathbf{await} \ q[j][i] \neq \phi;$ 
        if  $Head(q[j][i]) = ack$ 
            then  $ackRcvd[i][j] := 1$ 
            else  $localNum[i][j] := Head(q[j][i]);$ 
                if  $Head(q[j][i]) \neq 0$  then  $Append(q[j][i], ack)$ 
                end if
            end if ;
         $Behead(q[j][i])$ 
    end atomic ;
end while
end process
    
```

$localNum[j][i]$ and sends the *ack* message to i as a single atomic action. When process (i, j) at node i receives the *ack* message, it sets $ackRcvd[i][j]$ to 1 to notify subprocess j of process i that the *ack* has arrived. The setting of $localNum[j][i]$ to $number[i]$ in the deconstructed algorithm is replaced by statement $L0$ that waits for $ackRcvd[i][j]$ to equal 1.

The rest of the code for the main process i is the same as that of the corresponding process of the deconstructed algorithm, except that after i leaves the critical section, the asynchronous setting of all the registers $localNum[j][i]$ to 0 is replaced by sending the message 0 to all the processes j , and $ackRcvd[i][j]$ is reset to 0 for all j .

The asynchronously executed process (i, j) receives messages sent by j via $q[j][i]$. For an *ack* message, it sets $ackRcvd[i][j]$ to 1; for a message with a value of $number[j]$ it sets $localNum[i][j]$ and, if the value is non-zero, sends an *ack* to j .

The one remaining problem is the magical atomic reading and writing of the register $localCh[i][j]$. The value of that register is used only in statement $L2$. The purpose of $L2$ is to ensure that, before the execution of $L3$, there existed a time T when i was in the bakery with respect to j and j was not in the doorway with respect to i . We now show that statement $L2$ is unnecessary, because executing $L0$ ensures the existence of such a time T .

The execution of statement M by j and the sending of $number[j]$ in a message to i are part of a single atomic action, and j enters the bakery with respect to i when that message is received at node i . Therefore, j is in the doorway with respect to i exactly when there is a message with a non-zero integer in $q[j][i]$. Let's call that message a *doorway* message. Process i enters the bakery with respect to j when its message containing $number[i]$ is received at node j , an action that appends to $q[j][i]$ the *ack* that $L0$ is waiting to arrive. If there is no doorway message in $q[j][i]$ at that time, then immediately after execution of that action is the time T whose existence we need to show, since it occurred before the receipt of the *ack* that $L0$ was waiting for. If there is a doorway message in $q[j][i]$, then the required time T is right after that message was received at node i . Because of FIFO

message delivery, that time was also before the receipt of the *ack* that $L0$ is waiting for. In both cases, executing $L0$ ensures there was some time T after i entered inside the bakery with respect to j when j was not in the doorway with respect to i . Hence, statement $L2$ is redundant.

Because $L2$ is the only place where the value of $localCh[i][j]$ is read, we can eliminate $localCh$ and all statements that set it. Removing all the grayed statements in Figure 4 gives us the distributed bakery algorithm, with no magic.

The first paper devoted to distributed mutual exclusion was apparently that of Ricart and Agrawala.¹⁹ Their algorithm can be viewed as an optimization and simplification of our algorithm. It delays the sending of *ack* messages in such a way that a process can enter its critical section when it receives an *ack* from every other process, so it does not have to keep track of other processes' numbers. The number 0 messages sent upon exiting the critical section can therefore be eliminated, yielding an algorithm with fewer messages. Although nicer than our algorithm, the Ricart-Agrawala algorithm is not directly on the path we are traveling.

A Distributed State Machine

In a distributed state machine,¹² there is a set of processes at separate nodes in a network, each wanting to execute state-machine commands. The processes must agree on the order in which all the commands are executed. To execute a command, a process must know the entire sequence of preceding commands.

A distributed mutual-exclusion algorithm can be used to implement a distributed state machine by having a process execute a single command in the critical section. The order in which processes enter the critical section determines the ordering of the commands. It is easy to devise a protocol that has a process in its critical section send its current command to all other processes, which order it after all preceding commands. Starting with this idea and the distributed bakery algorithm, we will obtain the distributed state-machine algorithm¹² by eliminating the critical section.

The bakery algorithm is based on

the idea that if two processes are trying to enter the critical section at about the same time, then the process i with the smaller value of $(number[i], i)$ enters first. We now make that true no matter when the two processes enter the critical section. Define a version of the bakery algorithm to be *number-ordered* if it satisfies this condition: If process i enters the critical section with $number[i] = n_i$ and process j later enters the critical section with $number[j] = n_j$, then $(n_i, i) \ll (n_j, j)$. We now make the distributed bakery number-ordered. We can do that because we have generalized the bakery algorithm to set $number[i]$ to any number greater than the maximum value of the values of $number[j]$ it reads, not just to the next-largest number.

We add to the distributed bakery algorithm a variable $maxNum$, where $maxNum[i][j]$ is the largest value $localNum[i][j]$ has equaled, for $j \neq i$. We let $maxNum[i][i]$ be the largest value $number[i]$ has equaled. We then make two changes to the algorithm. First, we replace statement M with the statement in Figure 5.

Second, in process (i, j) , if $localNum[i][j]$ is assigned a non-zero value, then $maxNum[i][j]$ is assigned that same value. The FIFO ordering of messages assures the new value of $maxNum[i][j]$ will be greater than its previous value. Clearly, $localNum[i][j]$ always equals $maxNum[i][j]$ or 0. The value of $number[i]$ chosen this way is therefore allowed by statement M of the distributed algorithm, so this is a correct implementation of that algorithm. We now show that it is number-ordered.

Suppose i enters the critical section with $number[i] = n_i$ and j later enters the critical section with $number[j] = n_j$. It's evident that $(n_i, i) \ll (n_j, j)$ if $i = j$, so we can assume $i \neq j$. The proof of mutual exclusion for the deconstructed algorithm shows that either (i) $(n_i, i) \ll (n_j, j)$ or (ii) j chose n_j after reading a value of $localNum[i][j]$ written after i set it to n_i . In our modified version of the distributed algorithm, j reads $maxNum[j][i]$ not $localNum[i][j]$ to set $number[j]$, and $maxNum[j][i]$ never decreases. Therefore, $(n_i, i) \ll (n_j, j)$ is true also in case (ii), so the algorithm is number-ordered.

Since the algorithm is number-ordered, we don't need the critical

Figure 5. A new version of statement M.

```

M: number[i] := maximum(maxNum[i][1], ..., maxNum[i][N]);
   maxNum[i][i] := number[i]
    
```

Figure 6. A newer version of statement M.

atomic

```

M: maxNum[i][i] := maximum(maxNum[i][1], ..., maxNum[i][N]);
   Append((maxNum[i][i], Cmd), q[i][*])
end atomic
    
```

section to implement a distributed state machine. We can order the commands by the value ($number[i]$, i) would have had when i entered the critical section to execute the command. Process i can send the command it is executing in the messages containing the value of $number[i]$ that it sends to other processes. In fact, we don't need $number[i]$ at all. When we send that message, $number[i]$ has the same value as $maxNum[i][i]$. We can eliminate everything in the main process i except the atomic statement containing statement M , which can now be written as in Figure 6, where Cmd is process i 's current command.

There is one remaining problem. Process i saves the messages containing commands that it sends and receives, accumulating a set of triples (v, j, Cmd) indicating that process j issued a command Cmd with $number[j]$ having the value v . It knows that those commands are ordered by (v, j) . However, to execute the command in (v, j, Cmd) , it has to know that it has received all commands $(w, k, Dcmd)$ with $(w, k) \ll (v, j)$. Process i knows that, for each process k , it has received all commands $(w, k, Dcmd)$ with $w \leq maxNum[i][k]$. However, suppose i has received no commands from k . How can i be sure that k hasn't sent a command in a message that i hasn't yet received? The answer is to use the distributed bakery algorithm's *ack* messages. Here's how.

For convenience, we let process i keep $maxNum[i][i]$ always equal to the maximum of the values $maxNum[i][j]$ (including $j = i$). It does this by increasing $maxNum[i][i]$, if necessary, when receiving a message with the value of $maxNum[i][j]$ from another process j . Upon receiving a message (v, Cmd) from process j , process i sets

$maxNum[i][j]$ to v (possibly increasing $maxNum[i][i]$) and sends back to j the message $(maxNum[i][i], ack)$. When that message is received, j sets $maxNum[j][i]$ accordingly, (increasing $maxNum[j][j]$ if necessary). When i has received all the *ack* messages for a command it issued with $maxNum[i][i]$ equal to v , all its values of $maxNum[i][j]$ will be $\geq v$, so process i knows it has received all commands ordered before its current command. It can therefore execute all of them, in the appropriate order, and then execute its current command.

This is almost identical to the distributed state-machine algorithm,¹² where $maxNum[i][i]$ is called process i 's clock. (The sketch of the algorithm given there is not detailed enough to mention the other registers $maxNum[i][j]$.) The one difference is that, when process i receives a message from j with a new value v of $maxNum[i][j]$, the algorithm requires $maxNum[i][i]$ to be set to a value $> v$, whereas $\geq v$ suffices. The algorithm remains correct if the value of $maxNum[i][i]$ increases by any amount at any time. Thus, the registers $maxNum[i][i]$ could be logical clocks that are also used for other purposes.

We have described all the pieces of a distributed state-machine algorithm but have not put them together into pseudo-code. "The precise algorithm is straightforward, and we will not bother to describe it."¹²

Ancient and Recent History

In addition to being the author of this article, I am the author of the starting and ending algorithms of our journey. The bakery algorithm is among hundreds of algorithms that implement mutual exclusion using only read and

write operations to shared memory.²² A number of them improve the bakery algorithm, the most significant improvement being a bound on the chosen numbers.^{6,21} But all improvements seem to add impediments to our path, except for one: Moses and Patin¹⁷ optimized the bakery algorithm by allowing process i to stop waiting for process j at statement $L3$ if it reads two different values of $number[j]$. However, it is irrelevant to our path because it optimizes a case that cannot occur in the distributed bakery algorithm.

Mutual-exclusion algorithms based on read and write operations have been of no practical use for decades, since modern computers provide special instructions to implement mutual exclusion more efficiently. Now, they are studied mainly as concurrent programming exercises. The bakery algorithm is of interest because it was the first mutual-exclusion algorithm not to assume lower-level mutual exclusion, which is implied by atomic reads and writes of shared memory. The distributed state-machine algorithm is interesting because it preserves causality. But it too is less important than the problem it solves.

The most important contribution of my state-machine paper was the observation that any desired form of cooperation in a network of computers can be obtained by implementing a distributed state machine. The obvious next step was to make the implementation fault tolerant. The work addressing that problem is too extensive to discuss here. Fault-tolerant state-machine algorithms have become the standard building block for implementing reliable distributed systems.²⁰

There was no direct connection between the creation of the bakery algorithm and of the state-machine algorithm. The bakery algorithm was inspired by a bakery in the neighborhood where I grew up. A machine dispensed numbers to its customers that determined the order in which they were served. The state-machine algorithm was inspired by an algorithm of Paul Johnson and Robert Thomas.⁷ They used the \ll relation and process identifiers to break ties, but I don't know if that was inspired by the bakery algorithm.

The path between the two algorithms that we followed is not the one I originally took. That journey began when I was looking for an example of a distributed algorithm for notes I was writing. Stephan Merz suggested the mutual-exclusion algorithm I had used to illustrate the state-machine algorithm. I found it to be too complicated, so I simplified it. (I did not remember the Ricart-Agrawala algorithm and was only later reminded of it by a referee). After stripping away things that were not needed for that particular state machine, I arrived at the distributed bakery algorithm. It was obviously related to the original bakery algorithm, but it was still not clear exactly how.

I wanted to make the distributed algorithm an implementation of the bakery algorithm. I started with the generalization of having subprocesses of each process interact independently with the other processes; that was essentially how I had been describing the bakery algorithm for years. Delaying the setting of $number[i]$ to 0 was required because the distributed algorithm's message that accomplished it could be arbitrarily delayed. It took me a while to realize that I should deconstruct the multi-reader register $number[i]$ into multiple single-reader registers, and that both the original bakery algorithm and the distributed algorithm implemented that deconstructed algorithm.

The path back from the distributed bakery algorithm to the distributed state-machine algorithm was easy. It may have helped that I had previously used the idea of modifying the bakery algorithm to make values of $number[i]$ keep increasing. Paradoxically, that was done to keep those values from getting too large.¹⁰

Correctness of a concurrent algorithm is expressed with two classes of properties: *safety* properties, such as mutual exclusion, that assert what the algorithm may do, and *liveness* properties, such as starvation freedom, that assert what the algorithm must do.¹ Safety properties depend on the actions the algorithm can perform; liveness properties depend as well on assumptions, often implicit, about what actions the algorithm must perform.

The kind of informal behavioral

reasoning I have used here is notoriously unreliable. I believe the best rigorous proofs of safety properties are usually based on invariants—predicates that are true of every state of every possible execution.² Invariance proofs that the bakery algorithm satisfies mutual exclusion have often been used to illustrate formalisms or tools.^{5,11} An informal sketch of such a proof for the decomposed bakery algorithm is in an expanded version of this article, which is available on the Web.⁸ Elegant rigorous proofs of progress properties can be written using temporal logic.¹⁸

Rigorous proofs are longer than informal ones and can intimidate readers not used to them. I almost never write one until I believe that what I want to prove is true. For the correctness of our algorithms, that belief was based on the reasoning embodied in the informal proofs I presented—the same kind of reasoning I used when I discovered the bakery and distributed state-machine algorithms.

I understood the two algorithms well enough to be confident in the correctness of the non-distributed versions of the bakery algorithm and of the derivation of the state-machine algorithm from the distributed bakery algorithm. Model checking convinced me of the correctness of the distributed bakery algorithm and confirmed the confidence my informal invariance proof had given me that the deconstructed algorithm satisfies mutual exclusion.

More recently, Stephan Merz wrote a formal, machine-checked version of my informal invariance proof. He also wrote a machine-checked proof that the actions of the distributed bakery algorithm implement the actions of the deconstructed bakery algorithm under a suitable data refinement. These two proofs show that the deconstructed algorithm satisfies mutual exclusion. The proofs are available on the Web.¹⁶ □

References

1. Alpern, B. and Schneider, F. Defining liveness. *Information Processing Letters* 21, 4 (Oct. 1985), 181–185.
2. Ashcroft, E. Proving assertions about parallel programs. *Journal of Computer and System Sciences* 10, 1 (Feb. 1975), 110–135.
3. Dijkstra, E. Solution of a problem in concurrent programming control. *Commun. ACM* 8, 9 (Sept. 1965), 569.
4. Harel, D. On folk theorems. *Commun. ACM* 23, 7 (July 1980), 379–389.
5. Hesselink, W. Mechanical verification of Lamport's

- bakery algorithm. *Science of Computer Programming* 78, 9 (2013), 1622–1638.
6. Jayanti, P., Tan, K., Friedland, G., and Katz, A. Bounding Lamport's bakery algorithm. In L. Pacholski and P. Ruzicka, eds., *SOFSEM 2001: 28th Conference on Current Trends in Theory and Practice of Informatics 2234, Lecture Notes in Computer Science*, Springer (2001), 261–270.
7. Johnson, P. and Thomas, R. The maintenance of duplicate data bases. Request for Comment RFC #677, NIC #31507, ARPANET Network Working Group, (January 1975).
8. Lamport, L. Online supplemental material for Deconstructing the bakery to build a distributed state machine. <http://lamport.azurewebsites.net/pubs/bakery/deconstruction.html>.
9. Lamport, L. A new solution of Dijkstra's concurrent programming problem. *Commun. ACM* 17, 8 (Aug. 1974), 453–455.
10. Lamport, L. Concurrent reading and writing. *Commun. ACM* 20, 11 (Nov. 1977), 806–811.
11. Lamport, L. Proving the correctness of multiprocess programs. *IEEE Transactions on Software Engineering SE-3*, 2 (Mar. 1977), 125–143.
12. Lamport, L. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM* 21, 7 (July 1978), 558–565.
13. Lamport, L. On interprocess communication. *Distributed Computing* 1 (1986), 77–101.
14. Lamport, L. A theorem on atomicity in distributed algorithms. *Distributed Computing* 4, 2 (1990), 59–68.
15. Lipton, R. Reduction: A method of proving properties of parallel programs. *Commun. ACM* 18, 12 (Dec. 1975), 717–721.
16. Merz, S. Online TLA+ specifications and proofs for "deconstructing the bakery to build a distributed state machine." <https://members.loria.fr/SMerz/papers/distributed-bakery.html>.
17. Moses, Y. and Patkin, K. Mutual exclusion as a matter of priority. *Theoretical Computer Science* 751 (2018), 46–60.
18. Pnueli, A. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on the Foundations of Computer Science*, IEEE (Nov. 1977), 46–57.
19. Ricart, G. and Agrawala, A. An optimal algorithm for mutual exclusion in computer networks. *Commun. ACM* 24, 1 (1981), 9–17.
20. Schneider, F. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys* 22, 4 (December 1990), 299–319.
21. Taubenfeld, G. The black-white bakery algorithm and related bounded-space, adaptive, local-spinning and FIFO algorithms. In R. Guerraoui, (Ed.), *Proceedings of Distributed Computing, 18th Intern. Conf. 3274, Lecture Notes in Computer Science*, Springer (Oct. 4, 2004), 56–70.
22. Taubenfeld, G. Concurrent programming, mutual exclusion. In M-Y Kao, (Ed.) *Encyclopedia of Algorithms—2016 Edition*, 421–425. Springer, 2016.

Leslie Lamport is the recipient of the 2013 ACM A.M. Turing Award.



This work is licensed under a <http://creativecommons.org/licenses/by/4.0/>



Watch the author discuss this work in the exclusive *Communications* video. <https://cacm.acm.org/videos/deconstructing-the-bakery>

ACM Computing Surveys (CSUR)

2018 JOURNAL
IMPACT FACTOR:
6.131

Integration of computer science and engineering knowledge



ACM Computing Surveys (CSUR) publishes comprehensive, readable tutorials and survey papers that give guided tours through the literature and explain topics to those who seek to learn the basics of areas outside their specialties. These carefully planned and presented introductions are also an excellent way for professionals to develop perspectives on, and identify trends in, complex technologies. Recent issues have covered image understanding, software reusability, and object and relational database topics.

Both surveys and tutorials must develop a framework or overall view of an area that integrates the existing literature. Frequently, such a framework exposes topics that need additional research; a good CSUR article can fill such a void, but that is not its major purpose. Basically, a CSUR article answers the questions, "What is currently known about this area, and what does it mean to researchers and practitioners?" It should supply the basic knowledge to enable new researchers to enter the area, current researchers to continue developments, and practitioners to apply the results.

For more
information
and to submit
your work,
please visit:

csur.acm.org



Association for
Computing Machinery

A survey of recent efforts to combine SDN and BC shows promising results and points to directions for future research.

BY MAJD LATAH AND KUBRA KALKAN

When SDN and Blockchain Shake Hands

RECENTLY, SOFTWARE-DEFINED NETWORK (SDN) has gained great attention from both industry and academia as a tool to achieve more dynamic control and management compared with traditional networking architectures.¹¹ However, SDN's centralized control has introduced many drawbacks such as Denial of Service (DoS) attacks and single point of failure in the control plane, as well as additional security challenges related to application and data planes, since they need to communicate and interact with the control plane. Any flaw in this interaction can be viewed as a potential threat that must be seriously considered to limit its global impacts over a centralized SDN network. For instance, a hijacked SDN application allows the attacker to insert/modify flow entries of related SDN switches,

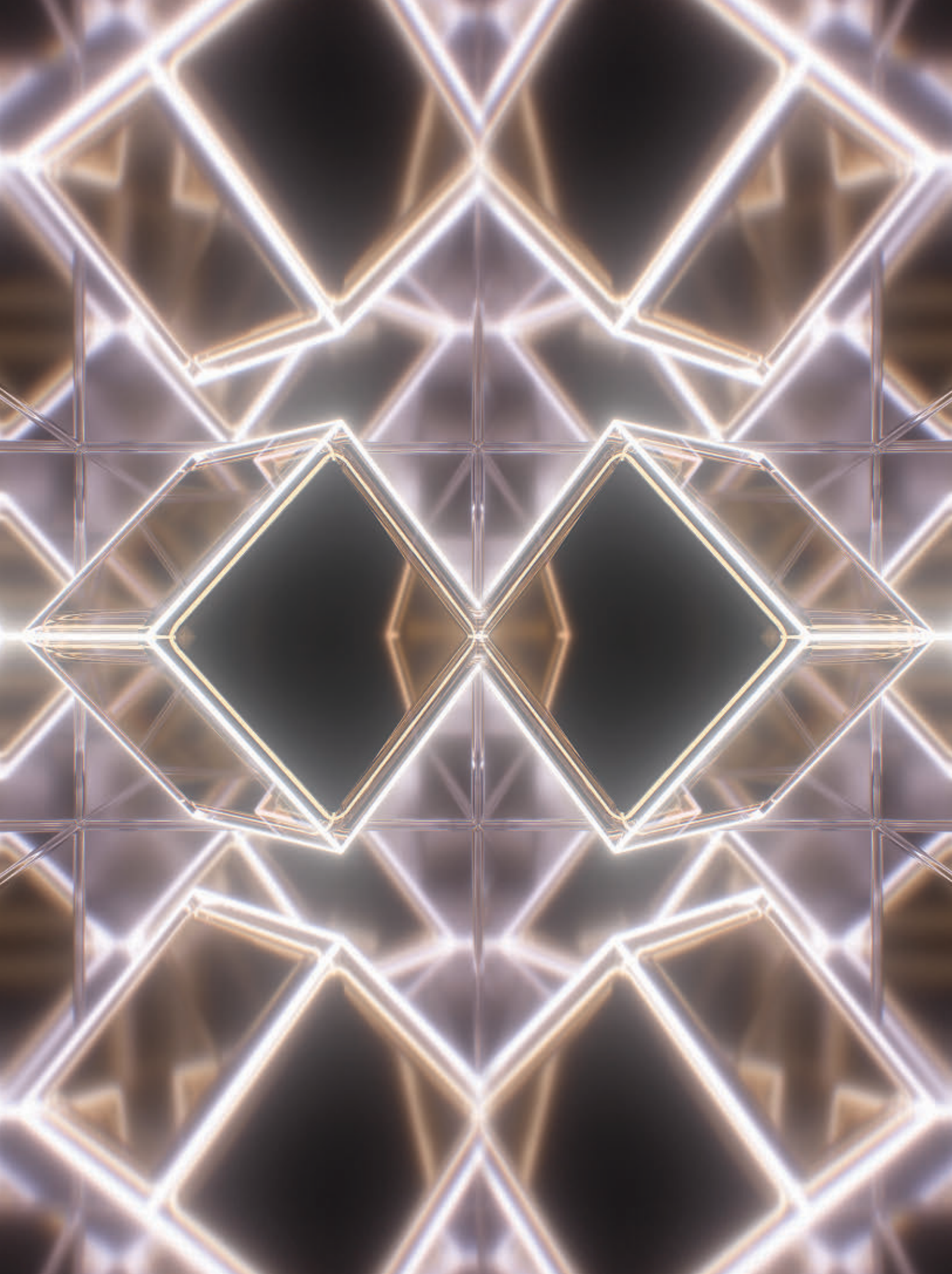
and consequently exhausts the Ternary Content-Addressable Memory (TCAM) resources. It will be more secure if we authenticate all application flows before provisioning and configuration of data plane resources.

On the other hand, a new technology can be utilized to address these challenges by introducing a fault-tolerant, decentralized control capabilities to the SDN paradigm. Blockchain is one of the most promising solutions toward trustworthy services based on a fault-tolerant, decentralized, and secure distributed ledger. It can be employed in SDN paradigm to provide both operational and structural enhancements. Blockchain (BC) enables monitoring, auditing, and autonomously taking appropriate actions once the preconditions are met. This can be accomplished at the application, the resource, and the flow levels. BC-enabled SDN provides more secure and reliable environment compared with using centralized SDN applications. Additionally, smart contracts can play a vital role in SDN by offloading proactive decision making while protecting SDN against unauthorized parties based on BC principles.

However, various attacks on the BC network layer can impact its performance, anonymity, and availability.

» key insights

- In this work, we study the integration between SDN and blockchain. We provide a fine-grained taxonomy that classifies this integration, based on utilization area, into two main categories: BC for SDN and SDN for BC.
- In BC for SDN, blockchain acts as an enabler for achieving a more secure and reliable environment compared with existing centralized SDN architecture. In SDN for BC, SDN is used to protect the underlying BC infrastructure against network-based attacks.
- We propose, BC-Sec-SDN, a hierarchical architecture that utilizes consortium BC approach and decouples the control plane into two separate BC-enabled control planes—reactive and proactive BC-SDN.



This is due to the fact the consensus layer depends on the data transmitted by the network layer. In this context, SDN can be used to protect the network layer of BC, and consequently maximize its availability and improve its resilience against potential threats including traffic analysis and DoS attacks, which aim to interrupt the data flow between BC nodes and ultimately limit their ability to reach consensus.

The integration between SDN and BC comes with additional challenges. For instance, adopting BC for securing SDN rises major issues in terms of how BC can meet the requirements of high throughput and low latency. This also may include introduction of new consensus protocols to satisfy these requirements. Moreover, considering SDN for protecting BC is also limited to network-based attacks, and potentially intruders may exploit the vulnerabilities of traditional SDNs to gain access to the underlying infrastructure, for instance, through poisoning the topology view of SDN via link spoofing attacks.²³

In this article, we investigate the integration between SDN and BC. We also show how BC can be utilized to address the limitations of both centralized and distributed SDN architectures. Then, we introduce a detailed literature survey that covers the recent research efforts on this topic. We classify these works into two main categories namely SDN for BC and BC for SDN. Then, we provide a fine-grained taxonomy based on these two categories. Following this, we introduce a BC-enabled architecture that adopts BC to meet the requirements of more protected, decentralized, and fault tolerant SDN.

Li et al.¹⁶ briefly discuss the integration among SDN and BC. They summarize and present a generic framework for BC-enabled SDN based on Dist-BlockNet.³² They also discuss the main security challenges in SDNs along with possible solutions. Our work, however, clearly shows that several frameworks with different underlying goals have been proposed in the literature,^{8,9,11,12,14,25-28,30,32,33,36-40}

In another study, Alharbi¹ summarizes the existing literature on the same topic. However, this work lacks the ap-



The SDN architecture simplifies decision making and facilitates development of new protocols as well as various SDN-tailored network applications.



propriate categorization of different proposed solutions.

In addition, this work does not discuss the limitations of BC. In contrast, our survey provides a fine-grained taxonomy that discusses the topic in a more detailed and comprehensive manner. We also highlight both the advantages and disadvantages of this integration. Furthermore, our proposed architecture decouples the control plane into two separate BC-enabled control planes namely reactive and proactive control planes. Finally, the survey suggests several directions for future research and improvement.

Software-defined networking. SDN has received considerable attention due to its central management and high flexibility. SDN replaces the local control plane in conventional hardware devices by a global controller, which programmatically manages packet forwarding on a per-flow level. The SDN architecture simplifies decision making and facilitates development of new protocols as well as various SDN-tailored network applications.

The controller manages the underlying data plane using the southbound application programming interface (API), where OpenFlow¹⁹ is the most prominent example of such an API. The northbound API, on the other hand, provides a common interface for developing SDN applications.¹⁵ An OpenFlow switch has several flow tables, each of which has matching rules along with actions that are executed when an incoming packet matches the corresponding rule. As a result, the behavior of an OpenFlow switch changes according to the rules installed by the SDN controller.¹⁵

In SDN, security is mainly integrated in the control plane, whereas mitigation of data modification/leakage takes place in the data plane.²⁵ However, SDN propounds several additional threats such as single point of failure,²⁴ DoS/Distributed DoS (DDoS) attacks,¹³ and illegal access through malicious SDN apps.³⁴ Unfortunately, these threats have a high impact that could ultimately prevent large-scale deployment of SDNs in many scenarios.³⁰ As a solution, distributed architectures can improve load distribution and avoid single controller failure. However, they come at the price of additional over-

head due to cross-controller communication as well as increased latency due to the need for synchronized and consistent state management among different SDN controllers.

Blockchain. Recently, BC has gained increasing attention due to its capability as a fundamental technology for cryptocurrencies such as Bitcoin²² and Ethereum.³⁵ Bitcoin²² is the first BC and also the most popular cryptocurrency, whereas Ethereum³⁵ represents a generalized BC platform for developing decentralized apps. BC is basically a decentralized distributed ledger that provides trustworthy services to members of a peer-to-peer network without relying on a central authority.²⁷ The decentralized distributed ledger is built and maintained through consensus among all BC members in a distributed fashion to achieve consistency while ensuring adequate fault tolerance.³⁶ BC systems aim to achieve higher level of security and reliability compared to most of non-BC systems.³ In other words, BC systems can operate in potentially hostile settings.³

In BC, a series of data blocks are generated using cryptography to be secure, immutable, and tamper-proof.³⁰ Each block contains the hash value of the previous block header, transaction data, a timestamp, and a nonce. Consequently, multiple blocks are connected to form a chain. Each new block is added through an agreement between BC nodes, thanks to consensus protocols, which define how to reach to an agreement without relying on any trusted third-party intermediaries. In general, there are three types of BC, including public, private, and consortium BCs. In public BC, every node can be involved in block creation whereas in private and consortium BCs only specific nodes can participate in this operation using a valid certificate obtained by an identity management infrastructure such as certificate authority (CA).²⁷ In public BCs, committed transactions are visible to everyone while the read permission is restricted to participating nodes in private BCs. In consortium BCs, the organization may decide whether the approved transactions are public or private.⁴¹ Private BCs are fully controlled by one organization whereas consortium BCs consist of more than one organization.

Private and consortium BCs are more efficient compared to public BCs. The fact of having fewer validators results in higher transaction throughput and lower latency.⁴¹ Hyperledger Fabric² is the most notable example for a business-oriented consortium BC. In addition, BC platforms, such as Ethereum³⁵ and Hyperledger Fabric,² provide smart contracts, which are automatically executed according to a preconfigured script code based on trusted data provided by decentralized distributed ledgers.³⁶ In this context, BC systems must maintain backward compatibility to validate earlier transactions and potentially to be able to interact with older versions of a deployed smart contract.³ BC developers may also need to test and verify the security requirements of their smart contracts before being deployed.³

Several approaches were proposed to reach consensus between BC nodes. Proof of work (PoW)²² is used in Bitcoin network. In PoW, network nodes need to solve a computationally difficult problem (that is, cryptographic puzzle) to create a new block,⁵ where 51% of computational capability is considered the threshold of PoW to gain control of the BC network.⁴¹ The PoW procedure wastes the resources of the mining nodes to be authentic.⁴¹ As an alternative to PoW, Proof of Stake (PoS) requires the proof of ownership to validate a newly generated block, considering those with more cryptocurrencies are more trustful than those with fewer cryptocurrencies.⁵ Compared with PoW, PoS saves more energy and is more effective. However, attacks might occur since the mining cost is nearly zero.⁴¹ Practical byzantine fault tolerance (PBFT)⁴ can also be utilized as a consensus algorithm and can handle up to $\frac{1}{3}$ malicious byzantine failures, where each node who votes for the consensus sends its vote to the other nodes.⁵ In each phase, a node will move to the next phase once it receives votes from $\frac{2}{3}$ of all BC nodes.⁴¹ Hyperledger Fabric² (in its version 0.6 and Hyperledger Sawtooth) employs PBFT to reach consensus among BC nodes. It is possible that valid blocks might be generated simultaneously when several nodes solve the puzzle at the same time, which results in

branches (or forks).⁵ PoW and PoS solve this issue by choosing the longest chain whereas PBFT handles this issue through multiple communication rounds. We refer the reader to Dai et al.⁵ and Zheng et al.⁴¹ for further details.

Integration Between SDN And BC

BC can address the limitations of SDN in terms of security, reliability, and single point of failure by introducing a decentralized control-plane and securely sharing critical information at different levels including the application, the resource, and the flow levels. However, this would present new challenges in terms of how BC can meet the throughput and latency requirements of SDN. On the other hand, SDN can be utilized to improve the availability and to provide access control for BC nodes enforced by network devices and SDN control plane.

Distributed architectures improve load distribution and avoid the single controller failure; however, they bring additional overhead from controller-to-controller communication and increase the latency due to the need for consistent state/policy and synchronized global network view among different SDN controllers. On the other hand, decentralized architectures have attracted the attention of the research community as a solution to address the limitations of SDN. In this regard, BC is considered a promising approach, which acts as an out-of-band party that can be utilized to achieve a consistent and synchronized global view among different SDN controllers. BC can also ensure security, dependability, and traceability requirements of SDN.²⁷ Security is required to define who has access to what, when, and in which conditions. This system should also be highly dependable in terms of reliability and availability. It also must provide an undisputable proof that cannot be fabricated or forged in possible compromises.

SDN as a networking layer for BC would provide a better protection for BC nodes. For instance, it can be used to maximize the availability of participating nodes against DoS/DDoS attacks³³ and to achieve flow-based access control.⁹ The existing integration

approaches can be classified according to many different criteria. However, the classification we have found to be the most beneficial is primarily focusing on the utilization area. This can be a useful basis for researchers seeking to identify the range of applicability, and the mutual benefits of combining these two different technologies. Therefore, we classify this integration as BC for SDN and SDN for BC. The first category improves SDN through BC, whereas the second category makes use of SDN to enhance BC. Figure 1 illustrates both SDN and BC architectures along with the major issues that can be solved by combining these two paradigms.

BC for SDN. As depicted in Figure 1, the SDN architecture consists of untrusted apps, resources, and controllers, which ultimately need to interact with each other. BC for SDN approaches make use of BC to secure traditional SDN paradigm. We categorize them into the following two groups: operational solutions that focus on enhancing the functionality of SDN, and structural solutions that provide core architectural refinements for the SDN paradigm.

Operational solutions. The operational solutions utilize BC to secure the core functionality of SDN through a cooperative defense among SDN

controllers, and to protect that collaboration by authenticating and establishing trust at the controller, the application, and the flow levels. Accordingly, these solutions can be classified into the following three sub-groups: cooperative defense, trust enhancement, and decentralized authentication.

Cooperative defense. Cooperative defense focuses on detecting sophisticated and coordinated attacks that require collaboration among different SDN controllers, which will allow a more effective mitigation near the origin of the attack and reduce the high volume of traffic exchanged across different SDN domains. BC is used to securely share critical information related to cooperative defense models and core SDN functionality such as routing among several network domains.

El Houada et al.⁸ propose a cross-domain collaborative DDoS mitigation scheme based on BC smart contracts, which allow multiple SDN-based domains to securely collaborate and transfer attack information in a decentralized manner. The collaboration is managed by the smart contract's owner who can add/remove the collaborators that report suspicious IP addresses within their domains. This approach provides a secure, low-cost, and flexible solution to mitigate

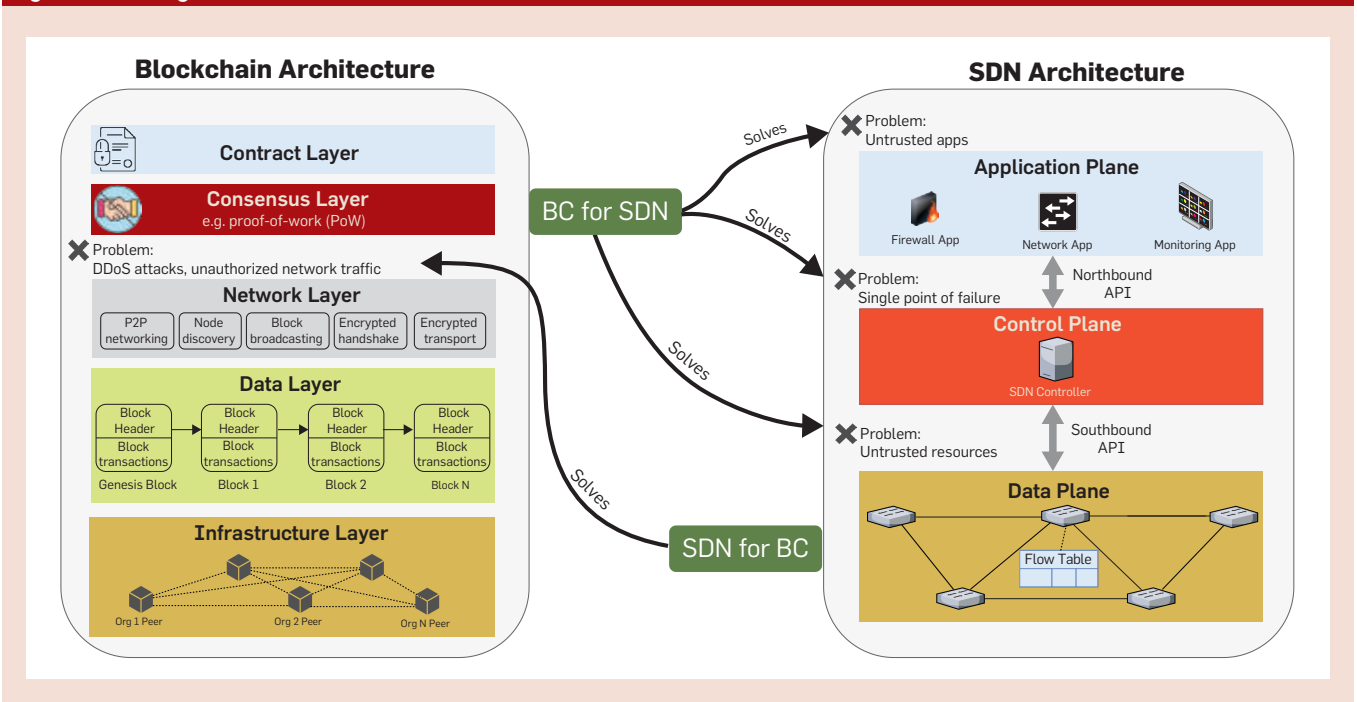
DDoS attacks in cross-domain SDNs.

Rathore et al.²⁸ introduce a decentralized and cooperative architecture for IoTs using a combination of SDN and BC as well as fog and edge computing. Attack detection is performed at the fog layer using deep learning approach. Fog nodes and a cloud server share data using BC, which is also used to regularly update the attack detection model and flow rules in the SDN switches. The cloud server acts as a manager that manages and initiates the attack detection models using BC smart contracts.

Qiao et al.²⁶ propose a BC-based approach to establish trust among SDN controllers for cross-domain routing. There are two types of transactions, the first one is related to cross-domain routing, and the other is related to network state information updating. Each transaction must be recognized by at least one node in each domain.

Cooperative defense approaches take advantage of BC to secure the information shared between SDN controllers and related applications, which removes the need to use a central entity and reduces the complexity of developing new protocols and/or modifying the existing ones. The effectiveness of these protocols also depends on global deployment and may suffer from single point of failure.⁸

Figure 1. The integration between SDN and BC.



Trust enhancement. Trust enhancement approaches make use of BC to establish trust among SDN controllers and related devices, which are connected within an SDN network. Trust management includes that each device connected to the SDN network to be registered into BC along with a trust level that changes over time based on the recent activities of that device. The advantage here is that BC ensures that trust scores stored within BC are secure, immutable, and tamperproof. Moreover, trust scores are stored through consensus among all BC nodes, which are more immune to fake information submitted by malicious devices.

Kataoka et al.¹⁴ utilize BC to prevent unsolicited traffic coming from IoT devices positioned at the edge of the network in a trustworthy, scalable, and distributed manner. To this end, they implemented a trust list across widely distributed edge networks using BC and SDN. First, SDN controllers receive the recent updates of BC, which contains profiles of validated devices. Then, they check whether the device is connected to their network. Finally, new flow rules are installed to allow that device to communicate with the server based on an existing service profile.

Zhao et al.⁴⁰ focus on protecting the control plane of SDN through a trust management architecture. More precisely, they propose a secure and efficient resource allocation for software-defined vehicular networks (SDVN). The main goal here is to protect SDVN against fake information submitted by malicious vehicles, which aim to deteriorate the quality of service (QoS) of other legitimate vehicles. A joint proof-of-stake and modified practical Byzantine fault-tolerance (PoS-mPBFT) algorithm is proposed to shorten the confirmation time and reduce the communication overhead of PBFT,⁴ where a fully trustworthy CA is used to collect, compare, and choose the leader.

Decentralized authentication. Decentralized authentication approaches utilize BC to authenticate legitimate devices that are connected to an SDN network. BC securely stores the information related to legitimate devices and reduces the re-authenti-



BC can be used as an access control method for SDN components and their associated critical assets, where only authorized entities can access the SDN network.



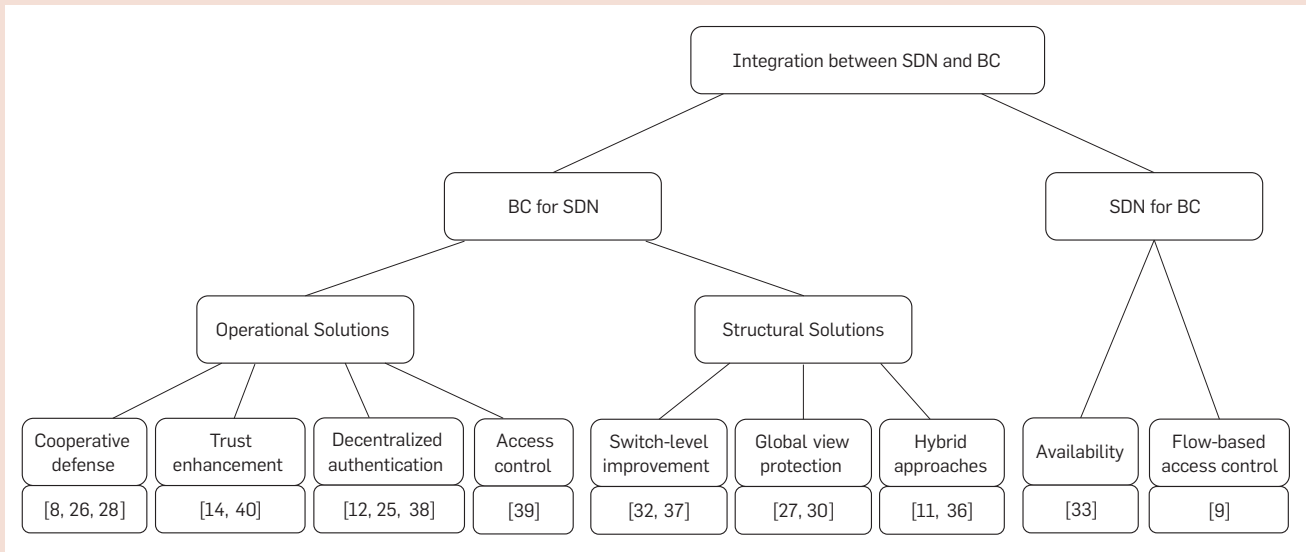
cation overhead among different SDN controllers.

Yazdinejad et al.³⁸ propose a BC-enabled handover authentication for 5G within SDN. This approach provides secure and fast authentication by employing BC, which eliminates reauthentication in repeated handovers among heterogeneous cells, in which unique characteristics of mobile user are shared between current and adjacent cells. In this work, BC manages the control plane of SDN and sends user's information to the SDN controller of that cell. Preshared-key handshakes are eliminated by using BC. In other words, the messages are approved by the BC instead of a third party. SDN, on the other hand, prepares the flow tables in accordance with a given policy.

Jindal et al. present SURVIVOR,¹² an edge-as-a-service platform that utilizes BC to provide a secure energy trading in SDN-enabled vehicle-to-grid (V2G) environments, where BC is utilized to authenticate the energy trading transactions in the system using the PoW²² approach. The study showed that the time required to generate header value and validate the transaction increases as the number of transactions increases.

Pourvahab et al.²⁵ introduce a BC-based approach for forensic purposes in SDN-based IoT environments. BC along with Linear Homomorphic Signature (LHS) algorithm are used to authenticate IoT devices based on a unique identity and Elliptic curve cryptography. An unauthenticated IoT device sends packets to the gateway, which forwards these packets to the switches and then authenticates them via the signature in SDN controllers. Event logs are used and stored on the blockchain.

Access control. BC can be used as an access control method for SDN components and their associated critical assets, where only authorized entities can access the SDN network. In addition, each entity signs related BC transactions to prove their identity. In this context, Yazdinejad et al.³⁹ employ BC as an access control method for the IoT devices and their associated data, where they presented a secure and energy-efficient BC-enabled SDN architecture for IoT networks us-

Figure 2. Taxonomy for the integration between SDN and BC.

ing a cluster structure, in which each SDN controller acts as a cluster head. The SDN controller manages several processes in each SDN domain via a private BC whereas different SDN controllers securely share data related to trusted IoT devices through a public BC. The authors replaced PoW with an authentication method to improve the performance of BC.

Structural solutions. The structural solutions take advantage of BC to enhance the structure of SDN paradigm. These solutions can be also categorized into the following groups: switch-level improvement, global view protection, and hybrid approaches. Switch-level improvement utilizes BC to enhance SDN's data plane. Global view protection shares SDN's global view through BC. Hybrid approaches apply BC technology to secure the SDN paradigm at different levels including the control and the data planes as well as the communication channel between the controller and the switch.

Switch-level improvement. The switch-level improvement approaches enhance SDN's data-plane in terms of protecting flow tables and including the ability to utilize BC in SDN switches. Sharma et al.³² present DistBlockNet, a switch-level solution, in which flow tables are verified, validated, and securely updated. The experimental results showed that DistBlockNet is capable of detecting attacks in the IoT

network in real time with low performance overheads.

Yazdinejad et al.³⁷ propose a BC-enabled packet parser (BPP) to support detecting attacks as well as BC structure in SDN's data plane. BPP supports the policies and rules defined by the control plane. When BPP in each SDN switch detects a malicious behavior, it will inform the corresponding SDN controller, and then it will report that incident to other BPPs based on P2P communication among other SDN switches.

Global view protection. The global view protection approaches focus on using BC to securely share and synchronize SDN's global view among multiple controllers. As BC provides an immutable and tamper-proof ledger, it would prevent external attackers from tampering with the global view provided that the fraction of malicious nodes does not exceed $\frac{1}{3}$ of the total number of BC nodes, assuming PBFT⁴ is used. In other words, BC is utilized as an out-of-band party to reach consensus safely and dependably among different SDN controllers, without affecting the core functionality of SDN network. Qiu et al.²⁷ use BC to collect and synchronize the global view among different SDN controllers. In this work, each controller collects its local events and OpenFlow commands and then stores them into the best BC system at the edge network.

Shao et al.³⁰ collect view related in-

formation through features (Features Reply), statistics (Stats Reply), and Packet-In messages, and then store them on the BC to protect the global view of SDN. The requests for storing the global view are verified by the other controllers. To overcome the limitations of PBFT,⁴ they propose a Simplified PBFT (SPBFT), in which the messages are transferred and verified in a parallel manner. It also prioritizes the received requests to ensure urgent requests are handled quickly and efficiently.

Hybrid approaches. The hybrid approaches apply BC technology at different levels including: the control and data planes as well as the communication channel between the controller and the switch. Yang et al.³⁶ propose BlockCtrl, a BC-based architecture to secure software-defined optical networking (SDON). In BlockCtrl, a customizable smart contract is installed in each controller, which automatically performs the network functions loaded in the smart contract. On the other hand, the SDN switches use BC smart contracts to choose the optimal secondary controller on failure. BlockCtrl achieves better service correlation and resource utilization when compared with BFT,⁴ since BlockCtrl considers the traffic data stored in the BC ledger when selecting a new primary controller.

Jiasi et al.¹¹ use BC to address multiple security issues in SDN. They

design a monolithic security mechanism for SDN based on BC, which decentralizes SDN’s control plane to tackle the single point of failure while maintaining the global view by sharing network events among multiple controllers. High-performance secure Diffie-Hellman protocol is combined with BC to authenticate the communication channel between the controller and the switch. This approach is considered as a hybrid approach since it utilizes BC to protect the global view at control plane level and to authenticate the communication channel between the controller and the switch.

SDN for BC. In permissioned BC, only certain nodes can be involved in block creation and validation.²⁷ Permissioned BC uses Byzantine Fault Tolerance (BFT) protocols, such as PBFT, as a consensus protocol that employs state machine replication to cope with Byzantine failures. The advantages of permissioned blockchains are low cost and low latency. However, their performance is affected when BC nodes are faulty or under attack. To address this issue, SDN can be utilized to enhance the availability of BC nodes and to protect them against unauthorized access and various network attacks. As shown in Figure 1, SDNs can be used as a framework to protect and optimize the network layer of BCs. Accordingly, we categorized the research efforts to secure BC through SDN into the following two groups:

Availability. As we mentioned previously, permissioned BCs have a limited number of participants. Therefore, maximizing nodes’ availability is a major challenge in such peer-to-peer networks. To this end, Steichen et al.³³ propose Chain-Guard, an SDN-based firewall for protecting permissioned BCs against DoS/DDoS attacks, which can influence how consensus is reached and can even prevent BC from working correctly. To prevent malicious actions, all traffic to BC nodes must be forwarded by at least one of the switches controlled by ChainGuard, which directly communicates with the BC nodes to distinguish between legitimate and malicious traffic that need to be filtered to protect BC nodes.

Flow-based access control. BC nodes must be protected against intruders

that deliberately attack the infrastructure of BC to gain access or attempt to disturb legitimate BC transactions. To address this issue, El Houda et al.⁹ present ChainSecure to protect permissioned blockchains from DNS amplification attacks. ChainSecure consists of the following three schemes. First, stateful mapping scheme (SMS) that performs one-to-one mapping between DNS requests and responses. Second, entropy calculation scheme (ECS) to measure randomness of data to detect illegitimate DNS requests using sFlow. Third, DNS DDoS mitigation module. The experimental results showed that ChainSecure can achieve high detection rate with approximately 30% of false positives. Figure 2 depicts the taxonomy for the integration between SDN and BC. Additionally, we highlight the advantages and disadvantages of this integration between SDN and BC in the accompanying table.

Proposed Architecture

The nature of BC can be utilized to tackle the problem of single point of failure, and to ensure the consistency of global view among cross-domain controllers in a decentralized manner. In this section, we propose BC-Sec-SDN, a hierarchical architecture that adopts BC to fulfill the requirements of fault-tolerant, decentralized, and secure SDN based on consortium BC approach. In BC-Sec-SDN each SDN domain is managed by an SDN controller. Unlike the previously discussed approaches, our architecture decouples SDN control plane into two separate control layers namely reac-

tive and proactive layers. The reactive control layer consists of SDN controllers that receive Packet-In messages from SDN’s data plane and prepare proposals of BC transactions and smart contracts accordingly. Whereas the proactive control layer executes previously committed BC transactions and smart contracts. We utilize BC to securely authorize SDN’s core components including applications, resources, and flows. Moreover, we securely offload proactive network functionality through smart contracts, which provide higher level of adaptivity, decentralization, and fault tolerance. However, the main goal of the proactive layer is to reduce the monitoring task on SDN controllers in the reactive layer, which allows them to focus on certain important events and take critical decisions based on the decentralized approach of BC.

As shown in Figure 3, controllers are positioned vertically with different responsibilities. The reactive control layer has multiple distributed SDN controllers, which instead of directly provisioning flow tables, they deploy appropriate smart contracts and asynchronously send BC transactions to the second level of hierarchy (that is, proactive control layer) without incurring any further delay in control plane’s processing time,²⁹ which ultimately reduces the overall latency and increases the throughput. The SDN controllers utilize write-ahead-log approach,²¹ where all flow updates are written to a redo log¹⁰ before they are installed in the data plane. A redo log record is executed once the submitted transaction is committed by the

Summary of the advantages and disadvantages of the integration between SDN and BC.

	Advantages	Disadvantages
BC for SDN	<ul style="list-style-type: none"> ▶ BC transactions and smart contracts are immutable. ▶ Provides a decentralized control-plane. ▶ Authentication for application flows. ▶ Protection of SDN’s global view. ▶ Securely sharing data in cross-domain SDNs without relying on third parties. ▶ Ensures reliability, traceability, and auditability. ▶ Improves trust management. ▶ Facilitates collaborative defense approaches. 	<ul style="list-style-type: none"> ▶ Characterized by high power consumption. ▶ Additional latency and lower throughput due to computation and communication overhead. ▶ Its performance depends on consensus protocols. ▶ Data replication requires additional space.
SDN for BC	<ul style="list-style-type: none"> ▶ Enhances the network layer of BC. ▶ Improves the availability of BC nodes. ▶ Provides flow-based access control for BC nodes. 	<ul style="list-style-type: none"> ▶ Protects BC only from network-based attacks. ▶ Attackers may exploit traditional SDN vulnerabilities to gain access to BC.

BC controllers. This idea is mainly inspired by the transaction concept¹⁰ in database systems.

On the other hand, from BC perspective, controllers in the proactive layer (that is, BC controllers) are full BC nodes, which are responsible for executing verified BC transactions and smart contracts as well as taking proactive decisions on behalf of the upper level of hierarchy via SDN's southbound API. As a result, SDN here is implemented based on execution of verified transactions and BC smart contracts, which dictate how flow tables should be filled or updated and also determine how to trust and

revoke privileges from the underlying forwarding devices.

In reactive settings, first the SDN switch submits a new signed Packet-In message to the corresponding SDN controller (step 1), which verifies the message (step 2), and then updates the transaction log and submits a new transaction proposal to the BC controller (step 3), which will send the appropriate BC transaction to other BC controllers for consensus (step 4). Once the transaction is committed, the SDN controller will be notified (step 5), which finally sends a new signed Packet-Out to the corresponding switch to add a new flow entry

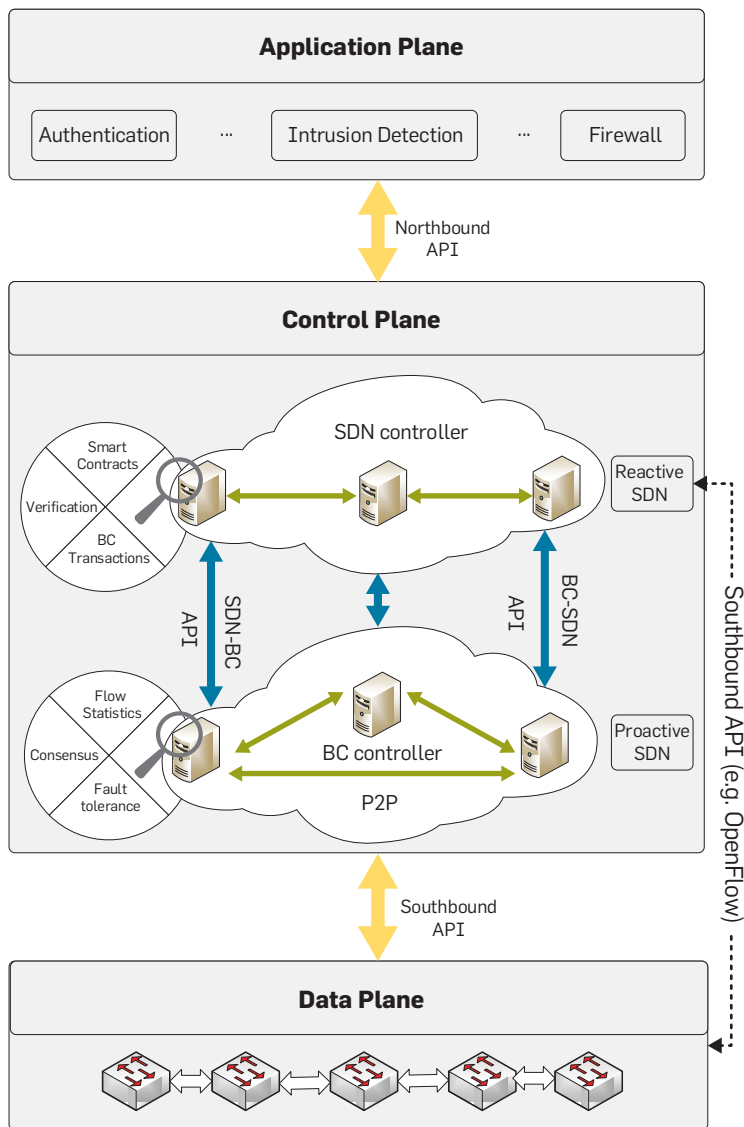
once the transaction is verified and matched with a corresponding entry in the transaction log (step 6). Figure 4 shows the reactive SDN control layer.

In proactive settings, as shown in Figure 5, once a flow is authorized further actions and access-control policies are defined by the proactive SDN layer based on committed BC smart contracts. The SDN controller submits the appropriate smart contract proposal (step 1). Next, the BC controller sends the appropriate contract to other BC controllers for PBFT [4] based consensus (step 2). Once the contract is committed (step 3), the BC controller periodically collects flow statistics from SDN switches using Stats Request messages (step 4) and then submits update transactions to the corresponding smart contract to update the current BC state (step 5). These updates will result in new events, which may further include offloaded or non-offloaded functionalities. Next, BC or SDN controller will be notified for provisioning offloaded or non-offloaded functionalities, respectively (step 6). Finally, the corresponding controllers send signed Packet-Out messages to provision the related switches with the appropriate flow entries (step 7). In case of failure, smart contracts will also allow SDN switches to connect to alternative controllers.

The advantage of our architecture is securely sharing of the global view of SDN through BC, which reduces the communication and synchronization overhead among SDN controllers at the reactive control layer. More precisely, these controllers directly communicate with BC-controllers at the next level of hierarchy, which verify submitted transaction proposals and maintain committed transactions in a decentralized, distributed ledger. Moreover, our architecture securely offloads proactive decision making from SDN controllers to smart contracts to increase the level of adaptivity of the whole SDN network.

As SDN controllers delegate the proactive decision making to BC controllers, the overload on SDN controllers will be significantly reduced. Therefore, our architecture shows that BC not only protects the SDN network but also it can be efficiently utilized to

Figure 3. Proposed integration between SDN and BC.



replace dumb packet forwarding with more secure, distributed, and autonomous fashion.

However, due to communication overhead of PBFT,⁴ throughput and latency for large-scale SDNs are among the main technical challenges in our architecture. This is primarily because PBFT requires a quadratic number of messages in the number of participating BC nodes.¹⁸ Additionally, the requirement of low-latency BC-enabled SDN can limit the maximum block size in BC network.²

This also implies that BC controllers need to scale their computing power and storage resources in accordance with the increase in number of transactions.¹⁷ On the other hand, consensus protocols adopted in BC-Sec-SDN need to take into consideration that different controllers have different computational capabilities as well as the constantly exchanged messages in all-to-all communication pattern may affect the controller throughput. Therefore, there is a need to improve the performance of BC so that it can meet the requirements of SDN in realistic scenarios.

Our architecture falls into structural solutions according to our taxonomy. Apart from the approaches discussed in this paper, we divide the SDN control plane into reactive and proactive SDN layers, and securely offload proactive decision making to BC smart contracts.

Future Directions

Here, we discuss future research directions that must be considered to meet the requirements of SDN while adopting BC technology.

Lightweight consensus protocols.

Consensus protocols need to be optimized to reduce the power consumption of the participating nodes, which may also need to perform different network functionalities while keeping the performance at a reasonable level. Lightweight consensus protocols can reach consensus in short time, which ultimately increases the overall throughput.

Security testing tools. Existing security testing tools commonly used for non-BC systems cannot be directly utilized in BC settings, and therefore, there is a need to design and imple-

Figure 4. Reactive SDN control.

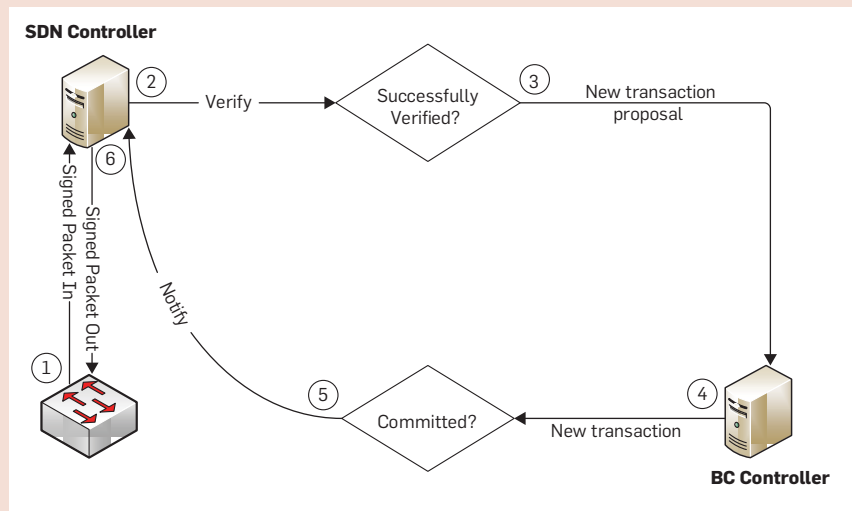
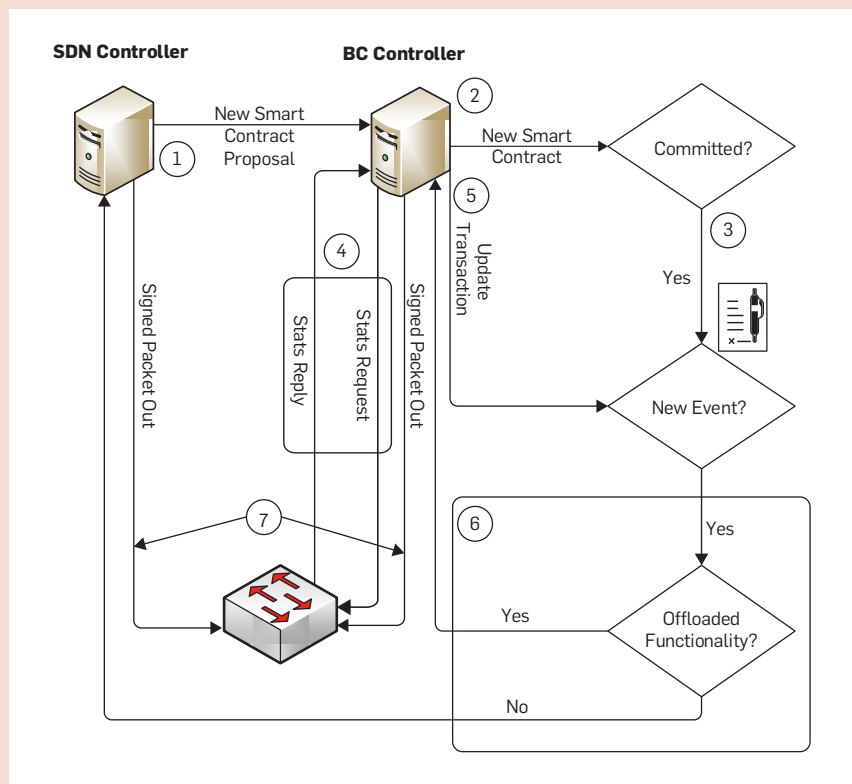


Figure 5. Proactive SDN control (Offloading proactive decision making to BC smart contracts).



ment BC-tailored tools that consider the main characteristics of BC systems.³

Decoupling data from block header. Decoupling data from block header allows BC transactions to be stored externally (that is, kept off-chain) to avoid consuming additional space while making node-specific (that is, SDN controller) inserting and querying faster.²⁰ This approach also allows

parallel insertion of new transactions in BC since these transactions are chained together in the same block based on the node’s block header.

Trust-based validation. Rather than validating all new transactions submitted to BC, trust evaluation algorithms can be employed to validate a fraction of these transactions based on each node’s trust score. This ap-

proach is adopted in Dorri et al.^{6,7} to provide a lightweight, scalable, and efficient blockchain for IoT environments. However, for new nodes which have no trust score, all their new transactions need to be verified. Additionally, BC nodes should not be fully trusted to defend against compromised nodes.⁶

Fine-grained multi-level trust. BC can also be used to provide fine-grained trust and authentication mechanisms among SDN components at different levels including the application, the control, and the data planes. In other words, instead of utilizing BC to only validate flow-level data, one can use it to improve the trustworthiness of the core components of the SDN architecture. This may also imply that these components need to be redesigned to interact only with trusted SDN components.

AI-enabled BC. AI-enabled BC improves the capabilities of BC by analyzing the previous transactions submitted to BC to provide a better decision making and rewarding mechanisms for BC participants based on their collective behaviors. Additionally, AI-enabled smart contracts allow more complex decisions to be taken based on BC principles.³¹ Therefore, AI-enabled smart contracts can be utilized to support secure, decentralized, and intelligent decision making in SDNs.

Conclusion

In this work, we provided a detailed survey on the potential integration between SDN and blockchain. We categorized the existing works into two main categories namely BC for SDN and SDN for BC. Our work showed that operational and structural enhancements were proposed to achieve a fault-tolerant, decentralized, and secure SDN by blockchain utilization. Whereas SDN can be utilized to improve the availability and to provide flow-based access control for the primary nodes participated in permissioned BC. We also took a step forward by introducing BC-Sec-SDN, a hierarchical architecture that improves SDN in terms of security as well as synchronization of SDN's global view in a cross-domain SDN network. Moreover, our architecture securely offloads proactive decision making from SDN controllers to

smart contracts. The main challenges, however, are related to limitations of current BC systems in terms throughput, latency, and consensus protocols, which need to be improved to meet the requirements of SDN under high load scenarios. C

References

- Alharbi, T. Deployment of blockchain technology in software defined networks: A survey. *IEEE Access* 8 (2020), 9146–9156.
- Androulaki, E. et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the 13th EuroSys Conf.*, 2018, 1–15.
- Bosu, A., Iqbal, A., Shahriyar, R., and Chakraborty, P. Understanding the motivations, challenges and needs of blockchain software developers: A survey. *Empirical Softw. Eng.* 24, 4 (2019), 2636–2673.
- Castro, M. et al. Practical byzantine fault tolerance. *OSDI '99* (1999), 173–186.
- Dai, H., Zheng, Z., and Zhang, Y. Blockchain for internet of things: A survey. *IEEE Internet of Things J.* 6, 5 (2019), 8076–8094.
- Dorri, A., Kanhere, S., and Jurdak, R. Mof-bc: A memory optimized and flexible blockchain for large scale networks. *Future Generation Computer Systems* 92 (2019), 357–373.
- Dorri, A., Kanhere, S., Jurdak, R., and Gauravaram, P. Lsb: A lightweight scalable blockchain for IoT security and anonymity. *J. of Parallel and Distributed Computing* 134 (2019), 180–197.
- El Houda, Z., Hafid, A., and Khoukhi, L. Cochain-sc: An intra-and inter-domain ddos mitigation scheme based on blockchain using SDN and smart contract. *IEEE Access* 7 (2019), 98893–98907.
- El Houda, Z., Khoukhi, L., and Hafid, A. Chainsecure—A scalable and proactive solution for protecting blockchain applications using SDN. In *Proceedings of the 2018 IEEE Global Communications Conf.* 1–6.
- Gray, J. et al. The transaction concept: Virtues and limitations. *Vldb 81* (1981), 144–154.
- Jiasi, W., Jian, W., Jia-Nan, L., and Yue, Z. Secure software-defined networking based on blockchain. 2019; *arXiv preprint 1906.04342*.
- Jindal, A., Aujla, G., and Kumar, N. Survivor: A blockchain based edge-as-a-service framework for secure energy trading in SDN-enabled vehicle-to-grid environment. *Computer Networks* 153 (2019), 36–48.
- Kalkan, K., Gur, G., and Alagoz, F. Defense mechanisms against DDoS attacks in SDN environment. *IEEE Commun.* 55, 9 (2017), 175–179.
- Kataoka, K., Gangwar, S., and Podili, P. Trust list: Internet-wide and distributed IoT traffic management using blockchain and SDN. In *Proceedings of the 2018 IEEE 4th World Forum on Internet of Things*, 296–301.
- Kreutz, D., Ramos, F., Verissimo, P., Rothenberg, C., Azodolmolky, S., and Uhlig, S. Software-defined networking: A comprehensive survey. In *Proceedings of the 2014 IEEE 103, 1* (2014), 14–76.
- Li, W., Meng, W., Liu, Z., and Au, M. Towards blockchain-based software-defined networking: Security challenges and solutions. *IEICE Trans. on Information and Systems* 103, 2, (2020), 196–203.
- Li, Z., Kang, J., Yu, R., Ye, D., Deng, Q., and Zhang, Y. Consortium blockchain for secure energy trading in industrial internet of things. *IEEE trans. on industrial informatics*, 14(8):3690–3700, 2017.
- Luu, L., Narayanan, V., Zheng, C., Baweja, K., Gilbert, S., and Saxena, P. A secure sharding protocol for open blockchains. In *Proceedings of the 2016 ACM SIGSAC Conf. on Computer and Commun. Security*, 17–30.
- McKeown, N. et al. OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM Computer Commun. Rev.* 38, 2 (2008), 69–74.
- Michelin, R. et al. Speedychain: A framework for decoupling data from blockchain for smart cities. In *Proceedings of the 15th EAI Intern. Conf. on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2018, 145–154.
- Mohan, C., Haderle, D., Lindsay, B., Pirahesh, H., and Schwarz, P. Aries: A transaction recovery method supporting fine-granularity locking and partial rollbacks using write-ahead logging. *ACM Trans. on Database Systems* 17 1, (1992), 94–162.
- Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.
- Nguyen, T. and Yoo, M. Analysis of link discovery service attacks in SDN controller. In *Proceedings of the 2017 Intern. Conf. on Information Networking*, 259–261.
- Pashkov, V., Shalimov, A., and Smeliansky, R. Controller failover for SDN enterprise networks. In *Proceedings of the 2014 Intern. Science and Technology Conf. (Modern Networking Technologies)*, 1–6.
- Pourvahab, M. and Ekbatanifard, G. An efficient forensics architecture in software-defined networking using blockchain technology. *IEEE Access* 7 (2019), 99573–99588.
- Qiao, Q., Li, X., Wang, Y., Luo, B., Ren, Y., and Ma, J. Credible routing scheme of SDN-based cloud using blockchain. In *Proceedings of the Intern. Conf. of Pioneering Computer Scientists, Engineers, and Educators*. Springer, 2019, 189–206.
- Qiu, C., Yu, F., Yao, H., Jiang, C., Xu, F., and Zhao, C. Blockchain-based software-defined industrial internet of things: A dueling deep q-learning approach. *IEEE Internet of Things J.*, 2018.
- Rathore, S., Kwon, B., and Park, J. Blocksecinet: Blockchain-based decentralized security architecture for IoT network. *J. of Network and Computer Applications* 143 (2019), 167–177.
- Sakic, E. and Kellerer, W. Impact of adaptive consistency on distributed sdn applications: An empirical study. *IEEE J. on Selected Areas in Commun.* 36, 12 (2018), 2702–2715.
- Shao, Z., Zhu, X., Chikuvanyanga, A., and Zhu, H. Blockchain-based SDN security guaranteeing algorithm and analysis model. In *Proceedings of the Intern. Conf. on Wireless and Satellite Systems*. Springer, 2019, 348–362.
- Sharma, P., Kumar, N., and Park, J. Blockchain technology toward green IoT: Opportunities and challenges. *IEEE Network*, 2020.
- Sharma, P., Singh, S., Jeong, Y., and Park, J. Distblocknet: A distributed blockchain-based secure sdn architecture for IoT networks. *IEEE Commun.* 55, 9 (2017), 78–85.
- Steichen, M., Hommes, S., and State, R. Chainguard—a firewall for blockchain applications using SDN with OpenFlow. In *Proceedings of IEEE 2017 Principles, Systems and Applications of IP Telecommunications*, 1–8.
- Wen, X., Chen, Y., Hu, C., Shi, C., and Wang, Y. Towards a secure controller platform for OpenFlow applications. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, 2013, 171–172.
- Wood, G. et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper 151* (2014), 1–32.
- Yang, H., Liang, Y., Yao, Q., Guo, S., Yu, A., and Zhang, J. Blockchain-based secure distributed control for software defined optical networking. *China Commun.* 16, 6 (2019), 42–54.
- Yazdinejad, A., Parizi, R., Dehghantaha, A., and Choo, K. P4-to-blockchain: A secure blockchain-enabled packet parser for software defined networking. *Computers & Security*, 2019, 101629.
- Yazdinejad, A., Parizi, R., Dehghantaha, A., and Choo, K. Blockchain-enabled authentication handover with efficient privacy protection in SDN-based 5g networks. *IEEE Trans. on Network Science and Engineering*, (2019), 1–1.
- Yazdinejad, A., Parizi, R., Dehghantaha, A., Zhang, Q., and Choo, K. An energy-efficient sdn controller architecture for IoT networks with blockchain-based security. *IEEE Trans. on Services Computing*, 2020.
- Zhao, N., Wu, H., and Zhao, X. Consortium blockchain-based secure software defined vehicular network. *Mobile Networks and Applications*, 2019, 1–14.
- Zheng, Z., Xie, S., Dai, H., Chen, X., and Wang, H. Blockchain challenges and opportunities: A survey. *Intern. J. of Web and Grid Services* 14, 4 (2018), 352–375.

Majid Latah is a Ph.D. candidate in the Department of Computer Science at Ozyegin University in Turkey.

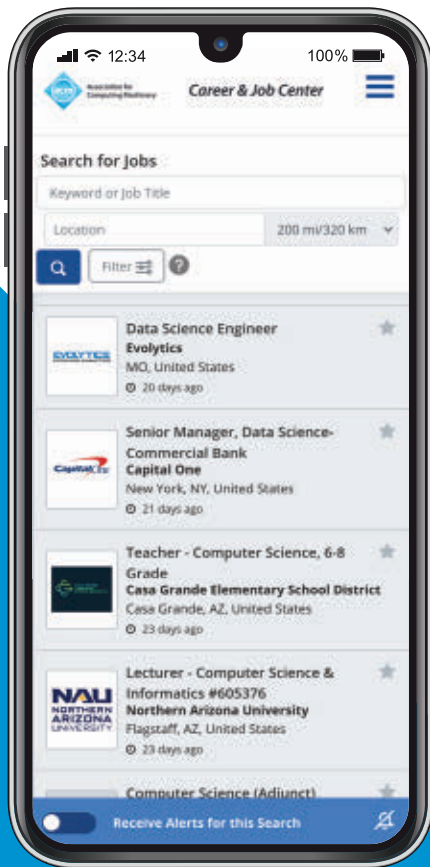
Kubra Kalkan is a faculty member in the Computer Science Department of Ozyegin University in Turkey. Previously, she was a researcher with various institutions, such as EPFL, Microsoft Redmond, Northeastern University, and the University of Oxford.

The #1 Career Destination
to Find Computing Jobs.

Connecting you with top
industry employers.



The new ACM Career & Job Center offers job seekers a host of career-enhancing benefits, including:



Access to new and exclusive career resources, articles, job searching tips and tools.



Gain insights and detailed data on the computing industry, including salary, job outlook, 'day in the life' videos, education, and more with our new Career Insights.



Redesigned job search page allows you to view jobs with improved search filtering such as salary, location radius searching and more without ever having to leave the search results.



Receive the latest jobs delivered straight to your inbox with **new exclusive Job Flash™ emails**.



Get a free resume review from an expert writer listing your strengths, weaknesses, and suggestions to give you the best chance of landing an interview.



Receive an alert every time a job becomes available that matches your personal profile, skills, interests, and preferred location(s).

Your next job is right at
your fingertips.
Get started today!

Visit <https://jobs.acm.org/>

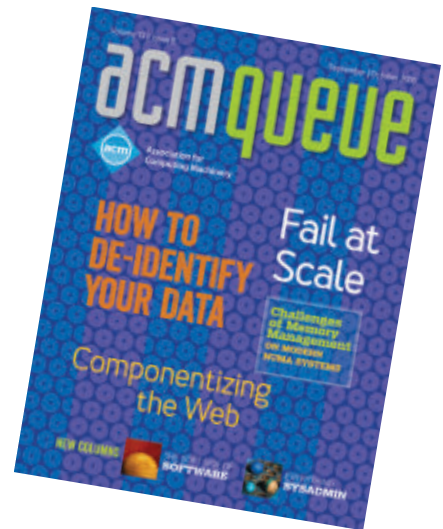
acmqueue

Check out the acmqueue app

FREE TO ACM MEMBERS

acmqueue is ACM's magazine by and for practitioners, bridging the gap between academics and practitioners of the art of computer science. For more than a decade *acmqueue* has provided unique perspectives on how current and emerging technologies are being applied in the field, and has evolved into an interactive, socially networked, electronic magazine.

Broaden your knowledge with technical articles focusing on today's problems affecting CS in practice, video interviews, roundtables, case studies, and lively columns.



Keep up with this fast-paced world on the go. Download the mobile app.



Association for
Computing Machinery

Desktop digital edition also available at queue.acm.org.
Bimonthly issues free to ACM Professional Members.
Annual subscription \$19.99 for nonmembers.

research highlights

P. 82

**Technical
Perspective
Physical Layer
Resilience through
Deep Learning in
Software Radios**

By Falko Dressler

P. 83

Polymorphic Wireless Receivers

By Francesco Restuccia and Tommaso Melodia

P. 92

**Technical
Perspective
The Effectiveness
of Security Measures**

By Nicolas Christin

P. 93

Measuring Security Practices

By Louis F. DeKoven, Audrey Randall, Ariana Mirian,
Gautam Akiwate, Ansel Blume, Lawrence K. Saul,
Aaron Schulman, Geoffrey M. Voelker, and Stefan Savage

Technical Perspective

Physical Layer Resilience through Deep Learning in Software Radios

By Falko Dressler

RESILIENCE IS THE NEW HOLY GRAIL in wireless communication systems. Complex radio environments and malicious attacks using intelligent jamming contribute to unreliable communication systems. Early approaches to deal with such problems were based on frequency hopping, scrambling, chirping, and cognitive radio-based concepts, among others. Physical-layer security was increased using known codes and pseudorandom number sequences. However, these approaches are not up to modern standards; they do not improve resilience and are rather easy to attack by means of intelligent jamming.

Conceptually, dynamic changing waveforms and physical layer parameters would help overcoming many of these issues. However, almost all modern radio technologies are rather inflexible when it comes to changing physical layer parameters on the fly. For example, Bluetooth is limited to frequency hopping, and Wi-Fi to switching channels and modulation/encoding schemes based on active scanning. What is needed is a system that continuously changes physical layer configurations, that is, carrier frequency, FFT size, symbol modulation, and even the position of header and pilots. This way, the overall resilience of the wireless system would be improved significantly.

There are some works that address the sender side, that is, how to enable a sender to vary physical layer parameters, switching even between completely different waveform designs. Off-the-shelf chips are usually not able to do this—for a good reason, they are bound to operate within the limits of the standardized protocols. There is, however, a concept called cross-technology communication (CTC), which acts as an enabler for dynamic changes of waveforms, modulation, and more. For example, it has been shown that normal Wi-Fi chips can be used to emulate Bluetooth, LTE, ZigBee, and many other waveforms by sending carefully chosen Wi-Fi signals. This way,


communication between entirely different technologies becomes possible—significantly enhancing the resilience of the communication system.

The following paper tackles the problem from a completely new perspective. The presented PolymoRF concept represents a polymorphic wireless receiver entirely based on physical-layer deep learning. PolymoRF focuses entirely on the receiver system: The receiver can determine on the fly the current parameterization used by the sender. The main idea is to apply a novel deep learning-based model, named RFNet, to infer physical layer parameters from I/Q samples collected by a software radio. The I/Q representation stands for in-phase and quadrature components using real and imaginary parts of complex numbers; and allows to accurately represent phase and amplitude of a signal. Modulated signals, thus, span an image in the plane, which is called a constellation diagram.

Conceptually, the authors go one step beyond classic software-defined radio approaches that use an analog radio frontend in combination with a software frontend (typically a mix of FPGA and CPU-based parts). As all signal processing up to demodulation and decoding is realized in software, new functionality, here the RFNet model, can be deeply integrated with the software radio in the FPGA implementation. This helps bring deep learning much closer to wireless signal processing; they achieve three goals eminent for resilient wireless communications:

A *deep learning architecture* is created to analyze radio signals without feature extraction and selection. This is achieved by conceptually arranging I/Q samples represented in the complex I/Q plane in the form of an image, which can be approached and analyzed more easily using convolutional neural networks. This is a rather novel approach to I/Q sample processing. A *general-purpose hardware/software ar-*

chitecture is designed that is like existing software-defined radio solutions, but unique in the way that pipelining and unrolling techniques have been integrated helping to reduce the latency of the RFNet model in the signal processing chain. A *system-level feasibility*—most important from a systems perspective and for combining theory and practice—is performed by implementing the proposed concepts in a testbed. For this, the RFNet model has been implemented in software and realized as a system on a chip using FPGA technology. Results demonstrate this can be done without introducing much additional latency. This reality check clearly outlines the advantages of the polymorphic receiver concept.

What is missing and what can we expect in the future from this line of research? The polymorphic wireless receiver concept is without doubt a major step forward, offering a door to next-generation resilient wireless communication systems. This physical layer deep-learning approach on the receiver side now must be combined with adequate transmitter concepts. This can be classic wireless transmitter modules or novel multidimensional scrambler techniques that make jamming more difficult and helps overcoming natural disturbances of the wireless radio signals including interference from other communication protocols. In this research, cross-technology communication may help reusing existing chips, like Wi-Fi chips, to enable such features. We can expect to see implementations combining polymorphic wireless receivers with approaches using learning concepts on the sending side. Eventually, this line of research will improve resilience in wireless communications well beyond our current imagination. 

Falko Dressler is a professor and chair of Telecommunications Networks in the School of Electrical Engineering and Computer Science at TU Berlin, Germany.

Copyright held by author.

Polymorphic Wireless Receivers

By Francesco Restuccia and Tommaso Melodia

Abstract

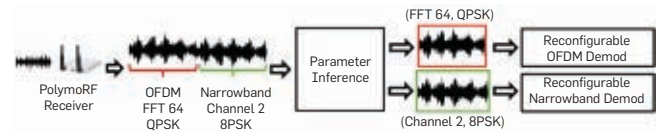
Today's wireless technologies are largely based on inflexible designs, which make them inefficient and prone to a variety of wireless attacks. To address this key issue, wireless receivers will need to (i) infer on-the-fly the physical layer parameters currently used by transmitters; and if needed, (ii) change their hardware and software structures to demodulate the incoming waveform. In this paper, we introduce *PolymoRF*, a deep learning-based polymorphic receiver able to reconfigure itself in real time based on the inferred waveform parameters. Our key technical innovations are (i) a novel embedded deep learning architecture, called *RFNet*, which enables the solution of key waveform inference problems, and (ii) a generalized hardware/software architecture that integrates *RFNet* with radio components and signal processing. We prototype *PolymoRF* on a custom software-defined radio platform and show through extensive over-the-air experiments that *PolymoRF* achieves throughput within 87% of a perfect-knowledge Oracle system, thus demonstrating for the first time that polymorphic receivers are feasible.

1. INTRODUCTION

It has been forecast that over 50 billion mobile devices will be soon connected to the Internet, creating the biggest network the world has ever seen.³ However, only very recently has the community started to acknowledge that squeezing billions of devices into tiny spectrum portions will inevitably create disruptive levels of interference. Although Mitola and Maguire first envisioned the concept of “cognitive radios” 20 years ago,⁸ today's commercial wireless devices still use inflexible wireless standards such as Wi-Fi and Bluetooth—and thus, are still very far from being truly real-time reconfigurable. Just to give an example of the seriousness of the spectrum inflexibility issue, DARPA has recently invested to launch the spectrum collaboration challenge (SC2), where the target is to design spectrum access schemes that “[...] best share spectrum with any network(s), in any environment, without prior knowledge, leveraging on machine-learning techniques.”²⁵

Intuitively, the issues of existing communication systems could be addressed by allowing transmitters to dynamically switch parameters such as carrier frequency, FFT size, and symbol modulation without coordination with the receiver. This will allow the transmitter efficient spectrum occupation using the most appropriate wireless scheme at any given moment. Figure 1 shows an example of a polymorphic receiver able to infer the current transmitter's physical layer scheme (e.g., OFDM vs. narrowband) and the scheme's parameters (e.g., FFT size,

Figure 1. Example of a self-adaptive polymorphic receiver.



channel, modulation), and then demodulate each portion of the signal.

Doing away with explicit coordination and inflexible physical layers is the first step toward wireless receivers able to self-adapt to demodulate many waveform with a single radio interface.¹⁵ Yet, despite their compelling necessity, these wireless receivers do not exist today. This manuscript aims to change the current state of affairs by proposing the first demonstration of *PolymoRF*, the first *polymorphic wireless receiver*. Achieving this goal required us to address a set of key research challenges summarized below:

- (1) *Keeping up with the transmitter.* A crucial aspect is the real-time parameter inference. In practical systems, however, transmitters may choose to switch its parameter configuration in the order of milliseconds (e.g., frequency hopping, rate adaptation). For example, if the transmitter chooses to switch modulation every 100ms, the learning model should run in (much) less than 100 ms to predict the parameters and morph the receiver into a new configuration. To this end, we will show in Section 5.5 that CPU latency is several orders of magnitude greater than what is required to sustain realistic sampling rates from the RF interface. Thus, we need hardware-based designs to implement low-latency knowledge extraction techniques.
- (2) *Creating learning architectures for the embedded RF domain.* Recent advances in RF deep learning^{10-12, 14} have demonstrated that convolutional neural networks (ConvNets) may be applied to analyze RF data without feature extraction and selection algorithms.⁴ Moreover, ConvNets present a number of characteristics (discussed in Section 3) that make them particularly desirable from a hardware implementation perspective. However, these solutions cannot be applied to implement real-time polymorphic wireless communications—as shown in Section 5.5, existing

The original version of this paper was published in *Proceedings of the 21st Int. Symp. on Theory, Algorithmic Foundations and Protocol Designs for Mobile Networks and Mobile Computing* (Oct. 2020), 271–280.

art^{10, 12} utilizes general-purpose architectures with a very high number of parameters, requiring hardware resources and latency that go beyond what is acceptable in the embedded domain. This crucial issue calls for novel, RF-specific, real-time architectures. We are not aware of learning systems tested in a real-time wireless environment and used to implement inference-based wireless systems.

- (3) *System-level feasibility of polymorphic platforms.* It is yet to be demonstrated whether polymorphic platforms are feasible and effective. This is not without a reason—from a system perspective, it required us to tightly interconnect traditionally separated components, such as CPU, RF front-end, and embedded operating system/kernel, to form a seamlessly running low-latency learning architecture closely interacting with the RF components and able to adapt at will its hardware and software based on RF-based inference. Furthermore, since polymorphic wireless systems are subject to inference errors, we need to test its performance against a perfect knowledge (thus, ideal and not implementable) system.

1.1. Technical contributions

This paper's key innovation is to finally bridge the gap between the extensive theoretical research on cognitive radios and the associated system-level challenges, by demonstrating that inference-based wireless communications are indeed feasible on off-the-shelf embedded devices. Beyond the examples and the evaluation conducted in Section 5, the main purpose of this work is to provide a *blueprint* for next-generation wireless receivers, where their radio hardware and software are not *protocol-specific*, but instead *spectrum-driven* and adaptable on-the-fly to different waveforms.

We summarize our main technical contributions as follows:

- (1) We design a novel learning architecture called *RFNet*, specifically and carefully tailored for the embedded RF domain. Our key intuition in *RFNet* is to arrange I/Q samples to form an “image” that can be effectively analyzed by the ConvNet filters. This operation produces high-dimensional representations of small-scale transition in the I/Q complex plane, which can be leveraged to efficiently solve a wide variety of complex RF classification problems such as RF modulation classification. Extensive experimental evaluation indicates that a pipelined version of *RFNet* significantly reduces latency with respect to a CPU implementation;
- (2) We propose a general-purpose hardware/software architecture for software-defined radios that enables the creation of custom polymorphic wireless systems through *RFNet*. Then, we implement a multipurpose library based on high-level synthesis (HLS) that translates an *RFNet* model implemented in software to a circuit implemented in the FPGA portion of the SoC. Moreover, we leverage key optimization strategies such as pipelining and unrolling to further reduce the

latency of *RFNet* by more than 50% with respect to the unoptimized version, with only 7% increase of hardware resource consumption. Finally, we design and implement the device-tree entries and Linux drivers enabling the system to utilize *RFNet* and other key hardware peripherals.

- (3) We prototype *PolymoRF* on a ZYNQ-7000 system-on-chip (SoC) and analyze its performance on a scheme where the transmitter can switch among three FFT sizes and three symbol modulation schemes without explicit notification to the receiver. A demo video of *PolymoRF* where the transmitter switches FFT size every 0.5s is available at https://youtu.be/5vf_pb0nvKk. We believe ours is the first demonstration of real-time OFDM reconfigurability without explicit transmitter/receiver coordination. Experiments on both line-of-sight (LOS) and non-line-of-sight (NLOS) channel conditions show that the system achieves at least 87% of the throughput of a perfect knowledge—and thus, unrealistic—*Oracle* OFDM system, thus proving the feasibility of polymorphic receivers.

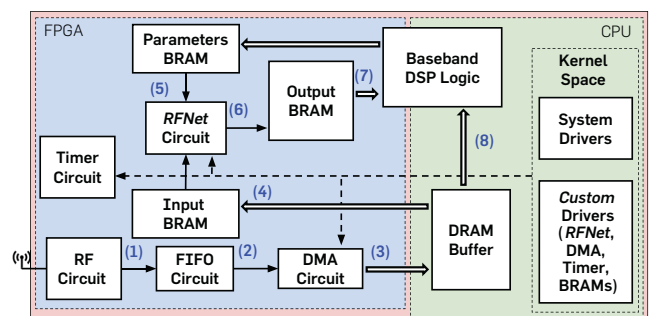
2. PolymoRF: AN OVERVIEW

The primary operations performed by the *PolymoRF* platform are summarized in Figure 2. In a nutshell, *PolymoRF* can be considered as a full-fledged learning-based software-defined radio architecture where both the inference system and the demodulation strategy can be morphed into new configurations at will.

We provide a walk-through of the main operations performed by *PolymoRF* with the help of Figure 2. Although for simplicity we refer to specific hardware equipment and circuits in our explanation, we point out that the building blocks of our platform design (BRAMs, DMA, FIFOs, etc.) can be implemented in any commercially available FPGA platform.

We assume the transmitter may transmit by choosing among a discrete set of physical layer parameters that are known at the receiver's side. We define as Y a tuple of such physical layer parameters, which may be changed at will by the transmitter but not before T_{sw} seconds between each change, which we refer to a *switching time*. For the sake of generality, in this paper we will not assume any particular strategy in the transmitter's parameter choice, which can be driven by a series of factors (including anti-jamming strategy, noise avoidance, throughput optimization, and so on)

Figure 2. Modules and operations of *PolymoRF*.



that will be considered as out of the scope of this paper, whose main focus is instead on the receiver's side.

- (1) *Reconfigurable radio front-end.* The RF signal is received (step 1) through a reconfigurable RF front-end. In our prototype, we used an AD9361¹ radio interface, which supports frequency range between 70 MHz and 6.0 GHz and channel bandwidth between 200 kHz and 56 MHz. We chose the AD9361 because it is commonly used in software-defined radio systems—indeed, it is also used by USRPs such as the E310 and B210. Moreover, the AD9361 provides basic FPGA reference designs and kernel-space drivers to ease prototyping and extensions. Perhaps more importantly, the AD9361 local oscillator (LO) frequency and RF bandwidth can be reconfigured at will through CPU registers.
- (2) *Conversion from RF to FPGA domain.* The AD9361 produces streams of I/Q samples of 200 Msamples/second—hence, it is clocked at 200 MHz. Since the AD9361 clock would be too fast for the other circuits in the FPGA, we implemented a FIFO to adapt the speed of samples from the AD9361 to the 100 MHz clock frequency used by the other circuits in the FPGA (step 2). We then use a direct memory access (DMA) core to store the stream of I/Q samples to a buffer in the DRAM (step 3). The use of DMA is crucial as the CPU cannot do the transfer itself, since it would be fully occupied for the entire duration of the read/write operation and thus unavailable to perform other work. Therefore, we wrote a custom DMA driver to periodically fill a buffer of size B residing in the DRAM with a subset of I/Q samples coming from the FIFO.
- (3) *Learning and receiver polymorphism.* After the buffer has been replenished, its first X I/Q samples are sent to a BRAM (step 4) constituting the input to *RFNet*, a novel learning architecture based on *ConvNets*. This circuit is the fundamental core of the *PolymoRF* system; therefore, we will dedicate Sections 3 and 4 to discuss in detail its architecture and implementation, respectively. The parameters of *RFNet* are read by an additional BRAM (step 5), which in effect allows the reconfiguration of *RFNet* to address multiple RF problems according to the current platform need. As explained in Section 3, *RFNet* produces a probability distribution over the transmitter's parameter set Y . After *RFNet* has inferred the transmitter's parameters, it writes on a block-RAM (BRAM) its probability distribution (step 6). Then, the baseband DSP logic (which may be implemented in both hardware and software) reads the distribution from the BRAM (step 7) selects the parameter set with highest probability and “morphs” into a new configuration to demodulate the I/Q samples in B (step 8).

3. LEARNING SYSTEM: RFNet

We first motivate the use of convolutional neural networks for *RFNet*, we discuss some RF-specific learning challenges, and then we describe in details the *RFNet* input construction

and its complete architecture.

- (1) *Why using deep learning and not machine learning?* Deep learning relieves from the burden of finding the right “features” characterizing a given wireless phenomenon. At the physical layer, this is a key advantage for the following reasons. First, deep learning offers high-dimensional feature spaces. In particular, O'Shea et al.¹² have demonstrated that on the 24-modulation dataset considered, deep learning models achieve on the average about 20% higher classification accuracy than legacy learning models under noisy channel conditions. Second, automatic feature extraction allows to reuse the same hardware circuit to address different learning problems. Critically, this allows to keep both latency and energy consumption constant, which are particularly critical in wireless systems. Third, deep learning algorithms can be fine-tuned by performing batch gradient descent on fresh input data, avoiding manual re-tuning of the feature extraction algorithms.

- (1) *Why using ConvNets for wireless deep learning?* There are several primary advantages that make the usage of ConvNet-based models particularly desirable for the embedded RF domain. First, *convolutional filters are designed to interact only with a very small portion of the input.* We show in Section 5.3 that this key property allows achieving significantly higher accuracy than traditional neural networks. Perhaps even more importantly, *ConvNets are scalable with the input size.* For example, for a 200×200 input and a DL with 10 neurons, a traditional neural network will have $200^2 \cdot 10 = 400k$ weights, which implies a memory occupation of $4 \cdot 400k = 16$ Mbytes to store the weights of a single layer (i.e., a float number for each weight). Clearly, this is unacceptable for the embedded domain, as the network memory consumption would become intractable as soon as several DLs are stacked on top of the other.

Moreover, *ConvNet filtering operations can be made low latency by parallelization*, which makes them particularly suitable to be optimized for the RF domain. Finally, we show in Section 5 that the same ConvNet architectures can be reused to address different RF classification problems (e.g., modulation classification in single- and multicarrier systems), as long as the ConvNet is provided appropriate weights through training. Our ConvNet hardware design (Section 4.1) has been specifically designed to allow seamless ConvNet reconfiguration and thus solving different RF problems according to the system's needs.

- (2) *RF-specific learning challenges.* There are a number of key challenges in RF learning that are substantially absent in the CV domain. Among others, we know that RF signals are continuously subject to dynamic (and usually unpredictable) noise/interference coming from various sources. This may decrease the accuracy of the learning model. For example, portions of a QPSK

transmission could be mistaken for 8PSK transmissions since they share part of their constellations. We address the above core design issues with the following intuitions. First, although RF signals are affected by fading/noise, in most practical cases their effect can be considered as constant over small intervals. Second, though some constellations are similar to each other, the transitions between the symbols of the constellations are distinguishable when the waveform is sampled at a higher sampling rate than the one used by the transmitter. Third, convolution operations are equivariant to translation, so they can recognize I/Q patterns regardless of where they occur.

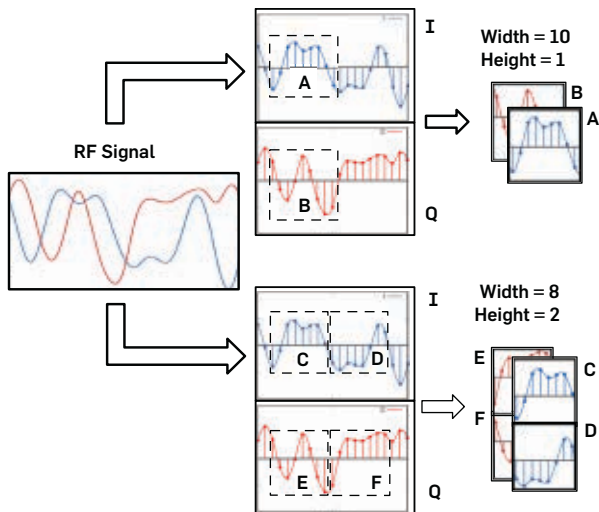
- (3) *RFNet input construction.* By leveraging these key concepts, we can design a learning system that distinguishes waveforms by recognizing transitions in the I/Q complex plane regardless of where they happen, by leveraging the shift-invariance property of convolutional layers. More formally, let us consider a discrete-time complex-valued I/Q sequence $s[k]$, where $k \geq 0$. Let us consider $M = W \cdot H$ consecutive I/Q samples $s[j]$, $0 \leq j \leq W \cdot H$, where W and H are the *width* and *height* of the input tensor. The input tensor \mathcal{T} , of dimension $W \times H \times 2$, is constructed as follows:

$$\mathcal{T}[r, c, d] = \text{Re} \{s(r \cdot W + c)\} \cdot (1-d) + \text{Im}\{s[r \cdot W + c]\} \cdot d, \quad (1)$$

where $d \in \{0, 1\}$, $0 \leq r \leq H$, $0 \leq c \leq W$

By construction, it follows that $\mathcal{T}[r + 1, c] = s[(r + 1) \cdot W + c] = s[r \cdot W + c + W]$, meaning that (i) I/Q samples in adjacent columns will be spaced in time by a factor of 1, and (ii) I/Q samples in adjacent rows will be spaced in time by a factor of W ; moreover, (iii) our input tensors have depth equal to 2, corresponding to the I and Q data, respectively, which will allow the *RFNet* filters to examine each element of the input tensor *without decoupling the I and Q components of the RF waveform*. Figure 3 depicts an example of a 2×4 and 1×3 filters operating on a waveform.

Figure 3. How *RFNet* constructs tensors from I/Q samples.



4. PolymoRF: HW/SW ARCHITECTURE

This section presents the hardware and driver design and implementation of our *PolymoRF* system. We discuss the design, hardware implementation, and main operations of *RFNet* in Section 4.1 (Figure 4).

4.1. RFNet: Architecture and operations

- (1) *Design constraints.* One of the core design issues to address is ensuring that *the same RFNet circuit can be reused for multiple learning problems and not just one architecture*. For example, the wireless node might want to classify only specific properties of an RF waveform, for example, classify only modulation since the FFT size is already known. This requires reconfigurability of the model parameters, as the device’s hardware constraints may not be able to accommodate multiple learning architectures. In other words, we want *RFNet* to be able to operate with a different set of filters and weight parameters according to the circumstances. For this reason, we have used high-level synthesis (HLS) to design a library that translates a Keras-compliant *RFNet* into an FPGA-compliant circuit. HLS interprets an algorithmic description of a desired behavior (e.g., C/C++) and creates a model written in hardware description language (HDL) that can be executed by the FPGA.²⁰
- (2) *Circuit design.* Figure 5 shows a block scheme of our HLS-based *RFNet* circuit and its main interactions with the CPU and other FPGA components. We also provide an example with some numbers to ease presentation. The main feature of our *RFNet* implementation is its modularity—indeed, the circuits implementing each layer are *independent from each*

Figure 4. *RFNet* captures small-scale I/Q pattern sequences.

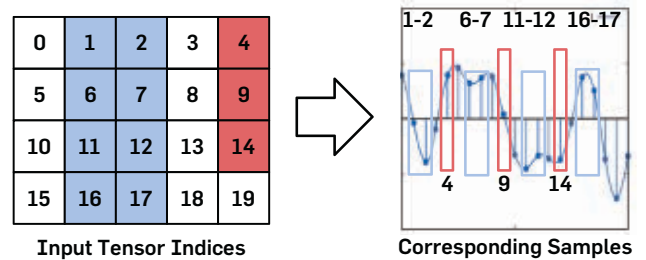
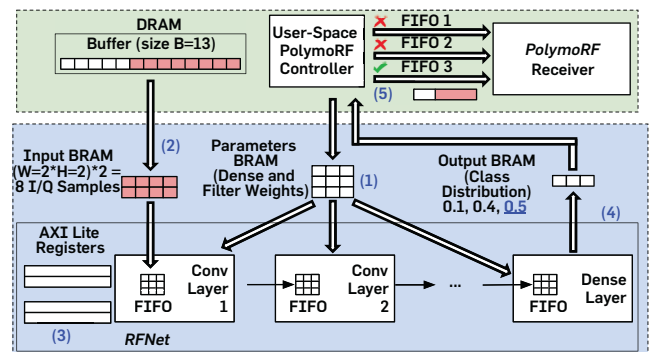


Figure 5. Block scheme of *PolymoRF*’s learning circuit.



other, which allows for ease of parallelization and transition from HLS to HDL. Consecutive layers in *RFNet* exchange data through high-speed AXI-Stream interfaces that then store the results of each layer in a FIFO, read by the next layer. Our architecture uses a 32-bit fixed point representation for real numbers, with 10 bits dedicated to the integer portion. We chose fixed point instead of floating point to decrease drastically computation and hardware architecture complexity, as we do not need the precision of floating point arithmetic. Another key advantage of our implementation is that it clearly separates the computation from the parameters, which allows for *seamless real-time reconfigurability*. This is achieved by writing the parameters in a BRAM accessible by the CPU and by the *RFNet* circuit.

- (3) *Main operations*. The first operation is to write the *RFNet*'s parameters into a BRAM through the user-space PolymoRF controller (step 1). These parameters are the weights of the convolutional layer filters and the weights of the dense layers. Since we use fixed point architecture, each parameter is converted into fixed point representation before being written to the BRAM. As soon as a new input buffer B (of size 13 in our example) has been replenished, the controller writes the *RFNet* input (the first 8 I/Q samples in our example) into the input BRAM (step 2). *RFNet* operations are then started by writing into an AXI-Lite register (step 3) through a customized kernel-level Linux driver. Once the results have been written in the output BRAM (step 4), *RFNet* writes an acknowledgement bit into another AXI-Lite register, which signals the controller that the output is ready. Then, the controller reads the output (in our example, class 3 has the highest probability) and sends the entire buffer B through a Linux FIFO to the PolymoRF receiver (step 5), which is currently implemented in Gnuradio software. The receiver has different FIFOs, each for a parameter set. Whenever a FIFO gets replenished, the part of the flow graph corresponding to that parameter set activates and demodulates the I/Q samples contained in the buffer B . Notice that for efficiency reasons the receiver chains do not run when the FIFO is empty, therefore only one receiver chain can be active at time.

5. EXPERIMENTAL RESULTS

We first discuss details on our *PolymoRF* prototype in Section 5.1, and then discuss the data collection and training process in Section 5.2. We then investigate the performance of *RFNet* in Section 5.3 on a single-carrier system. Then, we implement and test the throughput performance on a multi-carrier polymorphic OFDM system in Section 5.4. Finally, we report the latency and hardware performance of *PolymoRF* in Section 5.5.

5.1. Prototype and experimental setup

Our prototype is entirely based on off-the-shelf equipment. Specifically, we use a Xilinx Zynq-7000 XC7Z045-2FFG900C

system-on-chip (SoC), which is a circuit integrating CPU, FPGA, and I/O all on a single substrate.⁹ We chose an SoC since it provides significant flexibility in the FPGA portion of the platform, thus allowing us to fully evaluate the trade-offs during system design. Moreover, the Zynq-7000 fully supports embedded Linux, which in effect makes the ZC706 a good prototype for a wireless platform. Our Zynq-7000 contains two ARM Cortex-A9 MPCore CPUs and a Kintex-7 FPGA,²¹ running on top of a Xilinx ZC706 evaluation board.²²

For both intra-FPGA and FPGA-CPU data exchange, we use the *Advanced eXtensible Interface* (AXI) bus specification.²³ In the AXI standard, the data is exchanged during *read* or *write transactions*. In each transaction, the *AXI master* is charged with initiating the transfer; the *AXI slave*, in turn, is tasked with responding to the AXI master with the result of the transaction (i.e., success/failure). An AXI master can have multiple AXI slaves and vice versa, according to the specific FPGA design. Multiple AXI masters/slaves can communicate with each other by using *AXI interconnects*. Specifically, *AXI-Lite* is used for register access and configures the circuits inside the FPGA, while AXI-Stream is used to transport high-bandwidth streaming data inside the FPGA. AXI-Full is instead used by the CPU to read/write consecutive memory locations from/to the FPGA.

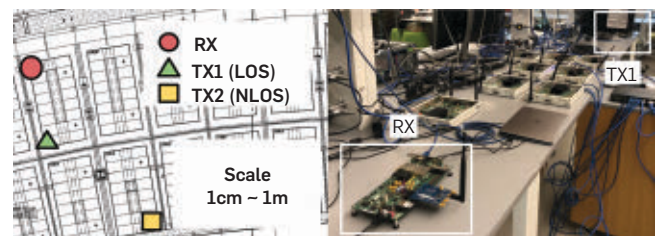
To study *PolymoRF* under realistic channel environments, we have used the experimental setup shown in Figure 6. These scenarios investigate a line-of-sight (LOS) configuration where the transmitter is placed approximately 3 m from the receiver, and a challenging non-line-of-sight (NLOS) channel condition where the transmitter is placed at 7 m from the receiver and in the presence of several obstacles between them. Thus, the experiments were performed in a contested wireless environment with severe interference from nearby Wi-Fi devices as well as multipath effect.

5.2. Data collection and training process

As far as the data collection and testing process is concerned, we first constructed a ~ 10 GB dataset by collecting waveform data in the line-of-sight (LOS) configuration, and then used this data to train *RFNet* through Keras. Then, we tested our models on live-collected data in both LOS and NLOS conditions. The transmitter radio used was a Zedboard equipped with an AD9361 as RF front-end and using Gnuradio for base-band processing. Waveforms were transmitted at center frequency of 2.432 GHz (i.e., Wi-Fi's channel 5).

To train *RFNet*, we use an ℓ_2 regularization parameter $\lambda = 0.0001$. We also use an Adam optimizer with a learning

Figure 6. (left) Placement of the radios for experimental evaluation; (right) experimental setting.



rate of $l = 10^{-4}$ and categorical cross-entropy as a loss function. All architectures are implemented in Python, on top of the Keras framework and with Tensorflow as the backend engine.

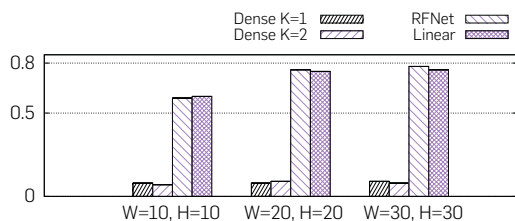
5.3. Single-carrier evaluation

We consider the challenging problem of joint modulation and channel recognition in a single-carrier system where (i) modulation is chosen among BPSK, QPSK, 8PSK, 16-QAM, 32-QAM, and 64-QAM; (ii) spectrum is shifted of 0, 1 KHz, and 2 KHz from its center frequency. Due to space limitations, we only report results on the LOS scenario for the single-carrier scenario and report in Section 5.4 the performance of *RFNet* on the NLOS scenario with the multicarrier OFDM system.

- (1) *Comparison with existing architectures.* We compare *RFNet* to,^{10,12} which is to the best of our knowledge^{10,12} the current state of the art in RF waveform classification using ConvNets. This approach, called for simplicity *Linear*, considers an input tensor of dimension $1 \times W \cdot H \times 2$ and convolutional layers with filters of dimension $1 \times F \times 2$. Thus, the filters in the first convolutional layer perform linear convolution over a set of F consecutive I/Q samples. We attempted to train the architecture in¹², which has $M = 7$ convolutional layers with $C = 64$ filters each and $K = 2$ dense layers with 128 neurons each. However, due to its huge dimensions, we were not able to synthesize this architecture on our test bed. Therefore, we compared *RFNet* with the architecture in,¹⁰ that is, $M = 2$ convolutional layers with $C = 25, 680$ and $K = 1$ with 256 neurons. For fair comparison with *Linear*, we selected the closest input size to ours (i.e., 1×128 vs. $10 \times 10, 1 \times 400$ vs. $20 \times 20, 1 \times 900$ vs. 30×30).

Figure 7 shows the test-set accuracy obtained for a subset of the considered architectures, where *RFNet* was trained with $M = 1$ convolutional layer with $C = 25$ filters, and no dense layer ($K = 0$). The obtained results indicate that traditional dense networks cannot recognize complex RF waveforms, as they attain slightly more accuracy (8%) than the random-guess accuracy (5.5%)—regardless of the number of layers. This is because dense layers are not able to capture localized, small-scale I/Q variations in the input data, which is instead done by convolutional layers. Moreover, Figure 7 indicates that *RFNet* has similar accuracy as obtained by *Linear*, despite using a much

Figure 7. Comparison among *RFNet*, *Dense*, and *Linear*.¹⁰



simpler architecture. This is due to the fundamental difference between how the convolutional layers in *PolymoRF* and *Linear* process I/Q samples.

- (2) *Hyper-parameter evaluation.* We study the impact of the number of convolutional layers M and dense layers K , as well as the input size (W) and filter size (F) on the performance of *RFNet*. Figure 8 shows accuracy as a function of W and H , for hyper-parameters $M = 1, 2$ and $C = 10, 25, 50$. The results conclude that increasing C does improve the performance but up to a certain extent. Indeed, we notice that switching to $C = 50$ does not improve much the performance, especially when $M = 2$. This is because the number of distinguishing I/Q patterns is limited in number among different modulations, and thus, the filters in excess end up learning similar patterns. Furthermore, increasing W and H increases accuracy significantly, since a larger input size allows compensating for the adverse channels/noise conditions. Furthermore, Figure 9 illustrates the impact of K . Figure 9 suggests that the accuracy does not increase when adding a dense layer, regardless of its size, which indicates the correctness of our choice to exclude dense layers.
- (3) *Impact of the sampling rate.* We investigate the impact of the transmitter's sampling rate in Figure 10, where we show the classification accuracy for differ-

Figure 8. (top) Test set classification accuracy vs. input size W/H vs. M , with $K = 0$ (no dense layer); (bottom) confusion matrices as function of M, W , and H .

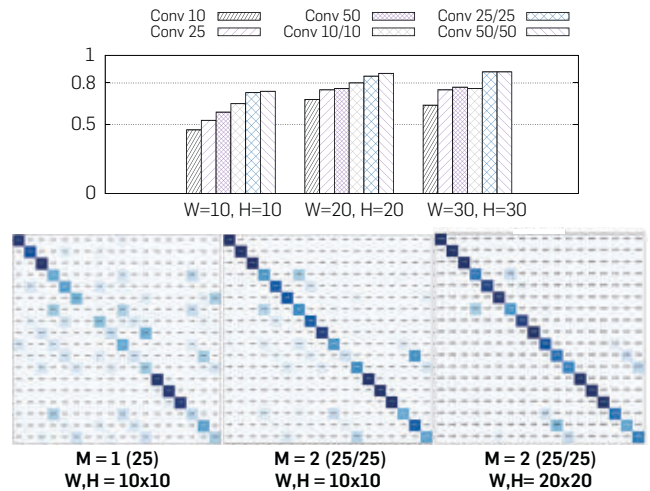
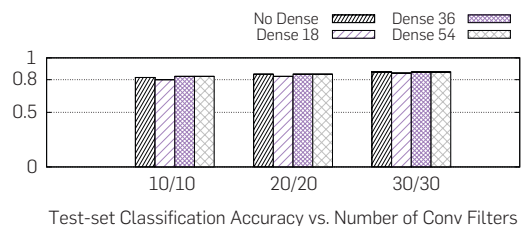


Figure 9. Accuracy vs. number of filters vs. dense layer size.



ent W , H , and C values. We also show the confusion matrices^a for the $W, H = 10, C = 50$ architectures in Figure 11. As expected, these results confirm that the performance of *RFNet* decreases as the transmitter's sampling rate increases. This is because, as shown in Section 3 *RFNet* learns the I/Q transitions between the different modulations. Therefore, as the transmitter's sampling rate increases, the model will have fewer I/Q samples between the constellation points. Indeed, the confusion matrices show that with 5 MS/s the model becomes further confused with QAM constellations, and with 10 MS/s higher-order PSKs and QAMs “collapse” onto the lowest-order modulations.

- (4) *Remarks.* The above results imply that oversampling the signal leads to better modulation classification accuracy. However, we would like to point out that oversampling does not mean that the physical layer has to process more data—indeed, the extra samples can be dropped when going through the demodulation chain, while the oversampled I/Q signal can be forwarded to *RFNet* for classification.

5.4. Multicarrier evaluation

We evaluated *PolyMoRF* on an OFDM system (in short, *Poly-OFDM*) which supports three different FFT sizes (64, 128, and 256) and three different symbol modulations in the FFT bins (BPSK, QPSK, and 8PSK), creating in total a combination of nine different parameter sets that are switched pseudo-randomly by the transmitter. A demo video where the transmitter switches FFT size every 0.5s is available at https://youtu.be/5vf_pb0nvKk. In the

a Class labels are ordered by modulation and frequency shift, that is, from “BPSK, 0 KHz”, “BPSK, 1 KHz”, ... to “64-QAM, 2 KHz”.

Figure 10. Accuracy vs. transmitter's sampling rate.

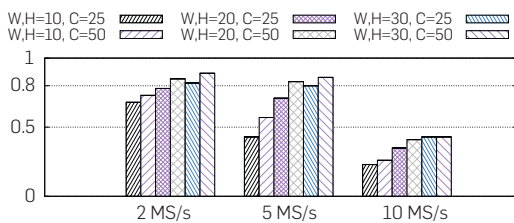
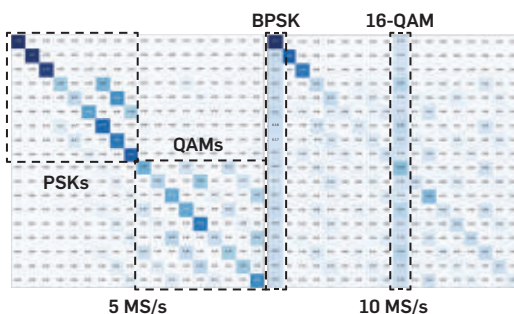


Figure 11. Confusion matrices for transmitter's sampling rate of 5 MS/s and 10 MS/s, $W, H = 10, C = 50$ model.



following, we use the $C = 25, 25, 20 \times 20$, pipelined *RFNet* architecture, which presents latency of about 17 ms (see Section 5.5). In these experiments, we set (i) the transmitter's sampling rate to 5M samples/sec; *PolyMoRF*'s buffer size B to 250k I/Q samples; (iii) the switching time of the transmitter to 250 ms. Thus, *RFNet* is run approximately five times during each switching time.

The most critical aspect to be evaluated is how *Poly-OFDM*, an inference-based system, compares with an ideal system that has perfect knowledge of the modulation and FFT size being used by the transmitter at each time, which we call for simplicity *Oracle*. Although *Oracle* cannot be implemented in practice, we believe this experiment is crucial to understand what is the throughput loss with respect to a system where the physical layer configuration is known a priori. In Figure 12, we show the comparison between *Oracle* and *Poly-OFDM* as a function of the FFT size and the symbol modulation. As we notice, the overall throughput results decrease in the NLOS scenario, which is expected given the impairments imposed by the challenging channel conditions. On the other hand, the results in Figure 12 confirm that *Poly-OFDM* is able to obtain similar throughput performance with that of a traditional OFDM system, obtaining on the average 90% and 87% throughput of that of the traditional system.

5.5. RFNet latency evaluation and comparison

Table 1 compares latency, the number of parameters, and BRAM occupation of *RFNet* vs. a C++ implementation running in the CPU of our test bed. As we can see, *RFNet* consumes at most 34% of the available BRAM of the platform. Moreover, Table 2 shows the comparison between the pipelined version of the ConvNet circuits and the CPU latency, as well as the look-up table (LUT) consumption increase with respect to the unpipelined version. Table 2 concludes that on the average, our parallelization strategies bring close to 60% and 100% latency reduction with respect to the unoptimized and CPU versions, respectively, with a LUT utilization increase of about 7% on the average.

Figure 12. Comparison between *Oracle* and *Poly-OFDM*, (top) LOS and (bottom) NLOS scenarios.

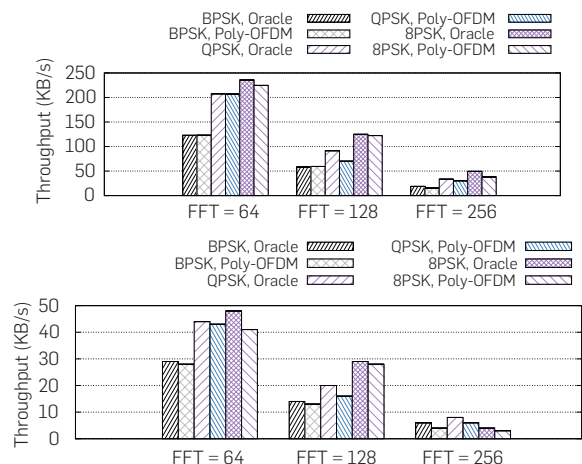


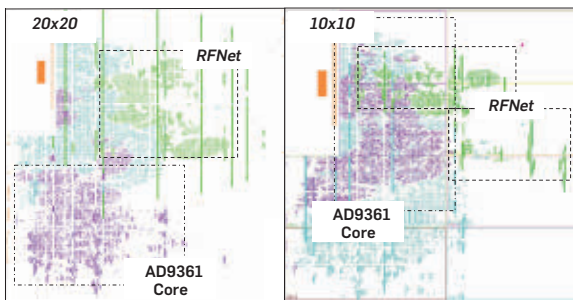
Table 1. Latency/hardware consumption evaluation.

Model	Input	Latency (ms)	Params (k)	BRAM (%)
RFNet C = 25	10 × 10	2.918	~11	1
	20 × 20	26.55	~45	7
	30 × 30	93.35	~81	15
RFNet C = 50	10 × 10	5.835	~23	3
	20 × 20	53.11	~90	14
	30 × 30	233.3	~203	29
RFNet C = 25,25	10 × 10	6.704	~10	2
	20 × 20	38.41	~17	8
	30 × 30	144.9	~34	17
RFNet C = 50,50	10 × 10	21.41	~31	4
	20 × 20	100.3	~40	16
	30 × 30	336.9	~81	34

Table 2. Pipelined vs. CPU latency.

Model	Input	CPU (ms)	Pipelined (ms)	LUT (%)
RFNet C = 25	10 × 10	49.31	1.19	+3
	20 × 20	478.4	8.077	+7
	30 × 30	1592	25.54	+9
RFNet C = 50	10 × 10	106.4	2.381	+6
	20 × 20	934.2	16.15	+12
	30 × 30	3844	63.81	+20
RFNet C = 25,25	10 × 10	122.1	3.959	+1
	20 × 20	677.9	16.29	+4
	30 × 30	2354	49.57	+7
RFNet C = 50,50	10 × 10	363.9	13.51	+2
	20 × 20	1826	48.87	+7
	30 × 30	5728	131.7	+11

Figure 13. PolymoRF FPGA implementations.



To give the reader a perspective of the amount of resources consumed on the FPGA, Figure 13 shows the FPGA implementation of respectively 10×10 and 20×20 *RFNet* model, both pipelined and with $C = 25,25$ architecture, where we highlight and color the resource consumption of *RFNet* with respect to the AD9361 circuitry. Figure 13 indicates that the resource consumption of the *RFNet* circuit is significantly lesser than the AD9361 one in the 10×10 case and becomes comparable with the 20×20 architecture. In any case, the overall resource consumption of our FPGA designs is approximately 50% of the total FPGA resources.

6. RELATED WORK

Learning-based radios are envisioned to be able to automatically infer the current spectrum status in terms of occupancy,¹⁷ interference,² and malicious activities.⁵ Most of the existing work is based on low-dimensional machine learning,^{4, 16, 24} which requires the cumbersome manual extraction of very complex, *ad hoc* features from the waveforms. For this reason, deep learning has been proposed as a viable alternative to traditional learning techniques.⁷ The key problem of RF modulation recognition through deep learning has been extensively investigated.^{6, 11, 12, 18, 19} The seminal work by O’Shea et al.¹² proposed ConvNets-based to address the issue. However, the authors do not address the issue of what to do with the inferred RF information. Moreover, the aforesaid work proposes models leveraging a significant number of parameters, thus ultimately not applicable to real-time RF settings. Recently, Restuccia and Melodia¹³ have demonstrated the need for real-time hardware-based RF deep learning. However, the main limitation of this study¹³ is that it focused on the learning aspect only, ultimately not addressing the problem of connecting real-time inference with receiver reconfigurability.

7. CONCLUSION

This paper has proposed *PolymoRF*, a prototype that can be reused to develop and test novel polymorphic wireless communication systems. One of the key insights brought by our experimental evaluation is that the RF channel may impact the performance of *RFNet* to a significant extent. To this end, we can (i) train different learning models for different channels and reconfigure the weights of *RFNet* in the FPGA accordingly; and (ii) apply small, controlled modifications to the RF signal at the transmitter’s side to compensate for the current RF channel condition. Another core aspect is the impact of polymorphism on the effectiveness of smart jamming attacks. We are conscious that the aforesaid issues are definitely worth investigating; however, they deserve separate papers and are the subject of our ongoing work.

Acknowledgments

This work is supported in part by the Office of Naval Research (ONR) under contracts N00014-18-9-0001. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements of the ONR or the U.S. government. □

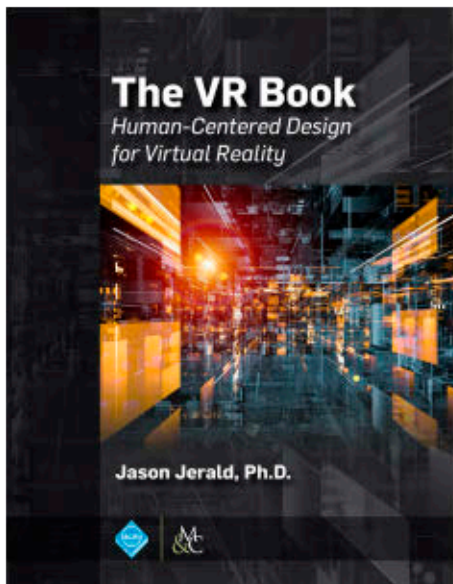
References

- Analog Devices Incorporated. AD9361 RF agile transceiver data sheet, 2018. <http://www.analog.com/media/en/technical-documentation/data-sheets/AD9361.pdf>.
- Chen, Y., Oh, H.-S. A survey of measurement-based spectrum occupancy modeling for cognitive radios. *IEEE Commun. Surv. Tutor* 18, 1 (2016), 848–859.
- Cisco Systems. Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021 white paper, 2017. <http://tinyurl.com/zzo6766>.
- Ghodeswar, S., Poonacha, P.G. An SNR estimation based adaptive hierarchical modulation classification method to recognize M-ary QAM and M-ary PSK signals. In *Proceedings of International Conference on Signal Processing, Communication and Networking (ICSCN)* (2015).
- Jin, X., Sun, J., Zhang, R., Zhang, Y., Zhang, C. SpecGuard: Spectrum misuse detection in dynamic spectrum access systems. *IEEE Trans. Mob. Comput* 17, 12 (Dec 2018), 2925–2938.
- Kulin, M., Kazaz, T., Moerman, I., Poorter, E.D. End-to-end learning

- from spectrum data: A deep learning approach for wireless signal identification in spectrum monitoring applications. *IEEE Access* 6, 2018, 18484–18501.
7. Mao, Q., Hu, F., Hao, Q., Deep learning for intelligent wireless networks: A comprehensive survey. *IEEE Commun. Surv. Tutor* 20, 4 (2018), 2595–2621.
 8. Mitola, J., Maguire, G.Q. Cognitive radio: Making software radios more personal. *IEEE Pers. Commun* 6, 4 (1999), 13–18.
 9. Molanes, R.F., Rodríguez-Andina, J.J., Fariña, J. Performance characterization and design guidelines for efficient processor – FPGA communication in cyclone V FPGAs. *IEEE Trans. Ind. Electron.* 65, 5 (2018), 4368–4377.
 10. O’Shea, T.J., Corgan, J., Clancy, T.C. Convolutional radio modulation recognition networks. In *International Conference on Engineering Applications of Neural Networks* (2016), Springer, 213–226.
 11. O’Shea, T.J., Hoydis, J. An introduction to deep learning for the physical layer. *IEEE Trans. Cogn. Commun. Netw* 3, 4 (2017), 563–575.
 12. O’Shea, T.J., Roy, T., Clancy, T.C. Over-the-air deep learning based radio signal classification. *IEEE J. Sel. Top. Signal Process* 12, 1 (2018), 168–179.
 13. Restuccia, F., Melodia, T. Big data goes small: Real-time spectrum-driven embedded wireless networking through deep learning in the RF loop. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, (2019).
 14. Restuccia, F., Melodia, T. DeepWiERL: Bringing deep reinforcement learning to the internet of self-adaptive things. *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2020.
 15. Restuccia, F., Melodia, T. Physical-layer deep learning: Challenges and applications to 5G and beyond. arXiv:2004.10113 (2020).
 16. Shi, Q., Karasawa, Y. Automatic modulation identification based on the probability density function of signal phase/ *IEEE Trans. Commun* 60, 4 (April 2012), 1033–1044.
 17. Subramaniam, S., Reyes, H., Kaabouch, N. Spectrum occupancy measurement: An autocorrelation based scanning technique using USR. In *Proceedings of IEEE Annual Wireless and Microwave Technology Conference (WAMICON)* (2015).
 18. Wang, T., Wen, C.-K., Wang, H., Gao, F., Jiang, T., Jin, S. Deep learning for wireless physical layer: Opportunities and challenges. *China Commun* 14, 11 (2017), 92–111.
 19. West, N.E., O’Shea, T.J. Deep architectures for modulation recognition. In *Proceedings of IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)* (2017).
 20. Winterstein, F., Bayliss, S., Constantinides, G.A. High-level synthesis of dynamic data structures: A case study using Vivado HLS. In *Proceedings of International Conference on Field-Programmable Technology (FPT)* (2013).
 21. Xilinx Inc. Zynq-7000 SoC data sheet: Overview, 2018. <https://www.xilinx.com/support/documentation/>
 22. Xilinx Inc. ZC706 evaluation board for the Zynq-7000 XC7Z045 all programmable SoC user guide, 2018. https://www.xilinx.com/support/documentation/boards_and_kits/zc706/ug954-zc706-eval-board-xc7z045-ap-soc.pdf.
 23. Xilinx Inc.. AXI reference guide, UG761 (v13.1) March 7, 2011 (2011). https://www.xilinx.com/support/documentation/ip_documentation/ug761_axi_reference_guide.pdf0.0.0.0.
 24. Xiong, W., Bogdanov, P., Zheleva, M. Robust and efficient modulation recognition based on local sequential IQ features. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)* (2019).
 25. Yu, Y., Wang, T., Liew, S.C. Deep-reinforcement learning multiple access for heterogeneous wireless networks. *IEEE J. Sel. Areas Commun.* 37, 6 (2019), 1277–1290.

Francesco Restuccia and Tommaso Melodia (frestuc, melodia}@northeastern.edu), Institute for the Wireless Internet of Things, Northeastern University, Boston, MA, USA

© 2022 ACM 0001-0782/22/9 \$15.00



Without a clear understanding of the human side of virtual reality, the experience will always fail.

“Dr. Jerald has recognized a great need in our community and filled it. The VR Book is a scholarly and comprehensive treatment of the user interface dynamics surrounding the development and application of virtual reality. I have made it a required reading for my students and research colleagues. Well done!”

- Professor Tom Furness, University of Washington VR Pioneer and Founder of HIT Lab International and the Virtual World Society



ISBN: 978-1-970001-12-9 DOI: 10.1145/2792790

<http://books.acm.org>

<http://www.morganclaypoolpublishers.com/vr>

Technical Perspective

The Effectiveness of Security Measures

By Nicolas Christin

IN THE LATE 1990S, we came to the realization that users were central to computer and information security. Ross Anderson famously argued that “the threat model was completely wrong” when referring to our historical focus on securing technical components while ignoring possible human mistakes. A large and growing body of research has subsequently attempted to study how people face computer security challenges. Studies in the adjacent field of information privacy revealed that user behavior is complex. People may profess caring about their privacy, but frequently end up making decisions that prove costly, for example, due to limited information or to behavioral biases that lead them to miscalculate long-term risks.

Measuring security behavior turns out to be even more difficult than measuring privacy preferences and actions but imagine for a second that we had the ability to do so. For instance, we could examine the practical relevance of the following well-known, but rarely evaluated, security advice: updating software frequently, browsing reputable websites, using encryption whenever possible, and trying to avoid operating systems that are too common and targeted by villains.

How should we go about it? Carefully controlled experiments, such as asking users to come to a lab and run through a set of predetermined activities, are unlikely to display the potentially risky behaviors in which people engage in the comfort of their home. On the other hand, field measurements, in which users are left to their own devices and are passively observed, without direct interaction, are cumbersome to set up for a range of reasons. Meaningful data collection must be done at scale to exhibit statistical patterns; such studies are highly sensitive, and researchers must take the greatest care in preserving

their participants’ privacy; and, perhaps most importantly, getting access to such user data is nearly impossible without access to a large infrastructure provider.

In the following paper, DeKoven et al. manage to overcome these challenges. The authors leverage their campus network to get access to 15,000 computers in university dorms. They build a remarkable infrastructure to acquire this network traffic, anonymize it to safeguard user privacy, and whittle it down to levels suitable for analysis. The scale of the experiment is particularly daunting. The authors collect approximately 4Gbps to 6Gbps of traffic for six months, which corresponds to roughly 100 petabytes of data in aggregate. A clever idea in the paper is to partially repurpose intrusion detection systems such as Bro/Zeek and Suricata to extract relevant information from these large volumes. In particular, the authors ingeniously create a set of fingerprints to automatically detect operating systems and installed software, and to track individual user behavior across many different, disjoint sessions, with potentially different IP addresses.

They analyze this data to provide a unique perspective into how users choose to implement common security advice in practice. The authors notably discover that there is no difference in system update frequency between users, regardless of whether their computer ends up being infected by malware. However, users tend to more frequently update their Web browsers and Flash software after having been compromised. The main and perhaps most interesting outcome of this part of the study is the absence of very strong, obvious correlations between the likelihood of compromise and lack of adherence to generally accepted security practices.

However, there is one important ex-

ception: Potentially risky Web browsing behavior tends to lead to riskier security outcomes. This result is particularly meaningful because it echoes what other studies run with different users, under different circumstances, had also observed. Going beyond this paper, Web browsing behavior appears to be an important determinant of the potential risk of security compromise, which in turn justifies the large amount of effort we should continue to put in securing the Web.

But, perhaps, the main contribution of this paper lies more in asking questions than providing definitive answers. This research clearly shows that while passive measurements are powerful in helping us establish correlations, they are much more limited when it comes to exhibiting causal relationships. For instance, this paper shows that users of the anonymous Tor network are more likely to get in trouble than others—but is it because users erroneously believe that Tor provides increased protection against many security compromises, or because the malware itself installs Tor to communicate anonymously?

Moving forward, to improve security practices and distinguish between folk remedies and advice rooted in empirical evidence, we should focus on understanding causal relationships between user actions, exposure to vulnerabilities, and security compromises. Combining large-scale passive measurements as described by the authors with finer-grained timeline reconstructions and user interviews could help reach these objectives. As this paper clearly demonstrates, it is a highly ambitious, technically challenging, but overall worthy goal. 

Nicolas Christin is a professor in the School of Computer Science and in the Department of Engineering and Public Policy at Carnegie Mellon University, Pittsburgh, PA, USA.

Copyright held by author.

Measuring Security Practices

By Louis F. DeKoven, Audrey Randall, Ariana Mirian, Gautam Akiwate, Ansel Blume, Lawrence K. Saul, Aaron Schulman, Geoffrey M. Voelker, and Stefan Savage

Abstract

Users are encouraged to adopt a wide array of technologies and behaviors to reduce their security risk. However, the adoption of these “best practices,” ranging from the use of antivirus products to keeping software updated, is not well understood, nor is their practical impact on security risk well established. To explore these issues, we conducted a large-scale measurement of 15,000 computers over six months. We use passive monitoring to infer and characterize the prevalence of various security practices as well as a range of other potentially security-relevant behaviors. We then explore the extent to which differences in key security behaviors impact the real-world outcomes (i.e., that a device shows clear evidence of having been compromised).

1. INTRODUCTION

Our existing models of security all rely on end users to follow a range of best practices; for example, the rapid installation of security updates to patch vulnerabilities. Implicit in this status quo is the recognition that security is not an intrinsic property of today’s systems, but is a byproduct of making appropriate choices—choices about what security products to employ, choices about how to manage system software, and choices about how to engage (or not) with third-party services on the Internet.

However, establishing the value provided by these practices is underexamined at best. First, we have limited empirical data about which security advice is adopted in practice. Users have a plethora of advice to choose from, highlighted by Reeder et al.’s recent study of expert security advice, whose title—“152 Simple Steps to Stay Safe Online”—underscores both the irony and the variability in such security lore.²⁰ A second, more subtle issue concerns the efficacy of such practices when followed: Do they work? Here the evidence is also scant. Even practices widely agreed upon by Reeder’s experts, such as keeping software patched, are not well justified beyond a rhetorical argument. In fact, virtually all established “security best practices” are of this nature, and as summarized by Herley, their “benefit is largely speculative or moot.”¹⁰

This paper seeks to make progress on both issues—the prevalence of popular security practices and their relationship to security outcomes—via the longitudinal empirical measurement of a large population of computer devices. In particular, we perform a preliminary study based on monitoring the online behavior of 15,291 independently administered desktop/laptop computers. We identify per-device security *behaviors*: what software they are running (e.g., antivirus products, password managers, etc.), how

is the software patched, and what is their network usage (e.g., does the machine contact file sharing sites), etc., as well as concrete security *outcomes* (i.e., whether a particular machine becomes compromised). In the course of this work, we describe three primary contributions:

- A large-scale passive feature collection: we develop and test a large dictionary of classification rules to *infer* software state on monitored machines (e.g., that a machine is using an antivirus of a particular brand).
- An outcome-based analysis: we show how to use a concrete evidence of security outcomes (operational security logs and network intrusion detection alerts) to identify the subset of machines in our dataset that are truly compromised (and not merely exhibiting “risky” behavior).
- Prevalence and impact of security practices: for our user population, we establish the prevalence of a range of popular security practices as well as how these behaviors relate to security outcomes. We specifically explore the hypotheses that a range of existing “best practices” and “bad behaviors” are correlated with host compromise.

Using this approach, we identify a number of “bad behaviors” that are positively correlated with host compromise, but find few “best practices” exhibiting strong negative correlations that would support their clear value in improving end user security.

2. BACKGROUND

Ours is far from the first research to empirically explore the security risks associated with user behavior. Although space does not allow a full exploration of related work, we highlight representative examples of past major efforts here.

Among the earliest of these studies is the work of Carlinet et al. which also used passive network analysis (albeit at a much smaller scale) to relate machine characteristics (e.g., such as operating system type) to security alerts. More recently, other researchers have specifically investigated how a user’s Web browsing habits reveal risk factors, notably Canali et al.’s⁴ study of antivirus telemetry (100,000 users) and Sharif et al.’s²² analysis of 20,000 mobile users.

The original version of this paper is entitled “Measuring Security Practices and How They Impact Security” and was published in *Proceedings of the Internet Measurement Conference, 2019, ACM*.

Both found that frequent, nighttime, and weekend browsing activity are correlated with security risk.

Another important vein of research has correlated poor software update habits with indicators of host compromise. Kahn et al.¹³ used passive monitoring to demonstrate a positive correlation between infection indicators and lack of regular updating practice over a population of 5000 hosts. At a larger scale, Bilge et al.³ used antivirus logs and telemetry from over 600,000 enterprise hosts to retrospectively relate such software updating practices to subsequent infections.

Finally, there is an extensive literature on the human factors issues involved in relating security advice to users, the extent to which the advice leads to changes in behaviors, and how such effects are driven by both individual self-confidence and cultural norms.^{8, 18, 19, 21, 23}

3. METHODOLOGY

Our measurement methodology uses passive network traffic monitoring to infer the security and behavioral practices of devices within a university residential network. In this section, we first focus on the technical aspects of our data collection methodology and then discuss some of its attendant challenges and limitations.

3.1. Network traffic processing

The first stage of our system takes as input 4–6 Gbps of raw bidirectional network traffic from the campus residential network, and outputs logs of processed network events at the rate of millions of records per second. As part of this stage, campus IP addresses are anonymized and, to track the contemporaneous mapping of IP addresses to a device's MAC addresses, this stage also collects and compatibly anonymizes contemporaneous dynamic host configuration protocol (DHCP) syslog traffic.

3.1.1. Residential network traffic

As shown in the network traffic processing stage of Figure 1, our server receives network traffic mirrored from a campus Arista switch using two 10G fiber optic links. In addition to load balancing, the switch filters out high-volume traffic from popular content delivery networks (CDNs) (e.g., Netflix, YouTube, Akamai, etc.), resulting in a load of 4–6 Gbps of traffic on our server.

Although intrusion detection systems (IDSes) are typically used for detecting threats and anomalous network behavior, we use Zeek to convert network traffic into logs, because it is extensible, discards raw network traffic as soon as a connection is closed (or after a timeout), and is able to parse numerous network protocols. We also customize the Bro output logs to record only the information needed to identify security practice and behavioral features.

Every 30 minutes Bro rotates the previous logs through an address anonymization filter that encrypts the campus IP addresses. At this stage of processing, the logs contain IP addresses and not MAC addresses because the DHCP traffic is not propagated to our network vantage point. After being so anonymized, the logs are rotated across the DMZ to another server for further processing (Section 3.2).

3.1.2. DHCP traffic

The server also runs a syslog collector that receives forwarded DHCP traffic from the residential network's DHCP servers. DHCP dynamically provides an IP address to a device joining the network. The IP address is leased to the device (by MAC address) for a specified duration, typically 15 minutes. Since we need to track a device's security and behavioral practices for longtime periods, we utilize this IP-to-MAC mapping in later processing.

Similar to the Bro IDS logs, every 30 minutes we process the previous DHCP traffic into a (MAC address, IP address, starting time, lease duration) tuple. Then, the entire IP address and the identified lower 24-bits of the MAC address are encrypted using a similar address anonymization filter. The anonymized DHCP logs are then rotated across the DMZ to the Log Decoration server.

3.2. Log Decoration

The second stage takes as input these intermediate network events and DHCP logs, and processes them further to produce a single stream of network events associated with a (anonymized) device's MAC addresses and domain names.

Associating flows to devices. Our goal is to model device behavior based upon network activity over longtime spans. Although we identify unique devices based upon their MAC address, the network events that we collect have dynamically assigned IP addresses. As a result, we build a dynamic IP address assignment cache to map IP-based network events to a specific device's MAC addresses.

Associating flows to domains. When using network activity to model a device's behavior, it is useful to know the domain name associated with the end points devices that are communicating with (e.g., categorizing the type of Website being visited). We also extract the registered domain and top-level domain (TLD) from each fully qualified domain name using the Public Suffix List.¹⁵ Again, because the network events we observe use IP addresses, we must map IP addresses to domain names. And because the mapping of domain name system (DNS) names to IP addresses also changes over time, we also dynamically track DNS resolutions as observed in the network to map network events to the domain names involved.

User agent. We parse HTTP user agent strings using the open-source ua-parser library. From the user agent string, we extract browser, operating system (OS), and device information when present.

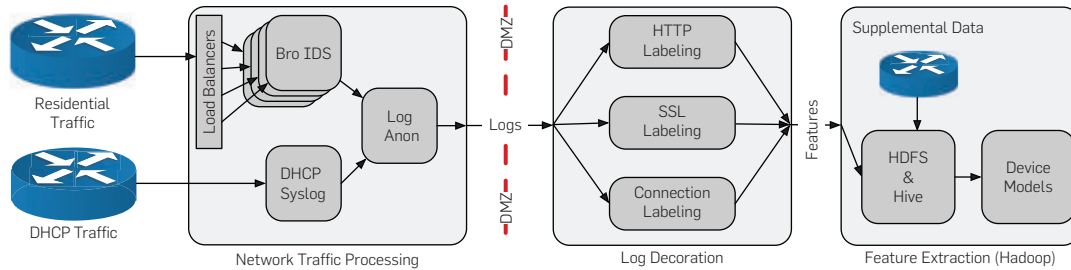
3.3. Feature extraction

In the final stage of our system, we store the log events in a Hive database and process them to extract a wide variety of software and network activity features associated with the devices and their activities as seen on our network. The last critical feature is device outcomes: knowing when a device has become compromised. We derive device outcomes from a log of alerts from a campus IDS appliance, and also store that information in our database.

3.3.1. Software features

To identify the features describing application use on devices,

Figure 1. System architecture. Network traffic is first processed into logs and its addresses are anonymized. The next stage replays the network traffic logs to extract further information and label each connection with (also anonymized) MAC address information. The decorated logs are then stored in Hive where they are labeled with security incidents, security practice features, and behavioral features. Lastly, device models are created for analysis.



we crafted custom network traffic signatures to identify application use (e.g., a particular peer-to-peer client) as well as various kinds of application behavior (e.g., a software update). To create our network signatures, we use virtual machines instrumented with Wireshark. We then manually exercise various applications and monitor the machine’s network behavior to derive a unique signature for each application. Fortunately most applications associated with security risk frequently reveal their presence when checking for updates. In total, we develop network signatures for 68 different applications, including OSs. For a subset of applications, we are also able to detect the application’s version. Knowing application versions allows us to compare how fine-grained recommended security practices (i.e., updating regularly) correlate with device compromise.

Antivirus software. Using an antivirus software is virtually always recommended. We created network signatures for 12 popular antivirus products, seven of which were recognized as offering the “Best Protection” for 2019.¹⁶

Operating system. We created six signatures to identify the OSes running on devices. As regular OS updating is a popular recommended security practice, we also created signatures to detect OS updates. Although Windows and Mac OS operating system updates are downloaded over a content delivery network (CDN) that is removed from the network traffic before reaching our system (Section 3.1), we can use the OS version information from the host header and User-Agent string provided in HTTP traffic to infer that updates have taken place.

Applications. Through a combination of network and User-Agent string signatures, we detect 41 applications, including those commonly perceived as risky such as Adobe Flash Player, Adobe Reader, Java, Tor, peer-to-peer (P2P) applications, and more. We also detect other popular applications, including browsers, Spotify, iTunes, Outlook, Adobe AIR, etc.

Password managers. As password managers are frequently recommended to avoid collateral damage of leaked passwords, we also crafted network signatures for nine popular password managers.⁵

3.3.2. Network activity

We track a wide variety of network activity features to quantitatively measure the protocols used (e.g., HTTP, and HTTPS), the categories of sites visited (e.g., file sharing

services), when devices are most active, etc. In doing so, we implement a set of features similar to those used by Canali et al.⁴ and Sharif et al.²² that focused on the Web browsing activity. As our dataset also includes traffic beyond HTTP, we can measure additional behaviors (e.g., remote DNS resolver usage, HTTPS traffic usage, etc.).

Content categorization. We use the IAB Tech Lab Content Taxonomy to categorize every registered domain in our dataset.¹² The domain categorization was generously provided by Webshrinker.²⁵ The IAB taxonomy includes 404 distinct domain categories.²⁴ We use the domain categorization to measure the fraction of unique domains each device accesses in a specific category. We also built a list of file hosting sites, and URL shortening services that we use to identify when a device accesses these types of services.

Usage patterns. We also develop a number of behavioral features that describe the quantities of HTTP and HTTPS traffic in each TLDs, and the number of network requests made. Additionally, we develop features that quantify customized or nonstandard behaviors such as the use of remote DNS resolvers, and the proportions of HTTP requests made directly to IP addresses (instead of a domain name).

3.3.3. Detecting security incidents

To identify compromised devices (i.e., ones with a security incident), we use alerts generated by a campus network appliance running the Suricata IDS. The campus security system uses deep packet inspection with an industry-standard malware rule set to flag devices exhibiting the post-compromise behavior.¹⁷

3.4. Ethical considerations and limitations

Having described our measurement methodology in considerable detail, we now consider the risks it presents—both to the privacy of network users and to the validity of conclusions drawn from these measurements.

Protecting user privacy. Foremost among the risks associated with the passive measurement approach is privacy. Even with the prevalence of encrypted connections (e.g., via TLS), processing raw network data is highly sensitive. From an ethical standpoint, the potential benefits of our research must be weighed against the potential harms from any privacy violations. In engaging with this question—and developing controls for privacy risk—we involved a broad range

of independent campus entities such as our institutional review board (IRB), the campus-wide cybersecurity governance committee, and our network operations and cybersecurity staff. Together, these organizations provided necessary approvals, direction, and guidance in how to best structure our experiment, and provided a strong support for the goals of our research. The campus security group has been particularly interested in using our measurements to gain insight into the security risks of devices operating on their network; indeed, during the course of our work, we have been able to report a variety of unexpected and suspicious activities to campus for further action.

Operationally, we address privacy issues through minimization, anonymization, and careful control over data. First, as soon as each connection has been processed, we discard the raw content and log only metadata from the connection (e.g., a feature indicating that device X is updating antivirus product Y). Thus, the vast majority of data is never stored. Next, for those features we do collect, we anonymize the campus IP and the last 24-bits of each MAC address, using a keyed format-preserving encryption scheme.² Thus, we cannot easily determine the identity of which machine generated a given feature and, as a matter of policy, we do not engage in any queries to attempt to make such determinations via reidentification. Finally, we use a combination of physical and network security controls to restrict access to both monitoring capabilities and feature data to help ensure that outside parties, not bound by our policies, are unable to access the data or our collection infrastructure. Thus, the server processing raw network streams is located in a secure campus machine room with restricted physical access, only accepts communications from a small static set of dedicated campus machines, and requires multi-factor authentication for any logins. Moreover, its activity is itself logged and monitored for any anomalous accesses. We use similar mechanisms to protect the processed and anonymized feature data although these servers are located in our local machine room. The feature dataset is only accessible to the members of our group, subject to IRB and campus agreements, and will not (and cannot) be shared further.

Limitations of our approach. In addition to privacy risk, it is important to document the implicit limitations of our study arising from its focus on a residential campus

Table 1. Dataset characterization. Note that our network vantage point provides DNS requests from the local resolver, which includes DNS traffic from devices in this paper as well as other devices using the university's networks.

Name	Value
Date range	June 2018–December 2018
Total filtered devices	15,291
DNS connections	17.1B
Non-DNS connections	1.92B
Total connections	19B
Outbound bytes	38.4TB
Inbound bytes	720TB
Total bytes	758TB

population—primarily undergraduates—as well as the use of a particular IDS and rule set to detect security incidents.

It is entirely possible that the behavioral modes of this population, particularly with respect to security, are distinct from the older, less affluent or more professional cohorts. This population bias is also likely to impact time-of-day effects, as well as the kinds of hardware and software used. Additionally, the security incidents we consider rely on the Suricata IDS, commercial network traffic signatures, and security-related network usage requirements of our university environment (e.g., residential students are nominally required to have antivirus software installed on their devices before connecting). It is entirely possible that these incident detection biases also influence the behaviors and software applications that correlate with device compromise. Thus, if our same methodology employed in other kinds of networks, serving other populations, or using different security incident detection techniques, it is possible that the results may differ. For this reason, we hope to see our measurements replicated in other environments.

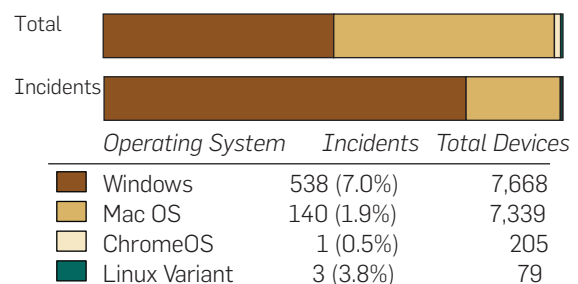
4. DATASET

We analyze six months of data from our passive network traffic processing system from June 2018 to December 2018. In this section, we describe our approach for identifying the laptop and desktop devices for use in analyzing security risk factors, and determining the dominant OS of devices used in our analysis. In the end, our dataset consists of 15,291 devices and Table 1 characterizes our dataset in terms of its traffic.

4.1. Device filtering

The university allows heterogeneous devices on its network, such as personal computers, mobile phones, printers, Internet of Things (IoT) devices, and more. Recommended security practices, however, are commonly offered for laptop and desktop computers, and therefore, we focus our analysis solely on such devices. As a result, we develop techniques to identify laptop and desktop computers among the many other devices on the network. We remove devices that are easily identifiable, and then develop heuristics to filter remaining devices. We remove devices that are not active for a minimum of 14 days, which never provide a major Web browser's User-Agent string, which consistently show a User-Agent string as having a mobile or IoT OS, and devices

Figure 2. Device OS classification after removing IoT and mobile devices: the total number of devices with each OS and the number with a security incident.



whose organizationally unique identifier (OUI) in their MAC address match an IoT vendor.

4.2. Identifying dominant OSs

Since different OSs have different risk profiles, identifying the OS used by a device is an important step. Being able to observe a device’s network traffic makes OS identification an interesting task. A majority of devices are straightforward: using the signatures of OS update events, we can immediately identify a single unambiguous OS for 79.1% of devices.

The remaining devices either have no OS update signatures, or have more than one. For these devices, we use a combination of OS update signatures, OS User-Agent strings, and organizational unique identifier (OUI) vendor name information to identify the dominant OS of a device (e.g., the host OS with virtual machines, Windows if tethering an iPhone, etc.). We assume that devices with an Apple OUI vendor name will be using Mac OS (7.2%). We then use the dominant OS extracted from User-Agent strings to assign an OS (11.5%). The remaining 340 devices (2.1%) have both Windows and Mac OS updates. We choose to assign Windows as the dominant OS in these cases because of strong evidence of device tethering.¹ For each heuristic, we confirmed the labeling by manually checking the traffic profile of a random sample of devices.

5. RECOMMENDED PRACTICES

There are a variety of security practices widely recommended by experts to help users become safer online. Prior work has explored some of these practices in terms of users being exposed to risky Websites.^{4,22} Since our data includes actual security outcomes, we start our evaluation by exploring the correlation of various security practices to actual device compromises in our user population: operating system choice, keeping software up to date, Websites visited, network use, antivirus use, and software used.

5.1. Operating system

Different operating systems have different security reputations, so it is not surprising that experts have recommendations of the form “Use an uncommon OS”.²⁰ Part of the underlying reasoning is that attackers will spend their efforts targeting

devices with most common systems, so using an uncommon operating system makes that device less of a target.

In terms of device compromise, as with previous work and experience, such advice holds for our user population as well. Using the OS classification method described in Section 4.2, Figure 2 shows the number of devices using major operating systems and the number of each that was compromised during our measurement period. Most devices use Windows and Mac OS, split nearly equally between the two. The baseline compromise rate among the devices is 4.5%, but Windows devices are 3.9× more likely to be compromised than the Mac OS devices. The Chrome OS population is small, and only one such device was compromised.

Of course, with modulo dual-booting or using virtual machines, this kind of advice is only actionable to users when choosing a device to use, and is no help once a user is already using a system.

5.2. Update software

Among the hundreds of security experts surveyed, by far the most popular advice is to “Keep systems and software up to date”.²⁰ In this part, we explore the operating system, browser, and Flash update characteristics of the devices in our population, and how they correlate with device compromise.

5.2.1. Operating system

Mac OS. We start by analyzing the update behavior of devices running Mac OS. We see that 7268 (47.5%) devices are identified as Mac and are never absent from the network for more than three days. Of these, we see at least one update for 2113 of them (29.1% of all Mac OS devices). Figure 3 shows the update pattern of these Mac OS devices over time, anchored around the three OS updates released by Apple during our measurement period. In general, Mac OS users are relatively slow to update, anecdotally because of the interruptions and risks Mac OS updates entail.

Of these devices, 57 (2.7%) of them were compromised. Compromised devices have a mean and median update rate of 16.2 and 14.0 days, respectively, whereas their clean counterparts have a mean and median update rate of 18.0 and 16.0 days. However, this difference is not statistically significant according to the Mann-Whitney U test ($p = 0.13$).

Windows. For Windows, we developed a signature to extract the knowledge base (KB) number of “Other Software” updates (for example, Adobe Flash Player, and so forth). Our signature detects when a device downloads the update, and we identify the update’s release day using a Microsoft’s Update Catalog service.¹⁴

Figure 3. Number of days a Mac OS device takes to update to a specific version. The version number on the x-axis denotes the day the specified version update was published.

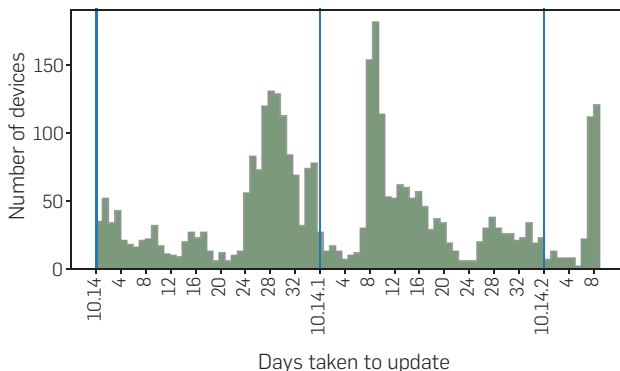
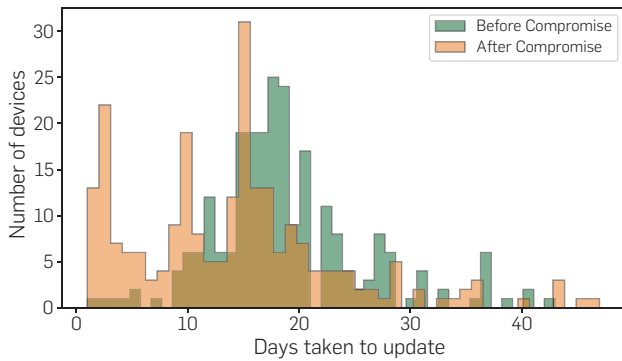


Table 2. Number of days between when an update is published and when devices update. Compromised devices update faster than their clean counterparts across their lifetimes.

Browser	Mean, Median, # (Cmp)	Mean, Median, # (Cln)
Chrome	14.4, 15.0 (421)	15.4, 15.0 (7883)
Firefox	5.64, 3.00 (24)	9.65, 5.00 (424)

Figure 4. Distribution of days a device takes to update Chrome before compromise and after compromise.



Across devices running Windows, we see at least one update for 6459 of them (84% of all Windows devices). Based upon the averages and medians, devices update with similar deltas (2.5 days and 0 days, respectively) regardless of whether they have a security incident. In short, the update behavior of compromised Windows devices is little different than that of clean devices.

5.2.2. Web browser

Browsers are large, complex pieces of software used on a daily basis and, as with most software, these large programs have vulnerabilities. As such, we explore the relationship between compromised and clean devices and browser updating behaviors. Similar to the Mac OS devices, we are able to detect the current browser version number from the User-Agent string of a device. Moreover, we only analyze the dominant browser for each device. While users may use different browsers for different use cases, we identify a dominant browser to remove the noise from user applications that spoof a browser in their User-Agent string.

As browser vendors publish the dates when they make updates available, we can check whether the browser on a device is out of date each time we see the device on the network. Across the measurement period, we then calculate how quickly devices update. We analyzed updates for devices that dominantly use Chrome, Edge, Firefox, and Safari. For devices that are on the network continuously (absent for less than three consecutive days), Table 2 shows the browsers with statistically significant differences in update time between clean and compromised devices (Mann-Whitney U: Chrome $p = 4.2 \times 10^{-4}$ and Firefox $p = 0.03$).

Surprisingly, clean devices appear to spend more time out of date than their compromised counterparts. Examining in more detail, we compare the update behavior of compromised devices before and after their compromise date. We focus on devices using Chrome that have two updates spanning the compromise event (other browsers do not have a sufficiently large sample size). Figure 4 shows the distribution of times devices were out of date with respect to when a browser update was released for updates before and after the device was compromised. The shift in distributions illustrates that devices update faster after compromise: devices

Table 3. Flash Player updates on Windows devices.

Incident?	# Devices	λ	Median	P90	P95	P99	ρ^2
No	1,702	4.2	1	16	20	30	53
Yes	149	3.7	1	16	21	26	47

that use Chrome have a before-compromise mean update rate of 18.9 days (18.0 days median) and an after-compromise mean update rate of 14.2 days (15.0 days median). This difference is significant with $p = 4.8 \times 10^{-12}$ using the Wilcoxon signed-rank test.

5.2.3. Flash Player

The Adobe Flash Player has long been associated with security risk and device compromise. The typical recommendation is to not use Flash at all, but if you do, to keep it up to date. We created a signature to detect Adobe Flash Player on Windows devices. We focused on the desktop version of Flash as major browser vendors issue Flash plugin updates directly. Adobe released six updates within our measurement period, and we used Adobe’s Website to identify the version and release date for each.

Somewhat surprisingly, the desktop Flash is still quite prevalent on devices. A total of 2167 devices (28% of Windows devices) are checked for a Flash Player update, of which 1851 are seen downloading an update. Table 3 shows the average, median, P90, P95, P99, and variance of the number of days between when an update is downloaded and when it is released. The rate of compromise of devices that update Flash is 8.1%, which is only slightly higher than the rate of all Windows devices (7.9%) (Chi-Square $p = 0.057$). Among the 316 devices that we detect Flash Player on, but do not see updates, only 15 are compromised (4.8%). We interpret these results as a community success story. A combination of widespread awareness, aggressive updates, and focused attention has mitigated the desktop Flash as a significant risk factor.

Curiously, compromised devices updated Flash slightly faster than clean devices (Mann-Whitney U test $p = 0.025$). We hypothesized that a compromised device’s update behavior will change after being compromised, so we compared the update patterns for compromised devices before and after becoming compromised. Out of the 149 compromised devices that update Flash, there are 60 devices (40.3%) with updates before and after their first incident. The median and average days compromised devices take to update before an incident are 6.5 and 9.9, respectively, and 0 and 1 days after becoming compromised (Wilcoxon signed-rank test $p = 1.73 \times 10^{-7}$). As with Chrome browser update behavior, these results suggest that shortly after a security incident, devices exhibit better Flash update hygiene.

5.3. Visit reputable websites

Experts recommend users to be careful in the websites that they visit (“visit reputable websites”²⁰), and indeed prior work has found that the category of websites users visit can be indicative of exposure to risky sites.^{4, 22} We perform a similar analysis for devices that are actually

compromised, and for the most part confirm the types of sites that lead to exposure to risky sites also correlate with actual compromise.

To categorize the content devices access, we use the IAB domain taxonomy (Section 3.3.2). We use the Kolmogorov-Smirnov (KS) test with Bonferroni correction to compare the ECDFs of the fraction of distinct registered domains in each category that clean and compromised devices access, and confirm that they are statistically significant (i.e., $p < 0.001$).

Table 4 shows the most substantial differences between the types of content accessed, for example, with clean devices accessing more business, advertising, and marketing content, compromised devices accessed more gaming, hobby, uncategorized, and illegal. We note that although the previous work found that exposed devices visit more advertising domains,²² our finding of the opposite behavior can be explained by differences in methodology. The previous finding used solely HTTP requests generated by static content, whereas our network traces included all HTTP requests (including those generated by JavaScript) as well as HTTPS traffic.

5.4 Network use

One trend is simply that compromised devices generate more Web traffic than clean devices. Figure 5 shows the distributions of average weekly device Web activity for clean and compromised devices. For every device, we count the number of fully qualified domains the device visits via HTTP and HTTPS combined per week, and normalize by averaging across all weeks that the device was active. Each bar in the histogram counts the number

Table 4. Types of content accessed more by clean or compromised devices. We show the median fraction of registered domains accessed in the category for clean (Cln.) and compromised (Cmp.) devices, and delta in median.

Clean devices access more			
Feature	Cln. Median	Cmp. Median	Delta
Business	22.36	20.14	2.22
Advertising	22.65	20.88	1.77
Marketing	12.96	11.66	1.3
Education	3.98	3.53	0.45
Content Server	6.96	6.58	0.38
Television and Video	2.18	1.89	0.29
Arts and Entertainment	2.54	2.27	0.27
Business Software	2.69	2.49	0.2
Web Design/HTML	1.39	1.24	0.15
Compromised devices access more			
Feature	Cln. Median	Cmp. Median	Delta
Computer Games	1.3	2.84	-1.54
Hobbies and Interests	2.61	3.78	-1.17
Uncategorized	26.25	26.97	-0.72
Technology	17.65	18.08	-0.43
Under Construction	5.33	5.65	-0.32
Network Security	1.43	1.65	-0.22
File Sharing	2.28	2.51	-0.23
News/Weather	2.44	2.64	-0.2
Illegal Content	0.15	0.33	-0.18

of devices that visit a given number of FQDNs per week, with 100-domain bins. The distribution for compromised devices is clearly shifted toward visiting more sites per week (and other traffic granularities show similar behavior). We interpret this result as just reflecting that more activity correlates to greater exposure and risk (much like automobile accidents).

5.5. Use antivirus

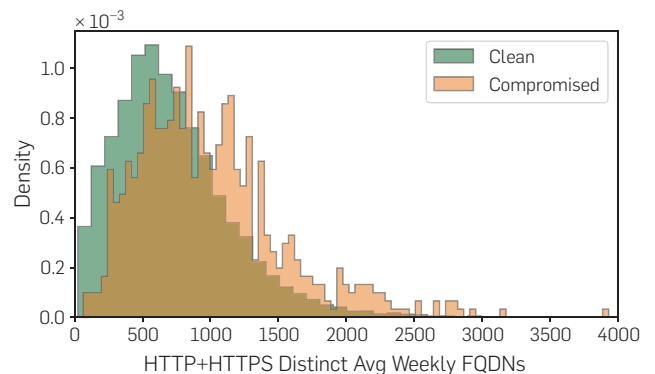
Using antivirus software is a nearly universal recommendation and, indeed, residential students on our campus are nominally required to have antivirus software installed on their devices to use the network. We crafted signatures to detect network activity for over a dozen antivirus products. Focusing on Windows devices, a larger percentage (7%) of devices with antivirus are compromised compared to devices that do not have it (4%). By definition, though, most compromised devices in our population are those that were compromised by malware that antivirus did not catch.

5.6. Software use

As discussed in Section 3.3, we extract a wide variety of features about the software used on devices observed on the network. We now explore how these software features correlate with a device being compromised. Since compromise depends on the operating system being used (Windows devices are compromised more often than Mac OS devices), we also explore software features not only in the context of all devices but also individual operating systems.

For each correlated software feature, Table 5 shows the device population, fraction of compromised devices with the feature, and fraction of compromised devices without the feature. These results provide direct comparisons on the compromise rates between the devices with a particular software feature and without: for example, devices using Tor are compromised 2 to 3.5× more often than devices that do not. To ensure that the comparisons are statistically significant, we use the Chi-Square test with Bonferroni correction because these are binary categorical features, and the very low p-values shown in Table 5 confirm their significance.

Figure 5. Distributions of average weekly device Web activity for clean and compromised devices.



Devices using some specific applications correlate very strongly with compromise, independent of operating system and network activity. Devices using Adobe AIR, P2P file sharing networks, Thunderbird, and Tor on average are much more likely to be compromised than devices that do not use such applications. Using these applications does indeed put devices at significantly more risk. The Thunderbird email client is particularly ironic; the one reason why people use Thunderbird is because of its PGP integration;⁷ yet, Thunderbird is rife with reported vulnerabilities (420 code execution vulnerabilities reported in CVE Details⁶).

6. RANKING FEATURE IMPORTANCE

Our analyses so far have focused on individual security practices. As a final step, we explore the relative importance of all the features we extract using statistical modeling, as well as the relative importance of features exhibited during the hour before a device is compromised. Our goal is not to train a general security incident classifier, but instead to generate a model for ranking the relative importance of our features.

6.1. Experimental setup

Logistic regression is a statistical technique for predicting a binary response variable using explanatory variables.¹¹ We set the response variable to be whether or not a device is compromised, and use all of the device features we extract from the network as the explanatory variables. We first split the data into training (50%) and test (50%), and normalize the explanatory variables to have zero mean and unit variance.

To find the important explanatory variables, we use L1 logistic regression because we have a high number of

Table 5. Software features across device populations correlated with compromise. Each feature shows the number of devices with the feature, p-value from the Chi-Square test, fraction of compromised devices with and without the feature. Compromise rates: All devices 4.5%, Windows devices 7.0%, and Mac OS devices 1.9%.

Group	Feature	# Dev	P-value	w/ Feat.	w/o Feat.
All	Adobe AIR	826	< 0.001	10%	4%
All	P2P	2,237	< 0.001	13%	3%
All	Thunderbird	69	< 0.001	33%	4%
All	Uses Tor	321	< 0.001	12%	4%
All	Password Mgr	434	< 0.001	8%	4%
All	Remote DNS	8,631	< 0.001	6%	2%
Win	Adobe AIR	490	< 0.001	13%	7%
Win	P2P	1,676	< 0.001	15%	5%
Win	Thunderbird	28	< 0.001	43%	7%
Win	Uses Tor	188	< 0.001	15%	7%
Win	Password Mgr	262	0.001	12%	7%
Win	Remote DNS	5,249	< 0.001	8%	5%
Mac	Adobe AIR	336	< 0.001	6%	2%
Mac	P2P	541	< 0.001	7%	2%
Mac	Thunderbird	29	< 0.001	34%	2%
Mac	Uses Tor	123	< 0.001	7%	2%
Mac	Password Mgr	159	0.755	1%	2%
Mac	Remote DNS	3,212	< 0.001	3%	1%

explanatory variables. To find the optimal regularization parameter, we implement hyperparameter tuning: we build 200 models, each with a different regularization parameter, and identify the model that performs best. To identify the best model while avoiding selection bias, for each model, we perform a 10-fold cross validation.

To compare the importance of each feature, we implement a greedy deletion algorithm.⁹ We start with the N important features used to predict security incidents identified by the best model (previous paragraph). For $N - 1$ feature combinations, we train regularized models with hyperparameter tuning. From the resulting models, we identify the model that has the maximum area under curve (AUC) (when predicting on validation data), and exclude the unused feature in the next iteration of the algorithm as it contributes least to the overall AUC compared to the other feature combinations. We repeat this process until we have a model that uses a single feature ($N = 1$); the remaining feature contributes the most to the AUC by itself and in the presence of other features. Finally, we interpret the results in terms of the changes to the test AUC when features are added to the final model.

6.2. All features

We run the greedy deletion algorithm multiple times with different device groupings: all devices, Windows devices, Mac OS devices, and devices with on-median more HTTP traffic. We consider devices that produce on-median more HTTP traffic based on our observations in Section 5.4. Table 6 shows the top four features for each grouping, the feature's AUC contribution when predicting on validation and test data, and the ratio of the feature's median (continuous) or mean (categorical) value for compromised and clean devices. Because we select the feature combination with the highest validation AUC, it is possible that adding in an extra feature will result in a small negative contribution

Table 6. AUC gains from the top four features used to detect devices with security incidents, as well as the ratio of median (continuous) or mean (categorical) values. Ratios > 1 (green) indicate compromised devices exhibit more of the feature.

Group	Feature	Val AUC	Test AUC	Ratio
All	IAB Computer Games	+68.3%	+69.7%	2.2×
All	HTTP Reg Domains	+7.0%	+5.2%	1.6×
All	HTTP in TLD.cn	+2.3%	+3.7%	3.5×
All	Windows Antivirus	+1.9%	+1.1%	1.7×
Win	HTTP FQ Domains	+71.9%	+71.1%	1.6×
Win	IAB Computer Games	+4.2%	+2.9%	1.7×
Win	UA Str Safari	+2.2%	+2.5%	3×
Win	UA Str IE	+1.4%	+1.3%	1.1×
Mac	HTTP in TLD.cn	+76%	+76%	∞
Mac	UA Str IE	+5.3%	+4.3%	6.2×
Mac	HTTP Traffic at 2AM	+3.8%	-1.3%	0.9×
Mac	HTTP in TLD co.kr	+1.5%	+3.7%	1×
HTTP	IAB Shareware	+66.3%	+60%	∞
HTTP	UA Str IE	+7.2%	+7.9%	1.9×
HTTP	UA Str Android	+3.4%	+1.3%	2.2×
HTTP	Uses P2P	+1.0%	+2.7%	1.3×

Table 7. AUC gains for the top eight features for detecting devices with security incidents one hour before compromise.

Feature	Val AUC	Test AUC
IAB Computer Games	+71.9%	+74.2%
IAB Web Search	+4.0%	+3.6%
IAB Illegal Content	+2.2%	+3.6%
IAB JavaScript	+1.0%	+0.1%
IAB Computer Networking	+0.7%	+0.1%
IAB Adult Content	+0.7%	+0.7%
IAB Shareware/Freeware	+0.7%	+0.4%
IAB Internet Technology	+0.5%	+1.5%

to the test AUC (e.g., the “HTTP Traffic at 2AM” feature for Mac OS devices).

Our results indicate that behavioral features, regardless of device grouping, are most correlated with device compromise. In all cases, the first feature in each grouping relates to how much a device accesses a Web content or the type of content being accessed. Having Windows antivirus products (a proxy for using Windows, which has a significantly higher compromise rate), or using P2P applications is the only two software features in the top four of any grouping. Having the IE User Agent feature highly ranked highlights the challenge of cursory feature extraction. Applications can make use of embedded browsers, and examining traffic with an IE User Agent string shows many of the detections are actually from the QQ chat application and Qihoo 360 security product, not the IE browser. We also find that compromised devices, in the majority of cases (except for two features within the Mac OS grouping), exhibit more of each feature compared to clean devices.

6.3. One hour before compromise

Lastly, we use our statistical model to examine the relative importance of security features focusing on the hour leading up to device compromise: compared to devices that are not compromised, how are compromised devices behaving differently leading up to becoming compromised? For each compromised device, we extract their features from the hour before their first incident. To compare differences in behavior, we construct a synthetic control by taking a pseudorandom sample of clean devices. Specifically, for each compromised device, we randomly select up to 300 clean devices that are (1) active in the same hour window and (2) visit at least 50 distinct registered domains.

Table 7 shows the most important features (relative to one another) for identifying the compromised devices an hour before they are compromised. For our devices, the type of Websites visited (Section 5.3) are the most distinguishing features. On average, compromised devices visit more Websites in each of the eight categories in Table 7 than the clean devices. The most popular domains our devices visit in these categories do correspond well to the category domains. For some of the very generic labels, “Computer Games” are gaming sites; “Computer Networking” include ISPs and IP geolocation services; “Internet Technology” include SSL certificate sites and registrars, etc.

7. CONCLUSION

The practice of cybersecurity implicitly relies on the assumptions that users act “securely” and that our security advice to them is well-founded. In this paper, we have sought to ground both assumptions empirically: measuring both the prevalence of key security “best practices” as well as the extent to which these behaviors (and others) relate to eventual security outcomes. We believe that such analysis is critical to making the practice of security a rigorous discipline and not simply an art.

However, achieving the goal of evidence-based security is every bit as formidable as delivering evidence-based healthcare has proven to be. In any complex system, the relationship between behaviors and outcomes can be subtle and ambiguous. For example, our results show that devices using the Tor anonymizing service are significantly more likely to be compromised. This is a factual result in our data. However, there are a number of potential explanations for *why* this relationship appears: Tor users could be more risk-seeking and expose themselves to attack, or they might be more targeted, or there might be vulnerabilities in Tor itself. Indeed, it is even possible that Tor use simply happens to correlate with the use of some other software package that is the true causal agent.

Thus, although some of our results seem likely to not only have explanatory power but also to generalize (e.g., the use of Thunderbird and Adobe AIR, both historically rife with vulnerabilities, has significant correlations with host compromise), others demand more study and in a broader range of populations (e.g., why are gamers more prone to compromise?). Those results that lack simple explanations are a reflection of the complexity of the task at hand. Having started down this path of inquiry, though, we are optimistic about answering these questions because we have shown that the methodological tools for investigating such phenomena are readily available. We look forward to a broader range of such research going forward as our community helps advance security decision-making from the “gut instinct” practice it is today, to one informed and improved by the collection of concrete evidence.

Acknowledgments

This work was supported in part by NSF grants CNS-1629973 and CNS-1705050, DHS grant AFRL-FA8750-18-2-0087, and the Irwin Mark and Joan Klein Jacobs Chair in Information and Computer Science. □

References

1. Apple. Update your iPhone, iPad, or iPod touch, 2018. <https://support.apple.com/en-us/HT204204>.
2. Bellare, M., Rogaway, P. The FFX mode of operation for format-preserving encryption. Manuscript (standards proposal) submitted to NIST (2010).
3. Bilge, L., Han, Y., Dell’Amico, M. RiskTeller: Predicting the risk of cyber incidents. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)* (Dallas, Texas, USA, November 2017).
4. Canali, D., Bilge, L., Balzarotti, D. On the effectiveness of risk prediction based on users browsing behavior. In *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security (CCS)* (Kyoto, Japan, June 2014).
5. Marshall, C., Ellis, C. The best free password manager 2019, 2018. <https://www.techradar.com/news/software/applications/the-best-password-manager-1325845>.
6. CVE Details. Mozilla Thunderbird Vulnerability Statistics, 2019. <https://www.cvedetails.com/product/3678/?q=Thunderbird>.
7. The Enigma Project. Enigma!—OpenPGP encryption for Thunderbird, 2019. <https://www.enigma.net/>

index.php/en/home.

8. Forget, A., Pearman, S., Thomas, J., Acquisti, A., Christin, N., Cranor, L.F., Egelman, S., Harbach, M., Telang, R. Do or do not, there is no try: User engagement may not improve security outcomes. In *Proceedings of the 12th Symposium on Usable Privacy and Security (SOUPS)* (Denver, CO, USA, June 2016).
9. Hastie, T., Tibshirani, R., Friedman, J. *The Elements of Statistical Learning*. Springer New York Inc., 2001.
10. Herley, C. So long, and no thanks for the externalities: The rational rejection of security advice by users. In *Proceedings of the 2009 Workshop on New Security Paradigms Workshop* (Oxford, United Kingdom, September 2009).
11. Hosmer Jr, D.W., Lemeshow, S. *Applied Logistic Regression*. 2nd edn. John Wiley & Sons, New Jersey, USA, 2000.
12. IAB. IAB Tech Lab Content Taxonomy. 2019. <https://www.iab.com/guidelines/iab-tech-lab-content-taxonomy/>.
13. Khan, M., Bi, Z., Copeland, J.A. Software updates as a security metric: Passive identification of update trends and effect on machine infection. In *Proceedings of IEEE Military Communications Conference (MILCOM)* (Orlando, Florida, USA, October 2012).
14. Microsoft. Microsoft update catalog. 2019. <https://www.catalog.update.microsoft.com/Home.aspx>.
15. Mozilla Foundation. Public suffix list website. 2019. <https://publicsuffix.org/>.
16. Rubenking, N.J. The best antivirus protection for 2019, 2019. <https://www.pcmag.com/article2/0,2817,2372364,00.asp>.
17. ProofPoint. ET Pro Ruleset, 2019. <https://www.proofpoint.com/us/threat-insight/et-pro-ruleset>.
18. Redmiles, E.M., Kross, S., Mazurek, M.L. Where is the digital divide?: A survey of security, privacy, and socioeconomic. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA, May 2017).
19. Redmiles, E.M., Kross, S., Mazurek, M.L. How well do my results generalize? Comparing security and privacy survey results from MTurk, web, and telephone samples. In *Proceedings of the 2019 IEEE Symposium on Security and Privacy* (San Francisco, CA, USA, May 2019).
20. Reeder, R., Ion, I., Consolvo, S. 152 Simple steps to stay safe online: Security advice for non-tech-savvy users. *IEEE Security and Privacy* 15, 5 (June 2017):55–64.
21. Sawaya, Y., Sharif, M., Christin, N., Kubota, A., Nakarai, A., Yamada, A. Self-confidence trumps knowledge: A cross-cultural study of security behavior. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA, May 2017).
22. Sharif, M., Urakawa, J., Christin, N., Kubota, A., Yamada, A. Predicting impending exposure to malicious content from user behavior. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS)* (Toronto, Canada, October 2018).
23. Vitale, F., McGrenere, J., Tabard, A., Beaudouin-Lafon, M., Mackay, W.E. High costs and small benefits: A field study of how users experience operating system upgrades. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA, May 2017).
24. Webshrinker. IAB categories. 2018. <https://docs.webshrinker.com/v3/iab-website-categories.html#iab-categories>.
25. Webshrinker. Webshrinker website. 2019. <https://www.webshrinker.com/>.

Louis F. DeKoven (ldekoven@cs.ucsd.edu), University of California, San Diego, CA, USA.

Audrey Randall (aurandal@eng.ucsd.edu), University of California, San Diego, CA, USA.

Ariana Mirian (amirian@cs.ucsd.edu), University of California, San Diego, CA, USA.

Gautam Akiwate (gakiwate@cs.ucsd.edu), University of California, San Diego, CA, USA.


Ansel Blume (ablume@ucsd.edu), University of California, San Diego, CA, USA.

Lawrence K. Saul (saul@cs.ucsd.edu), University of California, San Diego, CA, USA.

Aaron Schulman (schulman@cs.ucsd.edu), University of California, San Diego, CA, USA.

Geoffrey M. Voelker (voelker@cs.ucsd.edu), University of California, San Diego, CA, USA.

Stefan Savage (savage@cs.ucsd.edu), University of California, San Diego, CA, USA.

 This work is licensed under a <https://creativecommons.org/licenses/by/4.0/>

ACM Student Research Competition

Attention: Undergraduate and Graduate Computing Students



Association for Computing Machinery
Advancing Computing as a Science & Profession

The ACM Student Research Competition (SRC) offers a unique forum for undergraduate and graduate students to present their original research before a panel of judges and attendees at well-known ACM-sponsored and co-sponsored conferences. The SRC is an internationally recognized venue enabling undergraduate and graduate students to earn many tangible and intangible rewards from participating:

- **Awards:** cash prizes, medals, and ACM student memberships
- **Prestige:** Grand Finalists receive a monetary award and a Grand Finalist certificate that can be framed and displayed
- **Visibility:** opportunities to meet with researchers in their field of interest and make important connections
- **Experience:** opportunities to sharpen communication, visual, organizational, and presentation skills in preparation for the SRC experience

Learn more about ACM Student Research Competitions: <https://src.acm.org>

[CONTINUED FROM P. 104] plaining to individuals when the system itself is uncertain in a way that just makes people better reflect on their trust in the decisions being thrown at them by these agents.

In other words, more of a contextual prompt, rather than a broad disclaimer.

Right. We've started to examine this approach for use primarily in the self-driving domain. We have all this data about dangerous intersections—two highways that merge, for example, where the accident rate is higher. You can take that information and decode it. Let's say you're in your car, on autonomous mode. It is 3 P.M., and you're about to enter an intersection that is hazardous. Now, 3 P.M. is when the kids are out. So maybe the car says, "Hey, school kids are on the road and a child died last week."

What we're finding is that people make better decisions when risk is quantified in terms that are more personal. They either pay closer attention to the road, for example, or they'll do a full override of the autonomous system to get through that intersection.

It sounds like a more context-aware version of those radar speed signs, which prompt drivers to slow down by showing them how fast they're driving.

Right. It's a trigger, but it also gives people autonomy to make the decision themselves, and that's key.

You're still on the board of directors of Zyrobotics, an organization you co-founded in 2013 to create educational technologies for children. How has that work evolved during the pandemic, when many schools were closed?

During the pandemic, Zyrobotics had to pivot to focus more on the software side and on maintaining the therapy and STEM (science, technology, engineering, and math) education apps they'd already created, because the hardware supply chains froze up. Now that things are starting to open up, they're starting to interact with school districts again, which also slowed down when everyone went remote.

Zyrobotics works really hard to make technologies that are accessible to different learners. How do you make them financially accessible?

"What we're finding is that people make better decisions when risk is quantified in terms that are more personal."

One of the projects I'm still involved with is in the area of accessible coding, looking at the intersectionality of disability and socioeconomic status. If you're a parent from a middle-class neighborhood and your child has a disability, you pull your resources together and provide the scaffolds needed for your child. It's different in low SES (socioeconomic status) communities and unfortunately, the lack of resources also has an intersection with ethnicity and race.

This project did two things. First, we developed an open source robotics platform that's based on Arduino and rapid prototyping machines. This is modeled after a philosophy similar to the Helping Hand Project (www.helpinghandproject.org), which creates open source software and designs that college engineering students and even high school students can use to build hands for children and adults who have lost them. It can be very low-cost; it might cost maybe a hundred dollars to make a hand.

So you provide all of the software, plans, and designs to enable things to be built?

Yes, but we are also developing a software equivalent. If you can't get a local college to build the robotic hardware for a K-12 school, then you can download the software equivalent of a virtual world where you're learning the coding, and it's accessible for children with visual or hearing impairments or with motor disabilities.

You've made the point that designing educational tools for kids with special needs is actually a good way of designing educational tools for all kids. Can you elaborate on that?

A lot of times, even at the college level, instructors teach things based on the way that they learn. So, if I like to write and I learned by reading a lot, then I'm going to give my students a lot of reading assignments. But students have very different ways of consuming and processing information. People know that about children with special needs, but I don't think they realize that children have these nuances across the board. So when you design for what I call the extremes, you also incorporate the different learning styles of children who don't necessarily fit into the box that the teacher is teaching from.

How does that philosophy work when you're designing outside of an educational context?

I encourage people to think about who their opposite is in terms of attributes, and design for that person. If I'm a technologist living in a high-SES neighborhood, then I need to think about designing solutions for someone who is in, say, rural America. Then what happens is, even though that's not my lived experience, it makes me sit back and start rethinking my design choices. It doesn't get you to the other extreme, but it does break the habit of designing based on what you know, and it makes you explore other things you might otherwise not have.

It's difficult to argue against the idea of designing for a diverse audience and incorporating different perspectives. But it's also difficult to put in practice.

The practice part is still difficult, but I'm seeing much more concern about it at the upper management levels in industry and academia. That means it's propagating downward, whereas before it was coming from the grassroots. When it comes from the top, people tend to say, "Let's figure this out, and here are the incentives to institute change." So, I'm starting to see movement. A lot of people in the community are frustrated that the movement isn't fast enough, but I've been in this field for a long time, and if I look at the Delta of movement now versus the Delta of movement 20 years ago, it's exponential.

Leah Hoffmann is a technology writer based in Piermont, NY, USA.

© 2022 ACM 0001-0782/22/9 \$15.00

Q&A

Advancing the Ability of Robots to Help

ACM Athena Lecturer Ayanna Howard considers the benefits of robotics and the potential drawbacks of overtrust.

AYANNA HOWARD, ROBOTICIST, ACM Athena Lecturer, and dean of The Ohio State University College of Engineering, is optimistic about the ability of robots to help people. She understands the challenges that must be addressed for that to happen, and has worked throughout her career not just to advance the technical state of the art, but to quantify and overcome issues including trust and bias in artificial intelligence (AI). Here, she talks about self-driving cars, accessible coding, and how to incorporate different perspectives into hardware and software design.

The pandemic heightened public interest in robots—suddenly, we all want robot cleaners and robot grocery deliverers and so on. How is that impacting the robotics community?

I see two things. First, the robotics industry is getting robots out to people much quicker than we had anticipated. The pandemic accelerated the use of robots, which lowered costs, and therefore you now see the growth of a real market in community-facing robotics.

The second thing has to do with the robotics research community. There are still a lot of unmet problems, like mobile manipulation, that we really need to solve to meet this new demand, so I anticipate a lot more funding and focus on those problems.

The other area which I am actually more excited about is social interaction. That is also accelerated, in the sense that we can now see that robots *do* have



the ability to interact in a social way and are not necessarily replacing people.

In settings like factories, where robots probably will replace people, you have said you are optimistic about our ability to retrain workers.

I agree with the concept of the human dignity of work, but not all work is dignified. I think that there is a disconnect, because the individuals who say all work is good work are not the ones who have those jobs.

But companies are also starting to think a little more about their social responsibility and saying, “If we are going to be putting these individuals out

“I agree with the concept of the human dignity of work, but not all work is dignified. I think there’s a disconnect, because the individuals who say all work is good work are not the ones who have lost those jobs.”

of work, maybe we should also invest in retraining them for the other kinds of jobs that are going to come about.”

Let’s talk about your research into overtrust. Can you summarize the problem and share some of your recent findings about using explainable AI methods to counteract it?

Prior research from my group and others has shown that when you’re using robots and AI agents, and they are dependable, you start believing that they are always dependable, and you won’t even second-guess yourself after a while. One thing we’re looking at is ex- [CONTINUED ON P. 103]

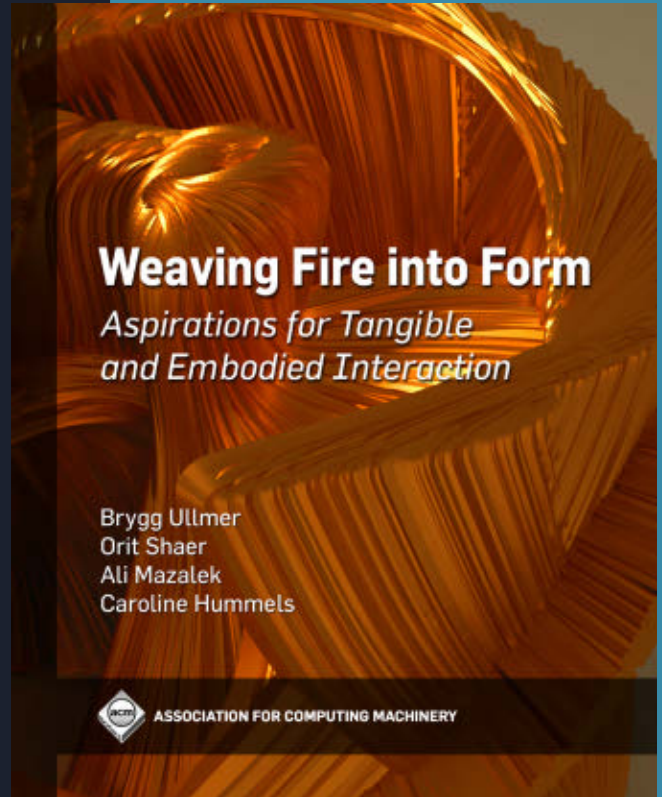


ACM BOOKS

Collection II

This book investigates multiple facets of the emerging discipline of Tangible, Embodied, and Embedded Interaction (TEI). This is a story of atoms and bits. We explore the interweaving of the physical and digital, toward understanding some of their wildly varying hybrid forms and behaviors. Spanning conceptual, philosophical, cognitive, design, and technical aspects of interaction, this book charts both history and aspirations for the future of TEI. We examine and celebrate diverse trailblazing works, and provide wide-ranging conceptual and pragmatic tools toward weaving the animating fires of computation and technology into evocative tangible forms. We also chart a path forward for TEI engagement with broader societal and sustainability challenges that will profoundly (re)shape our children's and grandchildren's futures. We invite you all to join this quest.

- Introduction
- Tangible and Embodied Interaction
- TEI in the Wild
- Framing TEI
- Theories of Embodiment
- Mediating Technologies
- Aesthetics of TEI
- Evaluating TEI
- Paths Forward: Aspirations for TEI
- Summary
- Author Biographies
- Appendices



Weaving Fire into Form Aspirations for Tangible and Embodied Interaction

**Brygg Ullmer
Orit Shaer
Ali Mazalek
Caroline Hummels**

ISBN: 978-1-4503-9767-4
DOI: 10.1145/3544564

<http://books.acm.org>

<http://store.morganclaypool.com/acm>



MATLAB SPEAKS MACHINE LEARNING

With MATLAB® you can use clustering, regression, classification, and deep learning to build predictive models and put them into production.

mathworks.com/machinelearning