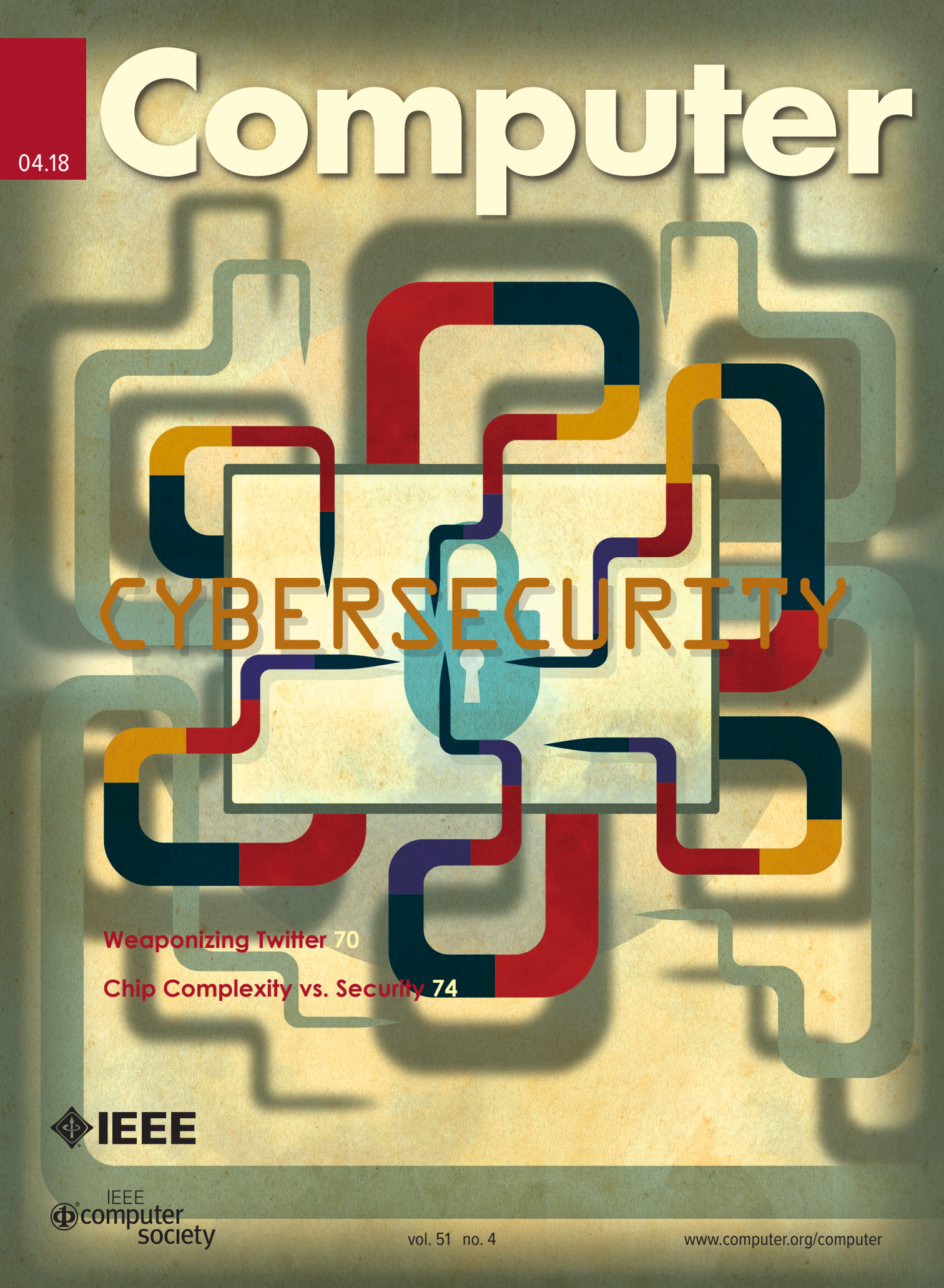
04.18

Computer



IEEE

IEEE computer society

vol. 51 no. 4

www.computer.org/computer

IEEE computer society

SHARE THE GIFT OF KNOWLEDGE:

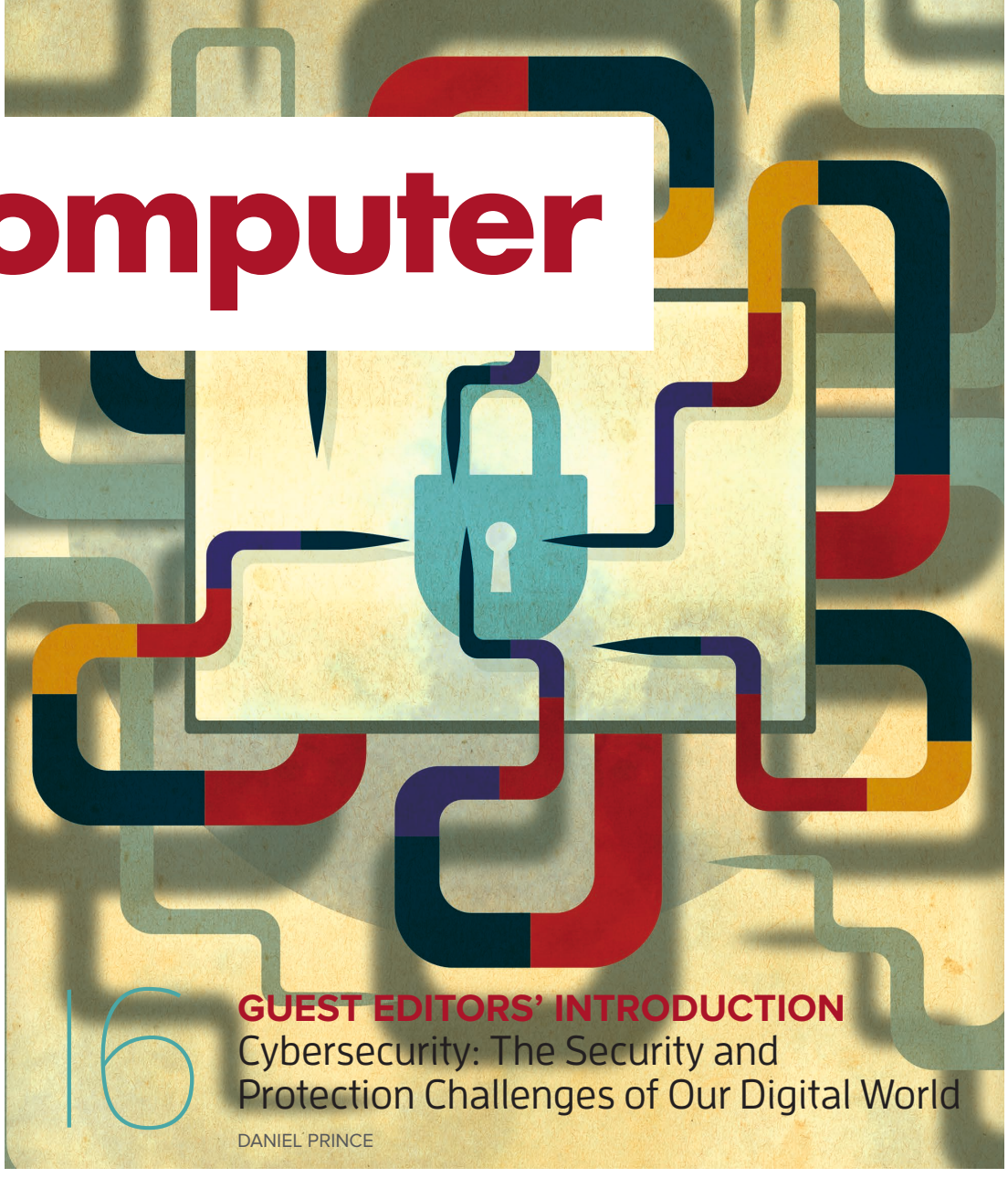
Give Your Favorite Student a Membership to the IEEE Computer Society!

With an **IEEE Computer Society Membership**, your student will be able to build their network, learn new skills, and access the best minds in computer science before they're even out of school. Your gift includes thousands of key resources that will quickly transition them from classroom to conference room, such as:

- **A subscription to *Computer* magazine** (12 issues per year)
- **A subscription to *ComputingEdge*** (12 issues per year)
- **Local chapter membership**
- **Full access to the Computer Society Digital Library**
- **Eligible for 3 student scholarships where we give away US$40,000 yearly**
- **Skillsoft:** Learn new skills anytime with access to 3,000 online courses, 11,000 training videos, and 6,500 technical books.
- **Books24x7:** On-demand access to 15,000 technical and business resources.
- **Unlimited access to computer.org and myCS**
- **Conference discounts**
- **Members-only webinars**
- **Deep member discounts** on programs, products, and services

Give Your Gift at: www.computer.org/2018gift

Computer



APRIL 2018

FEATURES

APRIL 2018

CONTENTS

ABOUT THIS ISSUE
CYBERSECURITY

Cybersecurity is now the pervasive technical, social, and economic issue affecting all of us.

PERSPECTIVES

RESEARCH FEATURE

See **www.computer.org/computer-multimedia** for multimedia content related the features in this issue.

Circulation: *Computer* (ISSN 0018-9162) is published monthly by the IEEE Computer Society. IEEE Headquarters, Three Park Avenue, 17th Floor, New York, NY 10016-5997; IEEE Computer Society Publications Office, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; voice +1 714 821 8380; fax +1 714 821 4010; IEEE Computer Society Headquarters, 2001 L Street NW, Suite 700, Washington, DC 20036. IEEE Computer Society membership includes a subscription to *Computer* magazine.

Postmaster: Send undelivered copies and address changes to *Computer*, IEEE Membership Processing Dept., 445 Hoes Lane, Piscataway, NJ 08855. Periodicals Postage Paid at New York, New York, and at additional mailing offices. Canadian GST #125634188. Canada Post Corporation (Canadian distribution) publications mail agreement number 40013885. Return undeliverable Canadian addresses to PO Box 122, Niagara Falls, ON L2E 6S8 Canada. Printed in USA.

COLUMNS

Departments

Membership News

Computer

IEEE COMPUTER SOCIETY http://computer.org // +1 714 821 8380
COMPUTER http://computer.org/computer // computer@computer.org

LETTERS

ANOTHER TAKE ON RISC

To the Editor:

After reading the article "Reduced Instruction Set Computers Then and Now," by David Patterson (December 2017, pp. 10–12), I wanted to share another perspective. In the 1980s, nobody knew what was going to happen in the computer industry over the ensuing 20 or 30 years. It went almost without saying that RISC ISAs (instruction set architectures) would take over the world, due to their predicted major advantages over the ugly old CISC chips such as the x86. Even within Intel, there were some powerful voices echoing that theme. We can now look back and see that that's not what happened.

Patterson suggests that there have been no new general-purpose CISC ISAs since RISCs first appeared. Even if true, I'm not sure what that proves. But I am sure that the archetypical CISC, the x86, was well established before RISC came along. In fact, it not only withstood but thrived in competition with an array of RISC attackers: PowerPC, Alpha, MIPS, Sparc, PA-RISC, M88K, AM29K. The x86 has been the dominant computer architecture in everything from servers to laptops for more than 30 years. Nobody would argue that this success was a result of some secret advantage of the x86 ISA per se. The very least one should conclude from this is that ISAs are evidently not all that matters.

Patterson asserts that "the hardest part of computer design is control." In a way, that's arguably true, but he essentially goes on to equate control complexity to microcode, in which case the assertion is inarguably false. As part of these architecture efforts, I personally supervised the microcode teams for the Pentium chips of the 1990s; out of teams with 500–900 designers, fewer than 10 of them were in the microcode group. Although microcode isn't trivial, design engineers don't consider it to be harder than many other aspects of modern industrial microprocessor design. If you want to win on the world stage, it's *all* hard no matter which ISA you're implementing.

I agree with Patterson that x86s surpassed RISC performance in the 1990s: specifically, with the P6/Pentium Pro in 1995. But I certainly do not agree that somehow the P6 achieved its supremacy by using "RISC ideas." Nonsense. P6 was based on speculation, out-of-order execution, deep (disjoint) pipelines, good branch prediction, superscalar microarchitecture, register renaming, fast clocks, and the concept of micro-ops. Every one of those ideas was already in use in machines prior to RISCs except for micro-ops, and that one we invented ourselves. Bottom line, we dragged an ugly ISA (x86) up a steep hill and beat the best of the RISC competition. If ISAs are so important, how was that possible?

Next, Patterson switches to unit sales volumes; he points out that the volumes of ARM cores being shipped today dwarf the x86 volumes being sold by Intel and AMD. He implies that this was somehow an inevitability, that there's a "renewed need in the post-PC era for simpler ISAs." But that's a leap to a conclusion for which he has provided no evidence. I know from first-hand experience that from 1996 through at least 2003, Intel simply refused to try to compete at the low-power end of the CPU space. As their chief architect for most of that period, I tried hard to get them interested in designing a very-low-power x86 for those emerging markets. Upper management refused, mainly on the grounds that the profit margins in those markets were an order of magnitude less than they already enjoyed. In effect, they refused to compete. So we cannot know whether x86 would have been able to compete squarely in these high-volume markets; by the time new management tried, it was too little, too late.

I have a RISC-V, it looks like a good architecture to me, and per Patterson's description, it has benefited from 30+ years of collective experience. For me, the upshot of the RISC/CISC debates of the 1980s wasn't to try to establish which was "best" (whatever that might mean). For me, the RISC research represented a break from the quasi-religious approaches that preceded it, such as "narrowing the semantic gap" between ISAs and assembly programmers, or quixotic quests for architectural purity such as Intel's 432. The RISC researchers quantified the performance of competing ideas, and by example forced the entire field onto a numerical basis for making progress. And that lesson, we *did* take to heart in designing Intel's chips.

Bob Colwell
bob.colwell@gmail.com

Author's Response:

I felt a wave of nostalgia when I read Bob Colwell's letter questioning my assessment of RISC in my short retrospective. His letter reminded me of stories of veterans of the US Civil War in the same retirement home arguing how they could have won critical battles with a few changes in tactics or luck.

Colwell and I are veterans of the RISC vs. CISC conflict of the 1980s. He signed up while a grad student to the CISC side,[1,2] and then did his best in his dissertation to find virtues in Intel's infamous 432 processor,

0018-9162/18/$33.00 © 2018 IEEE

certainly the most complicated microprocessor of the 1980s.[3] And he's the father of the P6, the most successful x86 microarchitecture.

Colwell argues that it's "nonsense" that the P6 is based on RISC ideas. RISC machines were the first microprocessors to:

- Have split instruction and data caches;
- Use the efficient five-stage pipeline, enabled by the split caches;
- Have a superscalar microarchitecture, which allowed fetching and executing multiple instructions per clock cycle;
- Use "superpipelining," a much deeper pipeline than five stages, enabling higher clock rates; and
- Offer multilevel caches on chip, which help overcome the performance bottleneck of long latency to memory especially as processor clock rates increased.

Although some of these ideas first appeared in larger computers, RISC engineers were the first to make them work within the constraints of a single chip. Using the industry standard SPECint benchmark as the measure, these innovations made RISC microprocessors the fastest in the 1980s and 1990s.[4]

Intel and AMD designers rose to the challenge. The first step was to translate the variable length and variable time x86 instructions into simpler fixed-length instructions (similar to what DEC VAX engineers did in the 1980s). Intel calls them *micro-operations*, and AMD calls them *RISC operations*. They then used all the ideas above. The RISC and x86 designers added out-of-order execution at about the same time, which comes from a high-end IBM S/360 mainframe of the 1960s. Intel and AMD had larger teams and access to better semiconductor processing than the RISC companies, and, with excellent engineering, x86 microprocessors became the fastest starting in the 2000s. As I said in my retrospective, by the end of the PC era, x86 had clearly won.

Like the Civil War veterans reliving battles, Colwell thinks Intel could have won the PostPC era as well with a small change in strategy. In his view, the flaw isn't the inherent complexity overhead of the x86, it was simply waiting too long to try. UC Berkeley's parallel computing research was cosponsored by Intel from 2007 to 2013, and we learned that the top priority of the Intel CEO Paul Otellini during those years was getting x86 into the mobile market. Intel finally gave up on that goal in 2017. Many blame the overhead in design, area, power, and verification of the complex x86 instruction set.

ARM ("Advanced RISC Machine") has virtually the same monopoly on mobile devices now that x86 had on PCs; it recently celebrated shipment of its 100 billionth chip. Surely fewer than 5 billion chips use the older x86 instruction set.

Indeed, the biggest competitor in the future is likely not x86, but the free and open RISC-V (www.riscv.org). It's a simple, elegant instruction set architecture. (Even Colwell likes it.) It competes as an open alternative to the proprietary instruction sets, much like open Linux competes with Microsoft Windows. Time will tell if RISC-V will have as much success in processors as Linux has had in operating systems, but that is the long-term goal.

Sadly, Otellini recently passed away. One obituary noted as his biggest regret that he sold off an ARM processor design and team that Intel had acquired from another company.[5] Had they kept the product, Steve Jobs might have used it in the first iPhone in 2007, since it was clearly the best ARM processor of that era. (In fact, Apple eventually hired the engineers who designed those processors.) Apple still uses ARM in iPhones and iPads, and is even rumored to be switching to ARM for its laptops in 2020. Imagine how much rosier Intel's future would be if it was the main supplier of chips for both the PC era *and* the Post-PC era!

I'd like to thank Colwell again for the trip down memory lane that let me recall the RISC vs. CISC battle of our youth. We both agree that the winners have varied over time. I believe the field benefits from the frank and open exchanges like the ones we've participated in now and in the past.

Architecture debates have traditionally been settled ultimately by companies spending billions of dollars developing products on both sides of an issue, and letting the marketplace determine the winner. Of the 20 billion 32-bit and 64-bit processors to be shipped this year, 99 percent will be RISC. I'll leave it to the readers to decide whether RISC or CISC dominates today, and whether the reason is inherently technical or simply better business decisions.

David Patterson
pattrsn@cs.berkeley.edu

REFERENCES

1. R.P. Colwell et al., "Peering through the RISC/CISC Fog: An Outline of Research." *ACM SIGARCH Computer Architecture News*, vol. 11, no. 1, pp. 44–50.
2. R.P. Colwell et al., "Instruction Sets and Beyond: Computers, Complexity,

and Controversy," *Computer*, vol. 18, no. 9, 1985, pp. 9–18.
3. R.P. Colwell, "The Performance Effects of Functional Migration and Architectural Complexity in Object-Oriented Systems," PhD dissertation, Carnegie Mellon University, 1985.
4. J. Hennessy and D. Patterson, *Computer Architecture: A Quantitative Approach*, 6th Edition, 2018, Elsevier. [see Figure 1.1]
5. D. Clark, "Paul S. Otellini, Who Led Intel and Saw It Grow Even More, Dies at 66," *The New York Times*, 3 October 2017.

IN DEFENSE OF PROGRAMMING LANGUAGE RESEARCH

To the Editor:

I read the opinion piece "On Methodological Irregularities in Programming Language Research," by Andreas Stefik and Stefan Hanenberg in the August 2017 issue (pp. 60–63). The authors question programming language (PL) design conferences such as PLDI, OOPSLA, ICFP, and ECOOP for a lack of "rigorous evidence standards like those in other sciences"—standards such as randomized controlled trials for evaluating the effectiveness of an intervention. The authors conclude by calling for the imposition of strict reporting standards in software engineering conferences and journals, like the Consolidated Standards of Reporting Trials (CONSORT) statement (www.consort-statement.org) in medicine and the What Works Clearinghouse (WWC) guidelines (http://ies.ed.gov/ncee/wwc/Docs/referenceresources/wwc_procedures_v3_0_standards_handbook.pdf) in education.

Stefik and Hanenberg's recommendations are reasonable for a certain class of software engineering research—specifically, for research aiming to show that a given intervention, such as the adoption of a particular language or technique, has certain economic or pedagogical benefits. Randomized controlled trials are the gold standard of evidence for this kind of claim, and repeatability requires careful attention to sample size and selection, control of confounding factors, and so on. Robust empirical claims require robust empirical evidence.

However, it is somewhat narrow-minded to expect all PL research to follow this empirical pattern. Not all scientific claims concern the effectiveness of an intervention on human subjects, and so not all papers are suitable targets for standards such as CONSORT and WWC. In particular, it is not appropriate to criticize conferences such as ICFP for a "lack [of] empirical foundation" when most of the papers published there do not make empirical claims. One might as well criticize Einstein's "Does the Inertia of a Body Depend upon Its Energy Content?," Turing's "On Computable Numbers, with an Application to the Entscheidungsproblem," and Watson and Crick's "A Structure for Deoxyribose Nucleic Acid," for such "methodological irregularities."

There is much more to PL research than empirical claims of the kind that Stefik and Hanenberg have in mind—for example, mathematical semantics of language features, verification and proof of correctness, type systems to allow the formal statement of certain properties of programs, static and dynamic analysis to check those properties, systems building and engineering design, compiler and performance optimizations, implementation techniques, and so on. For a longer discussion, see for example the "PL Enthusiast" blog post by Michael Hicks (www.pl-enthusiast.net/2015/05/27/what-is-pl-research-and-how-is-it-useful).

Jeremy Gibbons
Jeremy.Gibbons@cs.ox.ac.uk

Authors' response:

Jeremy Gibbons emphasizes that programming language researchers work on many aspects of language design. Sometimes, these considerations are fleshed out mathematically, like imagining a new semantics or working with verification, type systems, or something else. By declaring that such topics do not require empirical validation, Gibbons implies that programming language constructs are pure mathematical constructs, ignoring that the vast majority of programs are written by people.

Other scientific fields balance mathematical formalism and empirical data. In physics, including with Einstein's work, scholars have gone to great lengths to validate or refute the theories. The empirical work regarding Einstein's theories is quite famous and important. Other exemplars involving Faraday, Maxwell, and others also weave a more accurate story about how formalism and empirical data have been balanced over time in a field like physics. Most fields have such a balance for a variety of reasons. For example, nature is not guaranteed to agree with a formula or explanation imagined by a scholar.

Second, the focus of programming language research is generally no longer on the development of a calculus to understand what computation is, as it was during Turing's time. There is more to learn, but we now know

quite a bit about computation. Programming language research today, amongst other things, is often about providing tools and techniques to make programming, or programs, more efficient or less error-prone. Further, many kinds of programming languages have mathematically valid semantics, or sound type systems, but this does not mean their benefits and costs to society are known. Given these costs are in the billions and they impact millions of people, empirical evidence is needed to sort out the facts.

Finally, Gibbons argues that not all programming language researchers need to use empirical evidence or measure the human impact, which we agree with. However, empirical data is hardly an all-or-nothing proposition. We cited evidence that the amount of data gathered by the programming language research community on human factors approaches zero. While it is clear that "for all" is not necessary, pointing out meticulous observations from other scholars that suggest the value is "near zero" seems pretty reasonable. After all, any group or person that uses a programming language should be aware that little or no human testing has occurred in the language community. Acknowledging the observable fact that we lack evidence today is the first uncomfortable step in recognizing that change is needed for the benefit of the field tomorrow.

Formal statements are part of the picture, but they tell us little about the impacts specific technologies have on people or communities. Put another way, yet another proof of something like type soundness does not give us enough information about the impact of a language, feature, or technique. In our view, a larger and much more diverse collection of evidence-based ideas is needed.

Andreas Stefik
stefika@gmail.com
Stefan Hanenberg
stefan.hanenberg@uni-due.de

COMMENTS?
Please send us your letters to the editor at letters@computer.org.

ELSEWHERE IN THE CS

EDITOR **LORI CAMERON**
l.cameron@computer.org

Computer Highlights Society Magazines

The IEEE Computer Society's lineup of 13 peer-reviewed technical magazines covers cutting-edge topics ranging from software design and computer graphics to Internet computing and security, from scientific applications and machine intelligence to cloud migration and microchip design. Here are highlights from recent issues.

Computing in Science & Engineering

Math.Js: An Advanced Mathematics Library for JavaScript

Math.js is a JavaScript library that brings advanced mathematics to the web browser and server. The case study presented in this article from the January/February 2018 issue of *CiSE* demonstrates its flexibility by extending the library with custom functions to solve and optimize a rocket trajectory. Several benchmark comparisons with other JavaScript libraries and state-of-the-art mathematics software are presented, as well as the current challenges facing math.js, including performance and size.

IEEE Annals of the History of Computing

Imagining the Personal Computer: Conceptualizations of the Homebrew Computer Club 1975–1977

The Homebrew Computer Club was a hobbyist group in the San Francisco Bay Area dedicated to helping people build their own home personal computers. In this article from the October–December 2017 issue of *IEEE Annals*, Elizabeth Petrick from the New Jersey Institute of Technology analyzes their writings between 1975 and 1977 to understand how their values became embedded in the technology they built, establishing how the personal computer should be used and thought of. These values were based on ideals of open information, access to computers, and the computer as a universal tool, while also allowing for development of entrepreneurial ambitions to market the computer as a consumer product.

IEEE Cloud Computing

Context-Aware Ubiquitous Biometrics in the Edge of Military Things

Edge computing can play a crucial role in enabling user authentication and monitoring through context-aware biometrics in military and battlefield applications. For example, in the Internet of Military Things or Internet of Battlefield Things, an increasing number of ubiquitous sensing and computing devices worn by military personnel and embedded within military equipment—such as combat suits, instrumented helmets, and weapons systems—are capable of acquiring a variety of static and dynamic biometrics like visual features, fingerprints, heart rate, gait, gestures, and facial expressions. Such devices might also be capable of collecting operational context data that can be used to perform context-adaptive authentication in the wild and continuous monitoring of soldiers' mental and physical conditions in a dedicated edge computing architecture. Learn more in this article from the November/December 2017 issue of *IEEE Cloud Computing*.

IEEE Computer Graphics and Applications

A Generative Audio-Visual Prosodic Model for Virtual Actors

An important problem in the animation of virtual characters is the expression of complex mental states using the coordinated prosody of voice, rhythm, facial expressions, and head and gaze motion. The authors of this article from the November/December 2017 issue of *IEEE CG&A* propose a method for generating natural speech and facial animation in various attitudes using neutral speech and animation as input.

0018-9162/18/$33.00 © 2018 IEEE

IEEE Intelligent Systems

Robots in Retirement Homes: Person Search and Task Planning for a Group of Residents by a Team of Assistive Robots

In this article from the November/December 2017 issue of *IEEE Intelligent Systems*, researchers from the University of Toronto present a general multi-robot task planning and execution architecture for a team of heterogeneous mobile robots that interact with multiple human users. The architecture is implemented in an environment where such robots provide daily assistance to residents in a retirement-home setting. The robots are able to allocate and schedule activities throughout the day and find the appropriate residents with whom to engage in assistive activities.

IEEE Internet Computing

Internet of Things–Enhanced User Experience for Smart Water and Energy Management

Smart environments can engage a wide range of end users with different interests and priorities, from corporate managers looking to improve the performance of their business to school children who want to explore and learn more about the world around them. Creating an effective user experience within a smart environment (from smart buildings to smart cities) is an important factor to success. In this article from the January/February 2018 issue of *IEEE Internet Computing*, researchers reflect on their experience of developing Internet of Things–enabled applications within a smart home, school, office building, university, and airport, where the goal has been to engage a wide range of users (from building managers to business travelers) to increase water and energy awareness, management, and conservation.

IEEE micro

High-Integrity Performance Monitoring Units in Automotive Chips for Reliable Timing Validation and Verification

As software continues to control more system-critical functions in cars, its timing is becoming an integral element in functional safety. Timing validation and verification (V&V) assesses software's end-to-end timing measurements against given budgets. The advent of multicore processors with massive resource sharing reduces the significance of end-to-end execution times for timing V&V and requires reasoning on worst-case access delays on contention-prone hardware resources. While performance monitoring units (PMUs) support this finer-grained reasoning, their design has never been a prime consideration in high-performance processors. In this article from the January/February 2018 issue of *IEEE Micro*, researchers advocate for PMUs in automotive chips that: explicitly track activities related to worst-case software behavior, are recognized as a mandatory high-integrity hardware service, and are accompanied with detailed documentation that enables their effective use to derive reliable timing estimates.

IEEE MultiMedia

Word of Mouth Mobile Crowdsourcing: Increasing Awareness of Physical, Cyber, and Social Interactions

By fully exploring various sensing capabilities and multiple wireless interfaces of mobile devices and integrating them with human power and intelligence, mobile crowdsourcing (MCS) is emerging as an effective paradigm for large-scale multimedia-related applications. However, most MCS schemes use a direct mode, in which crowd workers passively or actively select tasks and contribute without interacting and collaborating with each other. This can hamper some time-constrained crowdsourced tasks. In this article from the October–December 2017 issue of *IEEE MultiMedia*, researchers from universities in China, Japan, and Sweden execute a different approach: MCS based on word of mouth (WoM), in which workers, apart from executing tasks, exploit their mobile social networks and/or physical encounters to actively recruit other appropriate individuals to work on the task.

IEEE Pervasive Computing

Designing Line-Based Shape-Changing Interfaces

In this article from the October–December 2017 issue of *IEEE Pervasive Computing*, researchers from Stanford and the MIT Media Lab present an overview of work on shape-changing line interfaces in the field of human–computer interaction (HCI), including their previous work on actuated-line interfaces (LineFORM and ChainFORM). They compare several potential implementation methods, discuss their potential for future research and applications, investigate the interaction design space around actuated line interfaces, and present potential applications and demonstrate their use with the LineFORM and ChainFORM prototypes. Envisioning a future where shape-changing lines are woven into daily life, this article aims to explore and initiate a broad research space around line-based shape-changing interfaces and to encourage future researchers and designers to investigate these novel directions.

IEEE SECURITY & PRIVACY

Enhancing Selectivity in Big Data

Today's companies collect immense amounts of personal data and enable wide access to it within the company. This exposes the data to external hackers and privacy-transgressing employees. In this article from the January/February 2018 issue of *IEEE S&P*, researchers show that, for a wide and important class of workloads, only a fraction of the data is needed to approach state-of-the-art accuracy. They propose selective data systems that are designed to pinpoint the data that is valuable for a company's current and evolving workloads. These systems limit data exposure by setting aside the data that is not truly valuable.

IEEE Software

Actionable Analytics for Strategic Maintenance of Critical Software: An Industry Experience Report

NASA has been successfully sustaining the continuous operation of its critical navigation software systems for over 12 years. To accomplish this, NASA scientists must continuously monitor their process, report on current system quality, forecast maintenance effort, and sustain required staffing levels. In this article from the January/February 2018 issue of *IEEE Software*, the authors present some examples of the use of a robust software metrics and analytics program that enables actionable strategic maintenance management of a critical system (Monte) in a timely, economical, and risk-controlled fashion.

IT Professional

Automatic Annotation of Text with Pictures

The vast array of information available on the Internet makes it challenging to quickly determine the importance and relevance of content. Text picturing is a cognitive aid that can help with text understanding, as it helps users decide if the text deserves a closer look by showing relevant pictures along with the text. Learn more in this article from the January/February 2018 issue of *IT Professional*.

IEEE computer society

PURPOSE: The IEEE Computer Society is the world's largest association of computing professionals and is the leading provider of technical information in the field.
MEMBERSHIP: Members receive the monthly magazine *Computer*, discounts, and opportunities to serve (all activities are led by volunteer members). Membership is open to all IEEE members, affiliate society members, and others interested in the computer field.
COMPUTER SOCIETY WEBSITE: www.computer.org
OMBUDSMAN: Direct unresolved complaints to ombudsman@computer.org.
CHAPTERS: Regular and student chapters worldwide provide the opportunity to interact with colleagues, hear technical experts, and serve the local professional community.
AVAILABLE INFORMATION: To check membership status, report an address change, or obtain more information on any of the following, email Customer Service at help@computer.org or call +1 714 821 8380 (international) or our toll-free number, +1 800 272 6657 (US):

- Membership applications
- Publications catalog
- Draft standards and order forms
- Technical committee list
- Technical committee application
- Chapter start-up procedures
- Student scholarship information
- Volunteer leaders/staff directory
- IEEE senior member grade application (requires 10 years practice and significant performance in five of those 10)

PUBLICATIONS AND ACTIVITIES
***Computer*:** The flagship publication of the IEEE Computer Society, *Computer*, publishes peer-reviewed technical content that covers all aspects of computer science, computer engineering, technology, and applications.
Periodicals: The society publishes 13 magazines, 19 transactions, and one letters. Refer to membership application or request information as noted above.
Conference Proceedings & Books: Conference Publishing Services publishes more than 275 titles every year.
Standards Working Groups: More than 150 groups produce IEEE standards used throughout the world.
Technical Committees: TCs provide professional interaction in more than 30 technical areas and directly influence computer engineering conferences and publications.
Conferences/Education: The society holds about 200 conferences each year and sponsors many educational activities, including computing science accreditation.
Certifications: The society offers two software developer credentials. For more information, visit www.computer.org/certification.

NEXT BOARD MEETING

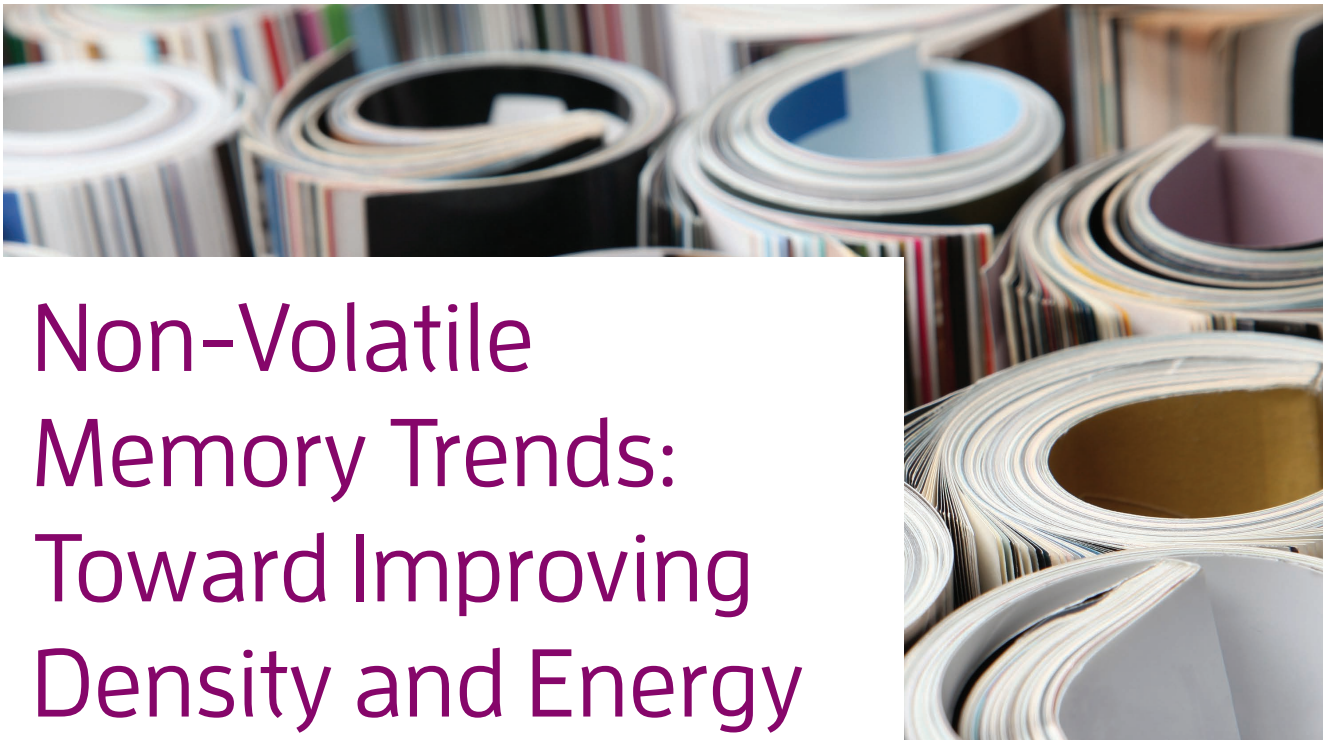
7-8 June 2018, Phoenix, AZ, USA

COMPUTER SOCIETY OFFICES

Washington, D.C.: 2001 L St., Ste. 700, Washington, D.C. 20036-4928
Phone: +1 202 371 0101 • **Fax:** +1 202 728 9614
Email: hq.ofc@computer.org
Los Alamitos: 10662 Los Vaqueros Circle, Los Alamitos, CA 90720 **Phone:** +1 714 821 8380
Email: help@computer.org

MEMBERSHIP & PUBLICATION ORDERS
Phone: +1 800 272 6657 • **Fax:** +1 714 821 4641 • Email: help@computer.org
Asia/Pacific: Watanabe Building, 1-4-2 Minami-Aoyama, Minato-ku, Tokyo 107-0062, Japan
Phone: +81 3 3408 3118 • **Fax:** +81 3 3408 3553
Email: tokyo.ofc@computer.org

revised 7 Feb. 2018



Non-Volatile Memory Trends: Toward Improving Density and Energy Profiles across the System Stack

Ronald F. DeMara, University of Central Florida
Paolo Montuschi, Polytechnic University of Turin

This installment of Computer's *series highlighting the work published in IEEE Computer Society journals comes from* IEEE Transactions on Computers.

Computer architects are well aware of not only the essential role served by memory devices but also their impact on overall system performance. The infamous "memory wall" illustrates a classic case-in-point. Recently, with the advent of mobile and low-power devices, as well as high-end data centers and large on-chip caches, another high-priority demand has emerged: non-volatile, dense, and low-energy-consuming memories. *IEEE Transactions on Computers* (*TC*) has continued to lead in this new research area. In 2017, *TC* published more than a dozen papers on this topic and commissioned a special section titled "Emerging Non-Volatile Memory Technologies: From Devices to Architectures and Systems." Approximately 60 submissions were received, and the accepted papers will be published in the first half of 2019. Given the interest and potential in non-volatile memory (NVM), what are some technical challenges facing its wider utilization?

At the circuit-level, increasing reliability is a prominent focus. It is sought to maximize NVM's resistive sensing margin, which is a measure of resilience between bit-level "data zero" and "data one" storage. One approach is to employ a pseudo-differential sensing (PDS) framework using an asymmetric sensing amplifier with self-voting error-detection correction to increase the read margin

0018-9162/18/$33.00 © 2018 IEEE

EDITOR **RON VETTER**
University of North Carolina Wilmington; vetterr@uncw.edu

by 35 percent while reducing area, read latency, and read energy.[1] For instance, PDS using a single transistor with a three-magnetic tunnel junction (1T3MTJ) cell cluster shows increased reliability and performance compared to a typical 1T1MTJ cell structure.[1] Research is ongoing on new mechanisms to increase NVM resilience using crosslayer approaches up through the system level.

At the system level, one challenge is the high write energy costs relative to SRAM that NVM technologies incur. Thus, last-level cache (LLC) using NVM for increased density while reducing leakage energy consumption and cooling demands of multicore and many-core dies must deal with the increased energy cost of writes. To reduce the frequency of writes, new system-level techniques based on various adaptive restore schemes (ARSs) are being developed for NVM technologies. ARSs alleviate restoration overhead by overwriting soft bit lines that are less likely to be read, or else upon eviction from higher-level cache. These techniques can decrease STT-MRAM energy consumption by 17 percent, while increasing instructions per cycle (IPC) by 9 percent across the PARSEC benchmark suite.[2]

Utilization of NVM also introduces new challenges to file structures and OSs, such as facilitating the storage, migration, and management of long-lifetime data throughout the memory hierarchy. Memory policies, data structures, and file systems capable of leveraging NVM continue to advance. For instance, storage subsystems in mobile processors can revise protocols for swapping with faster, byte-addressable NVM to support lazy swap-in. These techniques reduce memory copy operations by giving swapped-out pages a second chance to reside in byte-addressable NVM-backed swap areas. These new protocols can be coupled with resource-management methods to distribute or forego costly writes to NVM. Evaluation on various smartphones indicates that application relaunching delay and execution time can be reduced by 12 to 45 percent with improved wear-leveling characteristics.[3]

In *IEEE TC*, we will continue to address recent NVM developments from the perspectives of industry and academia. Please stay tuned for the newest issues of *TC* to keep current on these advances in memory, as well as in computing architectures, secure and real-time systems, and much more.

REFERENCES

1. W. Kang et al., "Pseudo-Differential Sensing Framework for STT-MRAM: A Cross-Layer Perspective," *IEEE Trans. Computers*, vol. 66, no. 3, 2017, pp. 531–544.
2. X. Chen et al., "Energy-Aware Adaptive Restore Schemes for MLC STT-RAM Cache," *IEEE Trans. Computers*, vol. 66, no. 5, 2017, pp. 786–798.
3. D. Liu et al., "Non-Volatile Memory Based Page Swapping for Building High-Performance Mobile Devices," *IEEE Trans. Computers*, vol. 66, no. 11, 2017, pp. 1918–1931.

RONALD F. DEMARA is a professor of electrical and computer engineering at the University of Central Florida. Contact him at ronald.demara@ucf.edu or visit http://cal.ucf.edu.

PAOLO MONTUSCHI is a professor of computer engineering at Polytechnic University of Turin. Contact him at pmo@computer.org or visit http://staff.polito.it/paolo.montuschi.

50 & 25 YEARS AGO

EDITOR **ERICH NEUHOLD**
University of Vienna
erich.neuhold@univie.ac.at

APRIL 1968

In its early years, *Computer* was published bimonthly. Stay tuned for more interesting historical highlights in the upcoming May 2018 issue.

APRIL 1993

www.computer.org/csdl/mags/co/1993/04/index.html

Chief Editor's Message: Education's Role in the Economy (p. 6) "Recognizing the waste and inefficiencies of the current system, Workforce 2000 proposes to revise the traditional K–12 track, reinventing it as a modern K–10 track followed by a college preparatory or vocational option. Students are graded at roughly three-year intervals, and after the tenth year, they opt to enter either a vocational school (run by the community colleges) or a two-year college-prep program. Vocational school prepares students for high-wage jobs and can take from two to six additional years to complete. It is modeled after similar highly successful programs in Europe." *[Editor's note: Unfortunately, the idea of following the "European model" didn't work out. Instead, after signing the Bologna Accord, Europe decided to follow the American bachelor–master model, giving up many of the advantages of the European educational system, including the principle that government ought to pay for education.]*

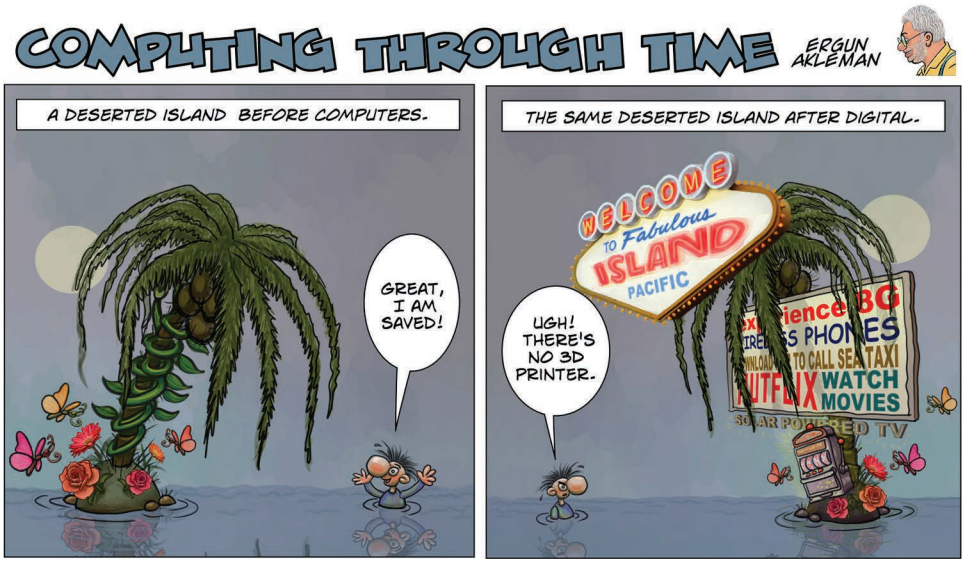
MCM: The High-Performance Electronic Packaging Technology (p. 10) "Although multichip module (MCM) technology is decades old, it has recently caught the attention of the research community again. This resurgence of interest is due to increasing awareness that packaging sets important bounds on system cost and performance. Some researchers, in fact, expect that MCMs will soon provide at least an order-of-magnitude improvement in performance over traditional packaging technology. ... Recently, MCMs have been applied to portable computers and telecommunication systems, as shown by Franzon and Evans. These applications have been driven to MCM technology to fill three needs: high performance, small size, and low cost. Advantages [include] performance, mixed technologies, size, and reliability. Challenges [include] (software) performance, thermal management, reparability, testing, and economics." *[Editor's note: The articles in this special issue explored these challenges. Looking back, tremendous progress has been made in circuit density, multicore technologies, and systems on a chip. However, the challenges mentioned remain valid, with additional concerns about security and utilization.]*

Development of a Multichip Module DSP (p. 13) "The unit was developed under the Signal Processor Packaging Design (SPPD) program sponsored by the Guided Interceptor Technology branch of the Air Force Wright Laboratory's Armament Directorate at Eglin. ... Experience shows that without an overall MCM test approach and particular attention to die testing, an MCM project can remain in the laboratory forever. Indeed, it became clear during the SPPD development that without some enhancement of the die testing offered by chip suppliers, the module yield would be unacceptable. ... The SPPD wafer measures 2.75 inches across, has 450 wafer 1/0 (216 pack-age 1/0), connects 400 nets on two signal layers using 0.001-inch-wide traces, and houses 28 major components with 40 percent silicon efficiency."

A Multichip Module Design Process for Notebook Computers (p. 41) "Multichip modules can obtain significant performance improvements with a reduced footprint. This article examines the packaging-system design process, with application to choosing an MCM card for a 386SL notebook computer. ... Minimizing thickness will then require that the PCB (Printed Circuit Board) be populated on one side only. To maximize the number of slots reserved for memory cards and other cards (like a modem), one must minimize the PCB area." *[Editor's note: This article investigates layout and placement techniques oriented toward notebook computers. With the advent of tablets, smartphones, and Internet of Things devices, the problem remains active today.]*

0018-9162/18/$33.00 © 2018 IEEE

An Integrated Multicomponent Synthesis Environment for MCMs (p. 62) "This vertically integrated design environment supports the development of application-specific MCMs that involve low-volume production and short lead times. ... We have been developing an integrated design environment for multichip modules. The environment, called MSS (Multicomponent Synthesis System), contains several research tools developed over the past three years along with several industrial-strength tools."

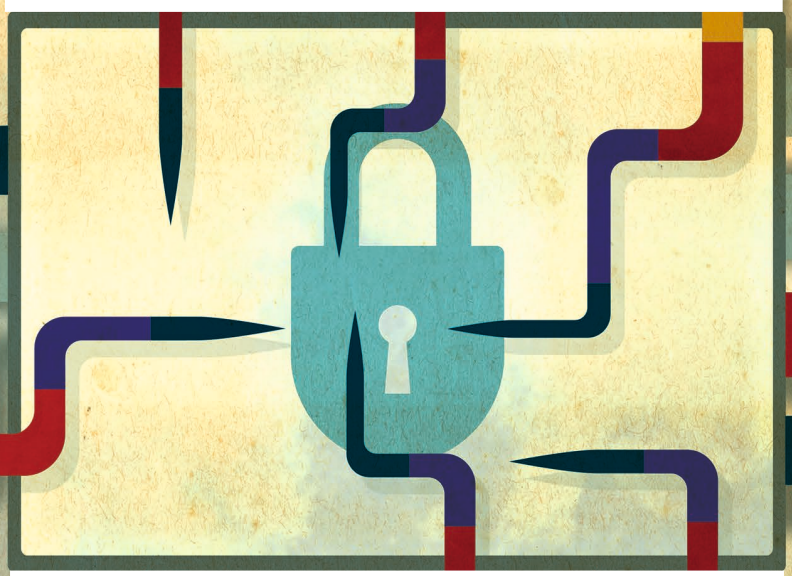


A Road Map to ARPA Involvement in Electronic Packaging (p. 82) "Advanced electronic packaging and interconnect (EP/I) technology is critical to the success of the US Defense Department's Advanced Research Projects Agency [ARPA] strategic plan. This report presents a road map to EP/I at ARPA. The EP/I domain includes packages, multichip modules (MCMs), connectors, printed circuit boards (PCBs), boxes, backplanes, and cages. ... Four programs center on EP/1: Diamond Substrates, Superconducting MCMs, Physical Electronic Packaging, and Application-Specific Electronic Modules. Many other programs have important roles, however, and each will be discussed." *[Editor's note: In 1993, ARPA's support of such programs was important to advance the state of the art. However, later substantial progress was achieved as a result of industry's need to produce increasingly powerful yet small devices.]*

Meeting the Grand Challenges (p. 88) "Scientists estimate that many computationally complex problems will nevertheless require computing power to increase by a factor of more than 100. The term 'Grand Challenges' refers to the class of computational applications requiring this kind of increase. ... In weather forecasting, a typical calculation treats the weather of the six-county area in California extending from Santa Barbara to San Diego as a single grid point. This assumes that the weather is the same all over that area, which is not the case. A much finer grid is needed." *[Editor's note: Of course, advances in computing power have been tremendous, and weather forecasting, for example, now works with much finer grids. However, grand challenges remain, having moved on to more advanced goals in the modeling of computationally challenging problems.]*

Luggable Multimedia (p. 91) "Micro Express has introduced the Multimedia Portable PC, which features VESA standard local bus graphics and a 0.26-mm dot pitch Sony Trinitron color CRT. The 33-MHz 486DX PC has 4 Mbytes of RAM, a 250-Mbyte IDE fast hard-disk drive, 5.25- and 3.5-inch high-density floppy-disk drives, a Sound Blaster card, two externally powered stereo speakers, an internal hi-fi speaker, a Sony CD-ROM drive, a mouse, Windows 3.1, and DOS 5.0. The monitor can provide up to 1,024 × 768-pixel resolution, measures 8.5 inches diagonally, and features brightness and contrast controls. ... The 30-pound system also comes with a carrying bag. The multimedia PC costs $3,250." *[Editor's note: No comment on progress is needed here!]*

Cybersecurity:



The Security and Protection Challenges of Our Digital World

0018-9162/18/$33.00 © 2018 IEEE

Daniel Prince, Lancaster University

Once the domain of national agencies empowered to protect civil society and support intelligence missions, cybersecurity is now the pervasive technical, social, and economic issue. It affects all aspects of our daily life and organizational functions. From financial institutions to smart homes, and from the classroom to the doctor's office, we are in a cyber arms race to protect ourselves from those driven to do us harm.

The utopian world we were promised—the one in which the advent of digital technology solved our social problems—is now rather harshly tempered by challenges and issues related to security. Although it is designed to make life better, or at least more efficient, technology lacks immunity from the people and organizations actively devising to do us harm. Moreover, these malicious actors are just as smart and driven as those heralded as the beneficent leaders of digital innovation and disruption.

Digital infrastructures increasingly pervade our lives, and this trend is set to continue. The resulting cyberspace in which so many of our transactions and interactions increasingly take place provides a perfect perch for society's criminal element to set up shop—there is not only plenty of opportunity for criminal exploitation but there is also less risk of being caught. In 2014, the RAND Corporation reported on the maturity of cybercrime markets and those operating it,[1] with some senior commentators indicating this market could rival the international drugs trade in size.[2]

Thus, we must operate under the assumption that criminals and their networks are ready to exploit us whenever we make a simple mistake in cyberspace. And if that is not enough to be concerned about, we also face challenges in trust vis-a-vis the public agencies specifically charged with protecting society. The Snowden revelations,[3] the theft and disclosure of NSA cyberattack tools,[4] and other similar revelations continue to erode this trust. What is interesting is how the large technology providers have stepped in as purported protectors of our digital lives. Whether it is Apple refusing to break its encryption for law enforcement,[5] or Microsoft fighting back for email privacy,[6] these massive companies now present themselves as the bastions of the average person's digital rights.

Globalization—a core tenet in the expansion of digital and technology markets—is also a source of many underlying security concerns. Regardless of where hardware or software is produced, the same products are being used everywhere. Security failures and related criminal activity in one product now affect a massive swath of the global populace. This makes it challenging for law enforcement agencies to use traditional investigative practices. In addition, a secure system's backdoor for legal practices is an obvious target for criminals.

Beyond transactional cybercrimes, new techniques and tools that exploit the pervasive digital infrastructure itself are increasingly common. The revelation that a nation-state influenced the outcome of the 2016 American election brings this into full view, and it illustrates an alarming new dimension of cyberthreat. Although social engineering as a tool of attack has long been understood in the cybersecurity profession, such approaches are now highly automated and delivered at a scale, resulting in significant geopolitical repercussions. Thus it is imperative to move toward new paradigms in understanding to effectively frame the techno-socio-political debate, such as the cyber-catastrophist, digital-realist, and techno-optimist[7] to help us explore critical positions beyond technological utopian view of the world.

It is against this socio-political backdrop that technologists and

computer scientists must undertake their security research. Many have argued[8-10] that socio-political context and market forces within an economy are core drivers for many security challenges. Perhaps it is a lack of fully understanding such factors that leads to key security failures in software and hardware, despite our expertise in the basics of engineering and building secure systems.

The recent discovery of the Spectre and Meltdown vulnerabilities in nearly all CPUs in use today presents another significant security challenge: a shift in focus for attacks to include hardware as well as software. This shift presents a new set of risks to our digital environment, especially as cyber-physical systems are increasingly common, running everything from our smart homes to our smart health applications. Evidence from the world of industrial control systems has shown that current risk management approaches do not go far enough,[11,12] and that our attitude toward technology changes dramatically depending on the context in which we are using it.[13]

In the future, the technology we design and build will not fit neatly into a three- or four-year replacement cycle that enforces security updates and enhancements. Instead, the Internet of Things (IoT) will mean we rely on some devices to be deployed for decades. This results in potential hazards accruing in the digital (not to mention physical) environment like never before, driving an increase in the totality of the risk in the system(s). It is incumbent upon us all to make sure that security and safety is not a "nice to have," but a fundamental feature of all the systems we think about, build, and operate. Only when security and protection is at the heart of our approach

will we be able to get ahead of those who wish to do us harm or be resilient in the face of failures in the complex digital system that supports our daily lives, and this issue explores this concept in a number of applications.

IN THIS ISSUE

In "End-to-End Trust and Security for Internet of Things Applications," Sulabh Bhattarai and Yong Wang explore the numerous security challenges of multiple IoT platforms that utilize the cloud and interconnect via the cloud. Many IoT platforms use a cloud-based infrastructure to provide support to the one site activity. This can range from data capture to providing near-real-time control. For example, Amazon Web Services are used to drive the Alexa voice assistant, which is connected to the could architecture of Samsung SmartThings to automate home IoT devices, which also connects to a cloud-based web cam security system to provide automatic capture. The authors argue for the need to consider end-to-end security when developing IoT platforms. They also present a new threat model to aid in the development of IoT cloud-based systems, by supporting designers in thinking through the security challenges such systems may face during operation.

In "Game-Model-Based Network Security Risk Control," Zhen Ni, Qianmu Li, and Gang Liu look to advance our thinking in the application of game theory to modeling the optimal attack and defense strategies using an example network. The aim of the work is to support decision makers in determining where the best investments in security would be to reduce the overall risk to the system. The development of an optimum attack-defense decision-making algorithm was successful in effectively identifying the optimum attack-defense strategy, thus supporting the decision-making process and plans.

In "Detecting Code Reuse Attacks with Branch Prediction," Yongsuk Lee and Gyungho Lee propose a new approach to protecting against attackers seeking to subvert the flow of code execution on modern machines. As we have become more proficient at protecting against arbitrary code execution—with mechanisms such as data execution prevention—attackers have moved to use what is, for all intents and purposes, legitimate code, as well as exploiting the complexity of modern processes in their use of predictive branch execution. The authors advocate an inline approach to ensure legitimate control flow, by incorporating control-flow validation into

the processor's instruction execution pipeline in a way that has little performance overhead.

This collection of articles explores the range of challenges security strategies must address. The complexity of the systems we are designing and operating, from chip level to city scale, are challenging us on every level. Whether this is the arms race that is protecting the execution of legitimate code, new challenges in cyber-physical and IoT systems, or developing new approaches to support decision makers to get the best defense for their budget. The security challenges we face now and in the future are broad, varied, and complex.

We hope you will gain insight from these articles and the wide range of challenges they represent. Moreover, we hope to inspire you to continue to drive forward the innovation needed in all parts of the socio-technical system that powers the modern world so that we might all enjoy the benefits it brings without suffering from the costs of security failures.

ABOUT THE AUTHOR

DANIEL PRINCE is a senior lecturer in cybersecurity at Lancaster University. His research interests include cybersecurity risk management, communication and perception, as well as the security of networks, specifically next-generation converged networks, which include cyber-physical systems. Prince received a PhD in computer science from Lancaster University. Contact him at d.prince@lancaster.ac.uk.

REFERENCES

1. L. Ablon, M.C. Libicki, and A.A. Golay, "Markets for Cybercrime Tools and Stolen Data," report, Rand Corporation, 2014; www.rand.org/content/dam/rand/pubs/research_reports/RR600/RR610/RAND_RR610.pdf
2. J. Dean, "Cybercrime Is Bigger than Drugs, Says Police Chief," *The Times*, 14 April 2015; www.thetimes.co.uk/article/cybercrime-is-bigger-than-drugs-says-police-chief-vhdcrw6ms3s.
3. "The NSA Files," collection of articles, *The Guardian*; www.theguardian.com/us-news/the-nsa-files.
4. M. Burgess, "Hacking the Hackers: Everything You Need to Know about Shadow Brokers' Attack on the NSA," *Wired*, 18 April 2017; www.wired.co.uk/article/nsa-hacking-tools-stolen-hackers.
5. D. Yadron, S. Ackerman, and S. Thielman, "Inside the FBI's Encryption Battle with Apple," *The Guardian*, 17 Feb. 2016; www.theguardian.com/technology/2016/feb/17/inside-the-fbis-encryption-battle-with-apple.
6. L. Matsakis, "Microsoft's Supreme Court Case Has Big Implications for Data," *Wired*, 27 Feb. 2018; www.wired.com/story/us-vs-microsoft-supreme-court-case-data.
7. M.J. Lacy and D.D.C. Prince, "Securitization and the Global Politics of Cybersecurity," *Global Discourse*, 2017; doi: 10.1080/23269995.2017.1415082.
8. R. Anderson, "Why Information Security Is Hard—An Economic Perspective," *Proc. 17th Annual Computer Security Applications Conference* (ACSAC 01), 2001; https://www.acsac.org/2001/papers/110.pdf.
9. L.A. Gordon and M.P. Loeb, "The economics of information security investment," *ACM Transactions on Information and System Security,* vol. 5, no. 4, 2002, pp. 438–457; doi:http://dx.doi.org/10.1145/581271.581274.
10. L.J. Camp and S. Lewis, editors, *The Economics of Information Security,* Kluwer, 2004.
11. B. Green et al., "How Long Is a Piece of String?: Defining Key Phases and Observed Challenges within ICS Risk Assessment," *Proc. ACM Workshop on Cyber-Physical Systems Security and Privacy* (CPS-SPC 17), 2017, pp. 103-109; doi: 10.1145/3140241.3140251.
12. B. Green et al., "The Impact of Social Engineering on Industrial Control System Security," *Proc. ACM Workshop on Cyber-Physical Systems Security and Privacy* (CPS-SPC 115), 2015, pp. 23-29; doi: 10.1145/2808705.2808717.
13. H. Jones, J. Towse, and N. Race, "Susceptibility to Email Fraud: A Review of Psychological Perspectives, Data-Collection Methods, and Ethical Considerations," *Int'l J. Cyber Behavior, Psychology and Learning*, vol. 5, no. 3, 2015, pp. 13–29; doi: 10.4018/IJCBPL.2015070102.

End-to-End Trust and Security for Internet of Things Applications

Sulabh Bhattarai and Yong Wang, Dakota State University

Many Internet of Things (IoT) devices utilize the cloud to store data and synchronize with one another. Because data breaches can occur anywhere en route to the cloud, ensuring end-to-end trust and security for IoT applications is critical. The authors propose a novel 3D threat model for the IoT and discuss nine security and privacy challenges as well as several ways to mitigate risks.

The Internet of Things (IoT) interconnects computing devices embedded in everyday objects. Enabled by advances in RFID, sensors, communication networks, and Internet protocols, the IoT is bridging diverse technologies and enabling new applications by connecting physical objects together in support of intelligent decision making.[1] An estimated 50 billion smart devices capable of sending and receiving data via the Internet will be connected worldwide by 2020.[2]

The IoT creates immense opportunities for economic development. According to the McKinsey Global Institute, the IoT has the potential to generate $3.9 trillion to $11.1 trillion a year in revenue by 2025.[3] Governments are working with various stakeholders to understand and harness the benefits of connected devices. For example, the White House's $400 million Advanced Wireless Research Initiative, led by the NSF, aims to deploy city-scale testing platforms for advanced wireless connections that could help the US achieve a ubiquitous network of devices and sensors.[2]

At the same time, the IoT presents major security challenges. The global costs of cybercrime jumped from an estimated $400–$500 billion in 2015 to $2–$3 trillion the following year.[4] One can imagine the enormous impact data breaches and security incidents will have in a world where billions of things—including vehicles, buildings, appliances, and mobile devices—are connected to the Internet. The threat to IoT devices is real. On 21 October 2016, a distributed denial-of-service (DDoS) attack was launched against Dyn, a critical technology service provider for some of the Internet's top destinations.[5]

0018-9162/18/$33.00 © 2018 IEEE

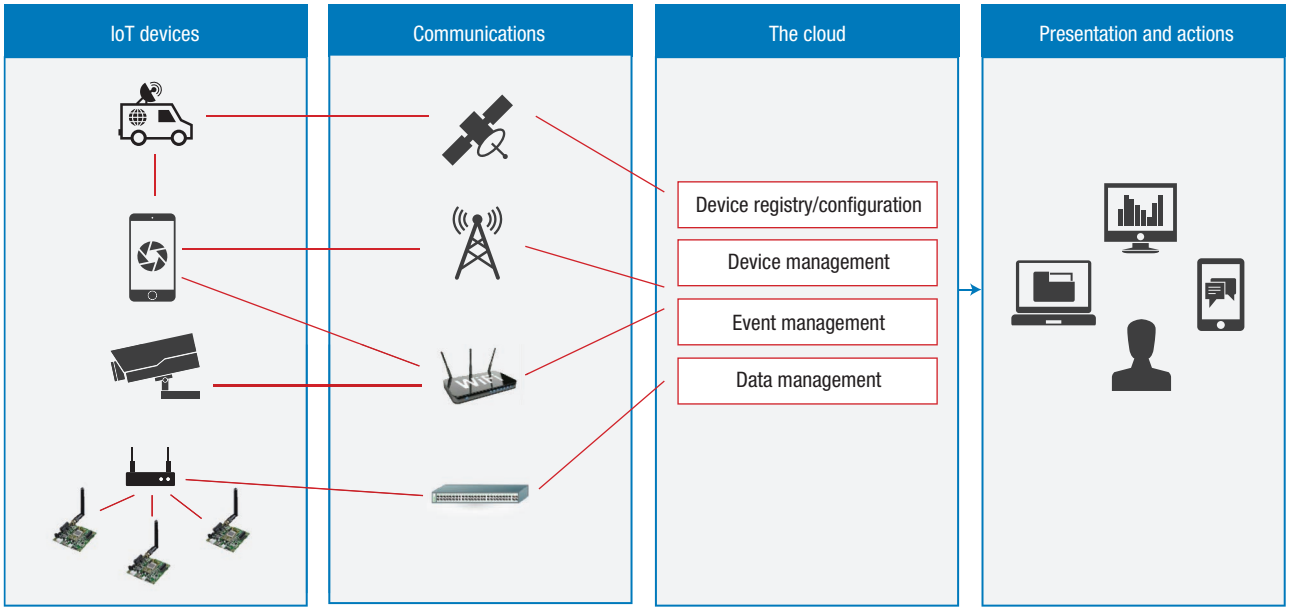


FIGURE 1. The architecture of Internet of Things (IoT) applications consists of four domains: IoT devices, communications, the cloud, and presentations and actions.

The attack traffic, which caused a major Internet outage on the US East Coast, came from the Mirai botnet, which consisted of 100,000 malicious IoT endpoints such as Internet routers, DVRs, and IP cameras. As more things become connected, IoT security becomes even more essential.[6,7]

IoT devices are resource constrained: they have limited memory, computation power, storage, and battery life. Many therefore utilize the cloud to store data and synchronize with one another. Securing IoT devices is important to ensure data security and privacy for IoT applications. However, such devices generate data that is transmitted over communication networks to a storage facility in the cloud, so data breaches can happen anywhere en route to the cloud. Thus, it is important to provide end-to-end trust and security for IoT applications.

Toward this, we propose a novel 3D IoT threat model and discuss existing security challenges as well as ways to mitigate security risks.

IOT APPLICATION ARCHITECTURE

Figure 1 shows the basic architecture of IoT applications.

Domains

The architecture consists of four domains: IoT devices, communications, the cloud, and presentation and actions.

IoT devices. This domain consists of resource-constrained devices ranging from wearables to healthcare devices. Devices come in many different sizes and run various embedded OSs and communication protocols. They are often deployed in open environments with little or no human intervention. The variety of IoT devices raises problems related to reliability, energy efficiency, and interoperability.

Communications. This domain provides Internet connectivity for IoT devices. Connecting these devices to one another and to the cloud is a major challenge for the underlying communication networks. Bluetooth, ZigBee, Wi-Fi, cellular networks, and power lines are the dominant IoT communication technologies, but others are available such as near-field communication (NFC) and Z-Wave.

The cloud. While IoT devices are resource constrained, the cloud has almost unlimited storage and processing power.[8] Consequently, the cloud commonly provides functions for the IoT such as device registry/configuration and device, event, and data management. The cloud and the IoT thus have a complementary relationship and are often viewed as two sides of the same coin. IoT devices are integrated with the cloud through APIs provided by a cloud service provider. AWS IoT (https://aws.amazon.com/iot) is an example of a cloud platform specifically designed for the IoT.

Presentation and actions. Data is presented to end users through cloud-based applications running on laptops or desktop computers. Management functions, such as activating and deactivating IoT devices, are also available through cloud-based applications for end users to manage IoT devices.

Communication models

The Internet Architecture Board (IAB), which oversees Internet development and growth, published RFC 7452 in March of 2015 (https://trac.tools.ietf.org/html/rfc7452). It outlines four communication models for IoT devices: device to device, device to gateway,

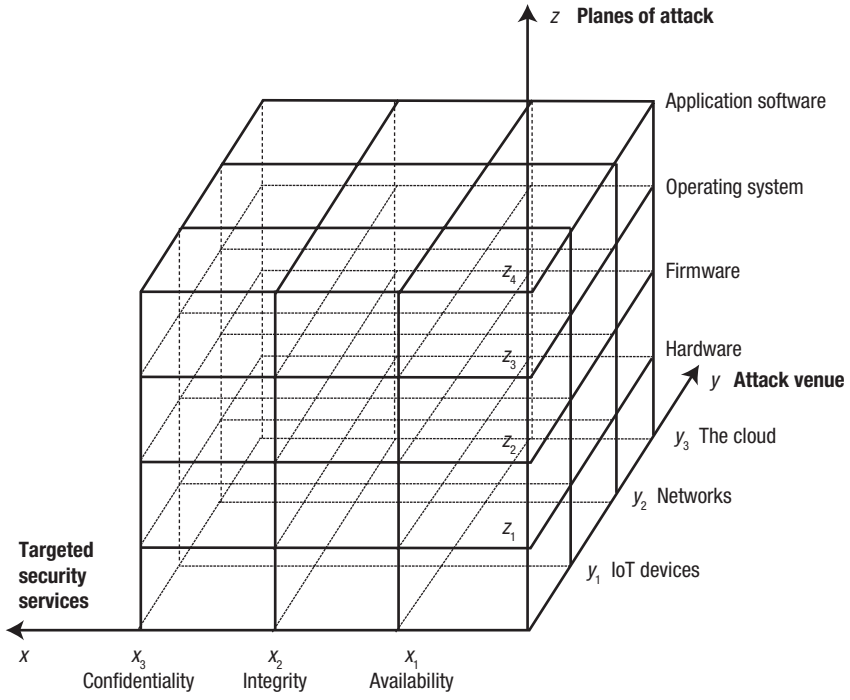


FIGURE 2. Proposed IoT threat model, with threats along three dimensions: targeted security services, attack venue, and planes of attack.

device to cloud, and backend data sharing.

Device to device. In this model, two devices directly connect to and communicate with each other rather than through an intermediary. Communications are often conducted through a wireless network using protocols such as Bluetooth or ZigBee. The device-to-device communication model is often suitable for home automation systems such as thermostats, smartwatches, and light bulbs wherein the packet size is small and the data throughput is relatively low.

Device to gateway. In this model, IoT devices access cloud services using intermediate gateways. Although gateway devices can come in many forms, they serve the same purposes: aggregating received data, bridging the gaps between devices using different communication protocols, implementing security, and forwarding data to the cloud. In some cases, a gateway device might relay the data to another gateway that will then forward the data to the cloud.

Device to cloud. In this model, customers devise rules to filter data generated from IoT devices and take actions accordingly through a cloud portal. For example, Microsoft Power BI (https://powerbi.microsoft.com/en-us) is a cloud service that lets nontechnical users analyze and visualize data generated by IoT devices. As these devices continuously generate data and transmit it to the cloud, network congestion can occur. The network traffic thus should be monitored constantly, and appropriate settings and actions must be taken to avoid traffic jams.

Backend data sharing. IoT devices often upload data to a particular cloud service provider, which can lead to data being isolated from other applications. The backend data-sharing model is an extension of the device-to-cloud communication model. It prevents data silos by allowing generated data to be shared among trusted parties. The backend data-sharing model also allows users to export, aggregate, and analyze data generated by IoT devices from other applications.

End-to-end trust and security

In a cloud-based IoT application, IoT devices generate and collect data. The data is then transmitted over communication networks to a storage facility in the cloud. Data breaches can happen anywhere en route to the cloud. Thus, it is essential to provide end-to-end trust and security for IoT applications. However, threats exist in each domain of the architecture. For example, in the IoT devices domain, devices can include hardware and software vulnerabilities. In the communications domain, eavesdropping and DoS attacks are possible. In the cloud domain, data breaches can occur due to lack of compliance with regulations. In the presentation and actions domain, attackers can steal a user's credentials and take over the IoT devices.

IOT THREAT MODEL

Figure 2 shows our proposed IoT threat model, which has three dimensions: targeted security services, attack venue, and planes of attack.

Targeted security services

Maintaining the confidentiality, integrity, and availability of data is the centerpiece of information security. *Confidentiality* means that only the authorized individual can view the sensitive information, *integrity* means that data is true and accurate and has not changed during storage or at transit, and *availability* means that data is readily available to the customers and businesses who need it. If malicious users find ways to compromise the confidentiality, integrity, and availability

of data, they can manipulate it for their own benefits. This will eventually damage end users' trust and could significantly impact business operations. This dimension could be expanded with other security services from ITU Recommendation X.800 (www.itu.int/rec/T-REC-X.800-199103-I/en), such as authentication and nonrepudiation.

Attack venue

As Figure 1 shows, a complete IoT network includes four essential parts: the devices that generate data, the networks that carry the data, the cloud services that process and mine the data, and the applications that consume the data. The applications are often cloud-based web applications that are available on any computing devices such as laptops and desktop computers. The application security is considered in the third dimension, the planes of attack. Thus, the attack venue includes three routes: IoT devices, communication networks, and the cloud.

IoT devices. The number of cyberattacks against IoT devices is rising. According to a report by Beaming, a UK business ISP, cyberattacks against IoT devices grew by 310 percent between the first and last quarter of 2016.[9] Billions of IoT devices are connected to the Internet and collect data in real time. Many of these devices have security vulnerabilities that make them desirable targets. In addition, IoT devices often operate without any human intervention and thus lack built-in security mechanisms to protect themselves from any form of attacks.

Communication networks. Common attacks in this category include interception, modification, false data injection, DoS, and replay attacks. For example, false data injection is an attack on data integrity in communication networks that can jeopardize end users' trust in a system.

The cloud. The cloud is used to store and process a large amount of data. Collected data might contain sensitive information that, if compromised, could threaten an organization's existence. End users generally access cloud data through cloud-based applications that have vulnerabilities similar to other web applications such as broken authentication, malicious code injection, and compromised credentials. Cloud providers can use authentication as the first line of defense against unauthorized data access but might not be able to enforce strong user credential requirements, which makes account enumeration attacks possible.

Planes of attack

The planes of attack include four layers: application software, OS, firmware, and hardware. To provide end-to-end trust and security for IoT applications, security must be enforced in each plane of attack for each attack venue. For example, to build a secure IoT device, manufacturers and developers must test security during the design and implementation phase of each layer.

Application software. An application running on a smartphone or website—for example, a healthcare app that monitors a patient's condition through wearable sensors—includes an interface for the end user to interact with the system. A poorly designed application can let unauthorized users take control of the device's administrative features. The inherent vulnerability arising from varying user inputs makes application layer attacks hard to defend against.

Operating system. Most devices connected to the Internet, from garage door openers to refrigerators, have their own OS. The open source community and various tech giants also promote their own IoT OS. For example, Windows 10 IoT (https://developer.microsoft.com/en-us/windows/iot) and Android Things (https://developer.android.com/things/index.html) are stripped-down versions of Windows 10 and Android from Microsoft and Google, respectively. To satisfy small memory footprints, vendors might be lax implementing security features, making IoT OSs vulnerable to attacks.

Firmware. Sophisticated attackers who gain physical access to an IOT device can read its internal memory and firmware through the onboard programmatic interface, giving them a better understanding of how the device works. They can use this knowledge to customize malicious firmware and upload it to the device to get full control over it. Attackers can also discover the device's cryptographic keys, vulnerabilities, and any backdoor in the firmware, enabling them to devise more effective attacks. Further, having physical access to the device lets attackers alter the configuration settings. They might be able to reset the device to the factory settings or install a custom-built SSL certificate to reroute traffic to their own server, enabling them to update the firmware using malicious firmware if the device accepts unsigned firmware.

Hardware. Software-based approaches are insufficient to ensure strong IoT security. An attacker who has physical

access to a device can tamper with its hardware to alter the security settings so that the device performs outside of the expected parameters, resulting in unreliable service. To ensure strong security, designing and building tamper-proof hardware is essential.

Extending the threat model

The ultimate goal of our 3D threat model is to provide a better view about where attacks can occur. Our analysis of IoT devices can be extended to include networking devices, such as routers and switches, and cloud platforms that integrate computing and communication components to provide software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS).

IOT SECURITY AND PRIVACY CHALLENGES

The US government, among others, has initiated steps to prevent the possibility of data breaches on IoT devices. For instance, the Department of Homeland Security provides six strategic principles for protecting such devices: incorporating security at the design phase, advancing security updates and vulnerability management, building on proven security practices, prioritizing security measures according to potential impact, promoting transparency across the IoT, and careful and deliberate connectivity.[10] Nevertheless, protecting data security and privacy remains difficult. Here, we outline nine challenges to IoT security and privacy.

Open architecture

The Internet is an open system that provides access to the public. The existing infrastructure and protocol stacks assume that all entities connecting to the Internet obey the principles of autonomy, fairness, and adaptiveness, leading to vulnerabilities that can be exploited by malicious users. We have already encountered huge challenges protecting security and privacy on the Internet, and these will be significantly magnified by connecting billions of devices to it.

System limitations

Most security services, including confidentiality, integrity, and authentication, are ensured by cryptographic techniques such as symmetric/asymmetric ciphers, message authentication code, hash functions, and digital signatures. Due to limitations of memory, computational power, and battery life in IoT devices, complicated cryptographic algorithms such as RSA cannot be used. In addition, all security mechanisms depend on secret keys—encryption/decryption keys, authentication keys, session keys, and so on—and distributing these keys securely in such a large-scale network presents a major obstacle. Further, as attacks continue to increase in sophistication, more security features must be integrated into hardware, which in turn will limit device functionality. Similarly, upgrading and developing countermeasures to cope with new attacks will remain a challenging issue on IoT devices.[11]

Lack of standardization

The variety of IoT devices is also an impediment to standardization. Each device is a standalone system that includes hardware, firmware, software, and communication interfaces. It is essential to incorporate security at the design phase, write secure code, and conduct thorough testing during the manufacturing process. However, there is no practical way to standardize and enforce these practices on each device. Although NIST provides general principles and practices for securing information technology systems,[12] there is a lack of security testing and auditing standards for the IoT. Even if such standards existed, conducting a security audit on each device would be impossible due to the heterogeneity and large volume of devices.

Insufficient trust and integrity

Each IoT device is a potential entry point for attacks, and given the large number of such devices expected to reach the market in the near future—up to 50 billion by 2020—it will be almost impossible to ascertain that each has proper controls and safeguards in place and is updated with the latest security patches. This raises the uncertainty of user acceptance of IoT devices. The risks of unsecured devices are high since just one weak link in a network can give attackers access to hundreds or thousands of other devices in the network. Research has already demonstrated that smart meter data can be remotely manipulated, which on a large scale could cause power outages, cost utility companies billions of dollars in revenue, and, in the worst case, cause the entire power grid to collapse. With billions of devices connected to the Internet, verifying the trust and integrity of the data received from each IoT device is essential.

Software vulnerabilities

If IoT devices are not configured to receive updates, the embedded software will eventually become out of date, exposing devices to attack if they continue to be in service. Further, if a device manufacturer goes out of business or discontinues a particular

product line, those devices are likely to remain vulnerable for the rest of their lives since no software updates or technical support will be available. Likewise, C and C++ have been the language of choice for IoT software due to their low memory footprint, but many programmers who write open source software programs are unaware of the security vulnerabilities introduced by unsafe function calls. For example, `strcpy` and `strncpy` in C do not check buffer length, which could introduce a buffer overflow vulnerability in a program.[13] Leveraging such a vulnerability, an attacker can execute arbitrary code or perform other unauthorized functions to steal encryption keys and extract user credentials stored on an IoT device. In 2016, researchers discovered heap buffer overflow and buffer overhead vulnerabilities in the MatrixSSL X.509 certificate, an open source implementation of SSL for low-power embedded devices, which allowed remote attackers to execute arbitrary code on the system with root privileges.[14]

Malware targeting IoT devices

IoT medical devices are of particular interest to attackers because they are devoid of advanced security features and thus vulnerable to malware. In the US, medical devices must meet rigorous safety standards and go through a lengthy Food and Drug Administration (FDA) approval process prior to commercial release. Any changes on medical devices such as installation of patches, security updates, and third-party security software must be validated and approved by the device manufacturer. It typically takes longer for medical devices to go through the formal validation and release process than nonregulated IT systems. This leaves vulnerabilities in such devices even if they are installed in secure locations behind firewalls. According to cybersecurity firm TrapX, in 2015 hackers succeeded in placing a backdoor on medical devices at three hospitals that gave them open access to hospital data.[15]

Other types of IoT devices have been targeted by malware, most notably Mirai. In 2013, three years before the Dyn cyberattack, Symantec discovered the Linux.Darlloz worm, which exploited PHP vulnerabilities to infect IoT devices like TVs and routers.[16] In 2016, a researcher reported that a security camera became infected by Mirai-like malware within 98 seconds after connecting to a Wi-Fi network.[17]

> JUST ONE WEAK LINK IN A NETWORK CAN GIVE ATTACKERS ACCESS TO HUNDREDS OR THOUSANDS OF OTHER DEVICES IN THE NETWORK.

Insecure web interfaces

Insecure web interfaces to IoT devices are vulnerable to attacks such as account enumeration, brute-force login, or account lock. For example, attackers able to log in to a website using a brute-force approach might gain access to the administrative features and sensitive data. They could change the credentials of legitimate users, resulting in denial of access for those users, or in some cases completely take over the device. This vulnerability is omnipresent in IoT devices because most application developers assume that only trusted internal users have access to a device's web interface. However, threats from external users are just as potent. A recent dynamic analysis of firmware images from more than 50 vendors found that 24 percent had web interface vulnerabilities, including 225 high-impact vulnerabilities.[18]

Privacy issues

As more IoT devices are used to collect personal data, protecting end users' privacy will become a challenge. On 6 February 2017, Vizio, one of the world's largest manufacturers of smart TVs, agreed to pay $2.2 million in fines to the US Federal Trade Commission and the Office of the New Jersey Attorney General for tracking viewing data from 11 million TVs without consumers' knowledge or consent. In addition, as big data technologies mature, the use of deep data analytics might eventually connect all of the sparse and unstructured data on the Internet. This will reveal far more information about us than we have ever known.

Weakest security link

In conjunction with industry and academia, the US government is taking steps to strengthen and maintain

infrastructure sectors identified as vital to the nation's security, economy, and public health or safety, all of which rely at least in part on the Internet.[20] As with all types of security, IoT security is not measured by the most advanced techniques deployed but rather the weakest link. Even when a vulnerability is reported and a security patch is created for an IoT device, there is no guarantee that the update will be applied—whether through ignorance, inertia, or incompetence. Vulnerabilities can and will always be found and exploited by malicious users.

RISK MITIGATION

Despite the many challenges providing end-to-end trust and security for IoT applications, steps can be taken to mitigate risks.

Security policy

Many existing security principles, such as layering and limiting, can help mitigate the security risks of IoT applications. For example, assuming IoT devices can be breached, they can be separated into a different layer from the core network. If IoT devices must reside in the core network, network segmentation and a virtual LAN can be used to isolate the devices. In addition, limiting access to information should be enforced on IoT devices. Other security policies such as diversity, obscurity, and simplicity can also be adopted to manage IoT devices.

The DDoS attack against Dyn shows how important it is to enforce security policies on IoT devices in a network. Exploiting the fact that many users do not change devices' default credentials, Mirai tries 68 common default username–password pairs to take control of devices and form a botnet.[21]

Device identification and location tracking

Identifying and tracking the location of all IoT devices in a network is crucial to security.[1] Some IoT devices are deployed in insecure locations such as roadsides and bridges, where anyone with sufficient knowledge could alter its hardware to generate wrong data. Indoor positioning systems are particularly vulnerable to physical-layer attacks such as location spoofing: signals used to identify a device can be captured and altered to fake its location. Techniques that match a requested service to a device's identity and location will help counter such threats—for example, to detect a security camera improperly requesting access to a retail store's financial database.

Fog computing

Fog computing bridges the gap between the IoT and the cloud by bringing computing closer to the data source. Resource-constrained IoT devices have little or no ability to protect themselves from sophisticated cyberattacks. By adopting a fog computing architecture, IoT application developers can leverage a defense-in-depth strategy with nearby fog nodes acting as sentries. An upstream fog node with enough processing power can detect and filter malicious traffic before it can pass into the system. Eavesdropping on data is also extremely difficult, as attackers must be in close proximity to the fog nodes, which increases the likelihood of detection.[22] Similarly, storing data on secured fog nodes can offer better privacy protection than storing data on devices, which can be easily stolen due to their small size.

Many IoT devices utilize the cloud to store data and synchronize with one another. Because data breaches can occur anywhere en route to the cloud, ensuring end-to-end trust and security for IoT applications is critical. We developed a novel 3D model to identify threats to the IoT and discussed nine existing security challenges as well as practices to mitigate security risks. In future work, we plan to extend the model to get a more complete picture of where IoT attacks can occur.

REFERENCES

1. A. Al-Fuqaha et al., "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Comm. Surveys & Tutorials*, vol. 17, no. 4, 2015, pp. 2347–2376.
2. The White House, "Fact Sheet: Administration Announces an Advanced Wireless Research Initiative, Building on President's Legacy of Forward-Leaning Broadband Policy," 15 July 2016; https://obamawhitehouse.archives.gov/the-press-office/2016/07/15/fact-sheet-administration-announces-advanced-wireless-research.
3. J. Manyika et al., *The Internet of Things: Mapping the Value beyond the Hype*, McKinsey Global Institute, June 2015; www.mckinsey.com/business-functions/digital-mckinsey/our-insights/the-internet-of-things-the-value-of-digitizing-the-physical-world.
4. Cybersecurity Ventures, "Cybersecurity Ventures Projects $1 Trillion Will Be Spent Globally on Cybersecurity from 2017 to 2021," 2017; https://cybersecurityventures.com/cybersecurity-market-report.
5. S. Hilton, "Dyn Analysis Summary of Friday October 21 Attack," blog, 26

Oct. 2016; http://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack.
6. A.R. Sfar et al., "A Roadmap for Security Challenges in Internet of Things," *Digital Communications and Networks*, Apr. 2017; www.sciencedirect.com/science/article/pii/S2352864817300214.
7. M. Irshad, "A Systematic Review of Information Security Frameworks in the Internet of Things (IoT)," *Proc. 2016 IEEE 18th Int'l Conf. High Performance Computing and Communications; IEEE 14th Int'l Conf. Smart City; IEEE 2nd Int'l Conf. Data Science and Systems* (HPCC/SmartCity/DSS 16), 2016, pp. 1270–1275.
8. J. Singh et al., "Twenty Security Considerations for Cloud-Supported Internet of Things," *IEEE Internet of Things J.*, vol. 3, no. 3, 2016, pp. 269–284.
9. R. White, *Cyber Report 2016: UK Businesses Targeted 230,000 Times each by Cybercriminals*, Beaming, 2017; www.beaming.co.uk/cyber-reports/2016-year-cyber-attacks.
10. US Dept. of Homeland Security, *Strategic Principles for Securing the Internet of Things (IoT)*, v1.0, 15 Nov. 2016; www.dhs.gov/sites/default/files/publications/Strategic_Principles_for_Securing_the_Internet_of_Things-2016-1115-FINAL....pdf.
11. A. Ukil, J. Sen, and S. Koilakonda, "Embedded Security for Internet of Things," *Proc. 2nd IEEE Nat'l Conf. Emerging Trends and Applications in Computer Science*, 2011; doi:10.1109/NCETACS.2011.5751382.
12. M. Swanson and B. Guttman, *Generally Accepted Principles and Practices for Securing Information Technology Systems*, SP 800-14, NIST, Sept. 1996; http://csrc.nist.gov/publications/nistpubs/800-14/800-14.pdf.
13. S.M. Alnaeli et al., "Vulnerable C/C++ Code Usage in IoT Software Systems," *Proc. 3rd World Forum on Internet of Things* (WF-IoT 16), 2016, pp. 348–352.
14. E. Kovacs, "MatrixSSL Vulnerabilities Expose IoT Devices to Attacks," *SecurityWeek*, 11 Oct. 2016; www.securityweek.com/matrixssl-vulnerabilities-expose-iot-devices-attacks.
15. TrapX Research Labs, *MEDJACK.2: Hospitals under Siege*, 2016; https://trapx.com/trapx-labs-discovers-new-medical-hijack-attacks-targeting-hospital-devices-2.
16. K. Hayashi, "Linux Worm Targeting Hidden Devices," blog, 27 Nov. 2013; www.symantec.com/connect/blogs/linux-worm-targeting-hidden-devices
17. D. Coldewey, "This Security Camera Was Infected by Malware 98 Seconds after It Was Plugged in," TechCrunch, 18 Nov. 2016; https://beta.techcrunch.com/2016/11/18/this-security-camera-was-infecte-by-malware-in-98-seconds-after-it-was-plugged-in.
18. A. Costin, A. Zarras, and A. Francillon, "Automated Dynamic Firmware Analysis at Scale: A Case Study on Embedded Web Interfaces," *Proc. 11th ACM on Asia Conf. Computer and Communications Security* (ASIA CCS 16), 2016, pp. 437–448.
19. Federal Trade Commission, "VIZIO to Pay $2.2 Million to FTC, State of New Jersey to Settle Charges It Collected Viewing Histories on 11 Million Smart Televisions without Users' Consent," press release, 6 Feb. 2017; www.ftc.gov/news-events/press-releases/2017/02/vizio-pay-22-million-ftc-state-new-jersey-settle-charges-it.
20. The White House, "Presidential Policy Directive—Critical Infrastructure Security and Resilience," 12 Feb. 2013; https://obamawhitehouse.archives.gov/the-press-office/2013/02/12/presidential-policy-directive-critical-infrastructure-security-and-resil.
21. B. Krebs, "Who Makes the IoT Things under Attack," blog, 16 Oct. 2016; https://krebsonsecurity.com/2016/10/who-makes-the-iot-things-under-attack.
22. S. Yi, Z. Qin, and Q. Li, "Security and Privacy Issues of Fog Computing: A Survey," *Wireless Algorithms, Systems, and Applications*, LNCS 9204, K. Xu and H. Zhu, eds., Springer, 2015, pp. 685–695.

ABOUT THE AUTHORS

SULABH BHATTARAI is an instructor in the Beacom College of Computer and Cyber Sciences at Dakota State University. His research interests include cyber-physical systems, the IoT, and wireless network security. Bhattarai received an ScD in information technology from Towson State University and an MS in electrical engineering from Fairleigh Dickinson University. Contact him at sulabh.bhattarai@dsu.edu.

YONG WANG is an associate professor in the Beacom College of Computer and Cyber Sciences at Dakota State University. His research interests include wireless networks, the cloud, mobile devices, and related security and privacy issues. Wang received a PhD in computer science from the University of Nebraska–Lincoln. He is a member of IEEE and the IEEE Communications Society. Contact him at yong.wang@dsu.edu.

Game-Model-Based Network Security Risk Control

Zhen Ni and Qianmu Li, Nanjing University of Science and Technology
Gang Liu, Suzhou Vocational University

Network security decisions must consider how to balance information security risk and output in light of limited resources. Using game theory, the authors propose an optimum attack-defense decision-making algorithm that considers the interaction between attackers and defenders.

Active security defense technologies based on network security evaluations are becoming mainstream. Compared with traditional passive defense technologies, active defense technologies could enable users to identify system vulnerabilities and potential security threats in advance, and choose active security defense measures and strategies that are in-line with optimum cost effect based on security needs.[1] An ideal defense system should have defense responses for all vulnerabilities and attack behaviors. However, limited organizational resources means that it is not rational to defend at all costs. A balance between information security risk and output, and limited resources must be considered to make the most rational decisions about attack defense strategies.

The effectiveness of defense cost is an important factor for security administrators to consider. The essence of attack and defense confrontation in information security could be abstracted to the strategy dependency of the two parties: whether or not the defense strategy of a defender is effective depends not only on its own behaviors, but also on the strategies of the attacker and the defense system. In the present study, we used a game theory to analyze information security attack-defense confrontation problems such as the attack-defense contradiction and its optimum defense strategies.

Many static and passive security defense methods are available, such as firewalls, intrusion detection, and antivirus software. These methods always lack initiative and predictive ability toward attacks. In contrast, active

0018-9162/18/$33.00 © 2018 IEEE

RELATED WORK IN GAME THEORY MODELS FOR NETWORK SECURITY

Paul Syverson[4] proposed using the stochastic game for rational analyses on normal nodes and malicious nodes in networks. David Burke[5] proposed behavioral modeling of attackers and defenders in information wars using repeated games of incomplete information. Kong-wei Lye and Jeannette Wing[6] analyzed and inferred the attack-defense relationship in networks using the stochastic game form and verified the applicability of game theory to intrusion detection and response. Jun Xu and Wooyong Lee[7] designed and analyzed the distributed denial-of-service (DDoS) defense system based on the static game of complete information and optimized the system's performance.

Peng Liu and his colleagues[8] built an attack intent, objectives, and strategies (AIOS) attack-prediction model based on intrusion intentions using the repeated game and the stochastic game, and analyzed the application of AIOS in DDoS defense. Lawrence Carin and his colleagues[9] proposed a network security risk quantitative evaluation method for security policy efficiency analyses, and analyzed key infrastructure protection strategies using the attack-defense economic model and game theory. To solve the intrusion-detection problem of mobile ad-hoc networks, Yu Liu and colleagues[10] analyzed intrusions using the Bayes game theory and studied the existence of Bayesian Nash equilibrium. Hadi Otrok and his colleagues[11] proposed a cooperative game model to analyze and check interactive behaviors and reduce the false alarm rate. Wei Jiang and his colleagues[1] combined the principle of game theory with network attack and defense to build a defense map model, and used the model to analyze the possible attack behaviors of rational attackers, but the defense map model focused on network state transitions and thus could not fully reflect the attack paths.

A network security situation assessment method described the confrontation between attacker and defender as a two-party stochastic game problem.[9] Game parameters were determined using the administrator's assessment of the importance of network nodes. The distribution of probabilities about the network in different security states was obtained through the Nash equilibrium of the attack-defense game to further quantify assessment results.[10,11] Yuanzhuo Wang and his colleagues[12] proposed a stochastic game network model combined with stochastic Petri net to solve complicated dynamic game problems. Guanhua Yan and his colleagues[13] proposed a nonstandard game framework which deduced the possible state of system using Bayesian network and made modeling for multiple levels, then assessed complicated DDoS service attack-defense scenarios.

defense avoids some of the issues that exist in static defense. However, active defense methods have an obvious shortcoming: balance between information security risk and output, which requires making the most rational decision using limited resources. In recent years, game theory has been used in the network security field because the attacker and the defender have a significant game relationship. In this article, we describe and test the performance of an optimal attack-defense decision algorithm. For a discussion on previous work in this area, please see the sidebar.

NETWORK SECURITY GAME MODEL

An ideal defense system should have the ability to protect all vulnerable points from any attack. Despite that, the practical situation requires equilibrium between risk and input of information security. Both the attacker and the defender aim to gain maximum profit at minimum cost, thus the attack-defense cost must be considered. Below, we describe the formalized definition of network security game model (NSGM) combining with state attack-defense graph (SADG).[2]

Definition 1: NSGM

NSGM is a tetrad, NSGM = (P, S, T, U), where P = (PA, PD) represents the set of players participating in the attack-defense game, PA is the attacker, PD is the defender; S = {s0, s1, ... , sN} represents the set composed of network security states; $T = T_A^i, T_D^i), i = 0,1,2,\cdots,N)$ represents the set of strategies of each player, in which T_A^i is the set of strategies adopted by the attacker when the network is in security state s_i, that is, the set composed of all possible attack paths arriving at security state s_i, and T_D^i

is the set of strategies adopted by the defender when the network is in security state s_i, that is, the set composed of all defensive measures corresponding to possible attack paths arriving at security state s_i; $U = (U_A^i, U_D^i) i = 0,1,2,\cdots,N$ represents the utility function set of each player, in which U_A^i is the utility function of the attacker in security state s_i, and U_D^i is the utility function of the defender in security state s_i. The set of utility function can be represented as a matrix U∈Rm×n. For ∀si∈S, if we suppose that the strategy sets of attacker and defender are $T_A^i = (t_{A_1}^i, t_{A_2}^i \cdots t_{A_m}^i)$ and $T_D^i = (t_{D_1}^i, t_{D_2}^i \cdots t_{D_n}^i)$, respectively, then the utility functions of attacker and defender can be expressed in the matrix forms in Equations (1) and (2).

The meaning of the matrices is as follows: when the network is in security state s_i, suppose attack strategy set $T_A^i = (t_{A_1}^i, t_{A_2}^i \cdots t_{A_m}^i)$ and defense strategy $T_D^i = (t_{D_1}^i, t_{D_2}^i \cdots t_{D_n}^i)$. Use matrix $U_A^i = (a_{jk}^i)_{m\times n}$ to represent the utility matrix of the attacker, in which a_{jk}^i represents the utility value of the attacker in strategy combination $(t_{A_j}^i, t_{D_k}^i)$, that is, $a_{jk}^i = U_A(t_{A_j}^i, t_{D_k}^i)$; use matrix $U_D = (b_{jk}^i)_{m\times n}$ to represent the utility matrix of the defender in which b_{jk}^i represents the utility value of the defender in strategy combination $(t_{A_j}^i, t_{D_k}^i)$, that is, $b_{jk}^i = U_D(t_{A_j}^i, t_{D_k}^i)$, where i = 0, 1, 2, ..., N; j = 1, 2, ..., m; k = 1, 2, ..., n.

Combine the utility matrices of attacker and defender and express the utility of attacker and defender, $\mathbf{U}^i$, in the double-matrix form Equation (3).

Suppose τ is an attack, and utility(τA), profit(τA) and cost(τA) represent the attacker's utility value, profit, and cost after launching attack τ; utility(τD), profit(τD) and cost(τD) represent the defender's utility value, profit, and cost after defending against attack τ. The attacker's utility value utility(τA) after launching the attack equals the difference between profit(τA), the attacker's profit after launching the attack, and cost(τA), the cost to launch the attack, i.e., utility(τA) = profit(τA) – cost(τA). The defender's utility value utility(τD) defending against the attack equals the difference between profit(τD), the defender's profit after defending against the attack, and cost(τD), the cost to defend against the attack, that is, utility(τD) = profit(τD) – cost(τD). The damage caused by an attack to network system was used to represent profit(τA), the attacker's profit from launching the attack. The defender's profit after defending against the attack, profit(τD), is the damage from which the network system is protected after the defender uses the defense strategy against an attack, and the value of profit(τD) equals the damage caused by the attack to the network system, and thus profit(τA) = profit(τD). Cost(τA), the cost of the attacker to launch an attack, and cost(τD), the cost of the defender to defend against the attack, are affected by many factors, such as the difficulty level, type, and damage caused by attack. To simplify the calculation, this article only takes the factor of difficulty level of attack into consideration and uses the success probability of attack in SADG to represent the defender's defense cost, that is, cost(τD) = 10 × p. So, the higher the success probability is, the bigger the input is and the higher the defense cost is; the reciprocal of the attack's success probability represents the attacker's attack cost to launch the attack, that is, cost(τA) = 1/p, so the higher the success probability is, the easier the vulnerability of the network system will be used and the less the attack cost is.

Suppose $\{\tau_1,\tau_2,\cdots,\tau_n\}$ is an attack path and the corresponding defense path is $\{\tau_1,\tau_2,\cdots,\tau_m\}$, then the utility of attack path is defined as Equation (4), in which $\tau_i \notin \{\tau_1, \tau_{2,\ldots,}\tau_n\}$ means $\{\tau_1, \tau_{2,\ldots,}\tau_n\}$ doesn't include the attack using the same vulnerable point of τ_i, and $\tau_j \in \{\tau_1, \tau_{2,\ldots,}\tau_m\}$ means $\{\tau_1, \tau_{2,\ldots,}\tau_m\}$ includes the attack using the same vulnerable point of τ_j.

The utility of defense path is Equation (5), in which $\tau_j \notin \{\tau_1,\tau_{2,\cdots,}\tau_m\}$ means $\{\tau_1,\tau_{2,\cdots,}\tau_m\}$ does not include the attack using the same vulnerable point of τ_j, and $\tau_i \in \{\tau_1,\tau_{2,\cdots,}\tau_n\}$ means $\{\tau_1,\tau_{2,\ldots,}\tau_n\}$ includes the attack using the same vulnerable point of τ_i.

Definition 2: Nash equilibrium

In security state s_i, the attack-defense strategy is the Nash equilibrium for $(t_A^{i*}, t_D^{i*}), t_A^{i*} \in T_A^i, t_D^{i*} \in T_D^i$. If, and only if, for each player, $t_p^{i*}(p \in \{A, D\})$ is the optimal strategy to the other player: for $\forall t_A^i \in T_A^i, U_A^i(t_A^{i*}, t_D^{i*}) \geq U_A^i(t_A^i, t_D^{i*})$; for $\forall t_D^i \in T_D^i, U_D^i(t_A^{i*}, t_D^{i*}) \geq U_D^i(t_A^{i*}, t_D^i)$. Nash equilibrium refers to a strategy combination composed of the optimal strategies of all players. A player cannot change current strategies or get higher profit by changing his or her strategy unilaterally. The specific strategy chosen solely is called the pure strategy. Due to the uncertainty about the attacker's and the defender's behaviors, the pure-strategy Nash equilibrium might not exist, in which case it is necessary to consider the mixed strategy of attacker and defender.

Definition 3: Mixed strategy

In a given NSGM, in secure state s_i, the portability distributions of the

$$U_A^i = \begin{pmatrix} & t_{D_1}^i & t_{D_2}^i & \cdots & t_{D_n}^i \\ t_{A_1}^i & U_A(t_{A_1}^i, t_{D_1}^i) & U_A(t_{A_1}^i, t_{D_2}^i) & \cdots & U_A(t_{A_1}^i, t_{D_n}^i) \\ t_{A_2}^i & U_A(t_{A_2}^i, t_{D_1}^i) & U_A(t_{A_2}^i, t_{D_2}^i) & \cdots & U_A(t_{A_2}^i, t_{D_n}^i) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{A_m}^i & U_A(t_{A_m}^i, t_{D_1}^i) & U_A(t_{A_m}^i, t_{D_2}^i) & \cdots & U_A(t_{A_m}^i, t_{D_n}^i) \end{pmatrix} \quad (1)$$

$$U_A^i = \begin{pmatrix} & t_{D_1}^i & t_{D_2}^i & \cdots & t_{D_n}^i \\ t_{A_1}^i & U_D(t_{A_1}^i, t_{D_1}^i) & U_D(t_{A_1}^i, t_{D_2}^i) & \cdots & U_D(t_{A_1}^i, t_{D_n}^i) \\ t_{A_2}^i & U_D(t_{A_2}^i, t_{D_1}^i) & U_D(t_{A_2}^i, t_{D_2}^i) & \cdots & U_D(t_{A_2}^i, t_{D_n}^i) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{A_m}^i & U_D(t_{A_m}^i, t_{D_1}^i) & U_D(t_{A_m}^i, t_{D_2}^i) & \cdots & U_D(t_{A_m}^i, t_{D_n}^i) \end{pmatrix} \quad (2)$$

$$\begin{pmatrix} & t_{D_1}^i & t_{D_2}^i & \cdots & t_{D_n}^i \\ t_{A_1}^i & (U_A(t_{A_1}^i, t_{D_1}^i), U_D(t_{A_1}^i, t_{D_1}^i)) & (U_A(t_{A_1}^i, t_{D_2}^i), U_D(t_{A_1}^i, t_{D_2}^i)) & \cdots & (U_A(t_{A_1}^i, t_{D_n}^i), U_D(t_{A_1}^i, t_{D_n}^i)) \\ t_{A_1}^i & (U_A(t_{A_2}^i, t_{D_1}^i), (U_D(t_{A_2}^i, t_{D_1}^i)) & (U_A(t_{A_2}^i, t_{D_2}^i), U_D(t_{A_2}^i, t_{D_2}^i)) & \cdots & (U_A(t_{A_2}^i, t_{D_n}^i), U_D(t_{A_2}^i, t_{D_n}^i)) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{A_m}^i & (U_A(t_{A_m}^i, t_{D_1}^i), U_D(t_{A_m}^i, t_{D_1}^i)) & (U_A(t_{A_m}^i, t_{D_2}^i), U_D(t_{A_m}^i, t_{D_2}^i)) & \cdots & (U_A(t_{A_m}^i, t_{D_n}^i), U_D(t_{A_m}^i, t_{D_n}^i)) \end{pmatrix} \quad (3)$$

$$utility(\{\tau_1, \tau_2, \cdots, \tau_n\}) = \sum_{\substack{i=1 \\ \tau_1 \notin \{\tau_1, \tau_2, \cdots, \tau_m\}}}^{n} utility(\tau_i) - \sum_{\tau_j \in \{\tau_1, \tau_2, \cdots, \tau_m\}} \cos t(\tau_j) \quad (4)$$

$$utility(\{\tau_1, \tau_2, \cdots, \tau_m\}) = \sum_{\substack{i=1 \\ \tau_1 \in \{\tau_1, \tau_2, \cdots, \tau_n\}}}^{m} utility(\tau_i) - \sum_{\tau_j \notin \{\tau_1, \tau_2, \cdots, \tau_m\}} \cos t(\tau_j) \quad (5)$$

$$V_A^i(p_A^i, p_D^i) = \sum_j^m p_{A_j}^i \left[\sum_k^n p_{D_k}^i U_A^i(t_{A_j}^i, t_{D_k}^i,) \right] = \sum_j^m \sum_k^n p_{A_j}^i \bullet p_{D_k}^i U_A^i(t_{A_j}^i, t_{D_k}^i,) \quad (6)$$

$$V_D^i(p_A^i, p_D^i) = \sum_k^n p_{D_k}^i \left[\sum_j^m p_{A_j}^i U_D^i(t_{A_j}^i, t_{D_k}^i,) \right] = \sum_k^n \sum_j^m p_{A_j}^i \bullet p_{D_k}^i U_D^i(t_{A_j}^i, t_{D_k}^i,) \quad (7)$$

attacker's strategy $T_A^i = (t_{A_1}^i, t_{A_2}^i, \ldots, t_{A_m}^i)$ and the defender's strategy $T_D^i = (t_{D_1}^i, t_{D_2}^i, \ldots, t_{D_n}^i)$ are $p_A^i = (p_{A_1}^i, p_{A_2}^i, \ldots, p_{A_m}^i)$ and $p_D^i = (p_{D_1}^i, p_{D_2}^i, \ldots, p_{D_n}^i)$ respectively, with

$$0 \le p_{A_j}^i \le 1, 0 \le p_{D_k}^i \le 1, \sum_{j=1}^m p_{A_j}^i = 1, \sum_j^n p_{D_k}^i = 1,$$

where $i = 0, 1, 2, \ldots, N$; $j = 1, 2, \ldots, m$; $k = 1, 2, \ldots, n$. The attacker and the defender choose attack and defense strategies in the form of probability. The pure strategy is an exception to the mixed strategy.

The expected utilities of attacker and defender are Equations (6) and (7), respectively.

Mixed strategy (p_A^{i*}, p_D^{i*}) is the Nash equilibrium if the mixed strategy is the optimal mixed strategy for attacker and defender, that is, it satisfies: for $\forall p_A^i, V_A^i(p_A^{i*}, p_D^{i*}) \ge V_A^i(p_A^i, p_D^{i*})$, while for $\forall p_D^i, V_D^i(p_A^{i*}, p_D^{i*}) \ge V_D^i(p_A^{i*}, p_D^i)$.

ALGORITHM FOR OPTIMAL ATTACK-DEFENSE STRATEGY

In this section, we describe our proposed algorithm for optimal attack-defense strategy.

Theorem 1: Existence of Nash equilibrium

Each finite strategic game has at least Nash equilibrium, pure-strategy, or mixed-strategy Nash equilibrium. Nash

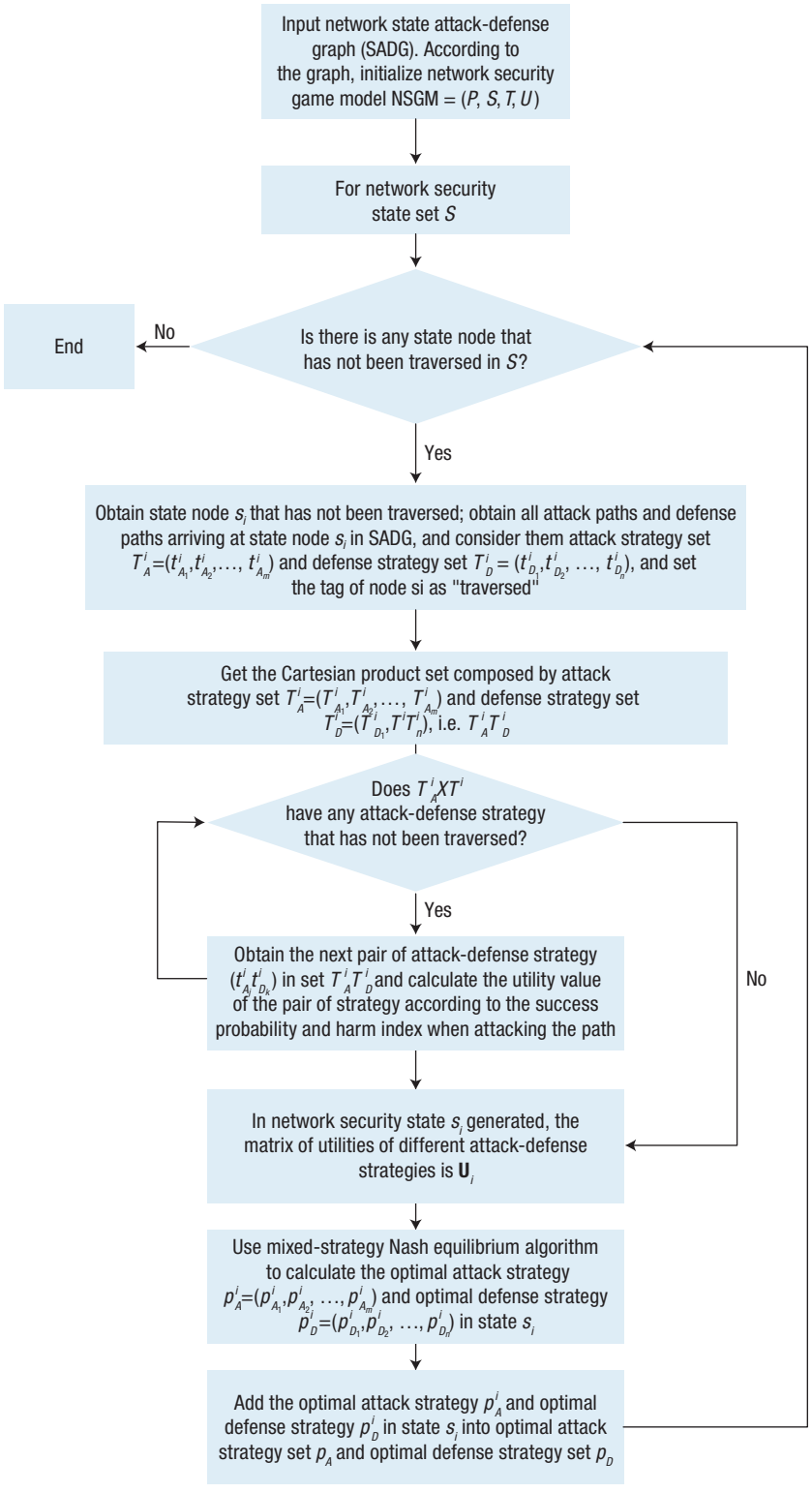


FIGURE 1. Network security optimal attack-defense decision algorithm.

proved theorem 1 using the Brouwer fixed-point theorem.[3] The network security attack-defense game is a finite strategic game, so we know, according to theorem 1, that there must be a solution of an optimal attack-defense decision.

SADG = (S, T, s0, SG) describes the attack and defense behaviors of a network, and is associated with NSGM = (P, S, T, U) through the network's security state set S. In SADG, the attacker can arrive at the same security state through different attack paths by aiming at different attack paths; thus, the defender must develop different security measures. If we consider different attack paths as the attacker's strategy set and the corresponding protective measures as the defender's strategy set, and then calculate the harm indices and success probabilities of different attack paths to the utility matrices of attacker and defender, it is possible to transform the attack-defense state map into an NSGM and then generate the optimal attack-defense strategies of attacker and defender in different network security states by solving and analyzing the NSGM.

On the basis of this idea, we detail below a proposed network security optimal attack-defense decision algorithm based on NSGM.

Input: SADG

Output: Optimal attack and defense strategies PA and PD

1. According to SADG, NSGM = (P, S, T, U)
2. For each network security state s_i
3. According to attack path arrived at s_i, construct attack strategy set TiA = (tiA1, tiA2, ..., tiAm) and defense strategy set TiD = (tiD1, tiD2, ..., tiDn)
4. For each pair of strategy (tiAj,tiDk)∈TiA×TiD
5. According to success probability of attack path and harm index, calculate the attacker's and defender's utility value under network security state s_i
6. End for each
7. According to the utility matrix

Ui call Mixed Strategy Nash Equilibrium algorithm, and calculate the optimal attack strategy piA = (piA1, piA2, ..., piAm) and the optimal defense strategy piD = (piD1, piD2, ..., piDn) in the network security state s_i

8. Add piA to pA, and add piD to pD
9. End for each
10. Return pA and pD

Now, we describe how to get the mixed-strategy Nash equilibrium through nonlinear programming.

Suppose the probability distributions of attacker and defender are $V_D^i(p_A^i, p_D^i) - p_A^i \mathbf{U}_D^i (p_D^i)^T$ and $p_D^i = (p_{D_1}^i, p_{D_2}^i, \cdots p_{D_n}^i)$ respectively, then $V_A^i(p_A^i, p_D^i) - p_A^i \mathbf{U}_A^i (p_D^i)^T$ and $V_D^i(p_A^i, p_D^i) - p_A^i \mathbf{U}_D^i (p_D^i)^T$. A two-player finite mixed-strategy game's Nash equilibrium can be obtained by solving the following nonlinear programming problem:

$$
\begin{aligned}
&\text{Max } z = p_A^i \mathbf{U}_A^i (p_D^i)^T - V_A^i \\
&+ p_A^i \mathbf{U}_D^i (p_D^i)^T - V_D^i, \\
&\text{s.t. } \mathbf{U}_A^i (p_D^i)^T \le V_A^i \mathbf{I}_m^T, (p_A^i \mathbf{U}_D^i)^T \\
&\le V_D^i \mathbf{I}_n^T, p_A^i \mathbf{I}_m^T = p_D^i \mathbf{I}_n^T = 1, p_{A_j}^i \\
&\ge 0, j = 1, 2, \cdots m \\
&p_{A_k}^i \ge 0, k = 1, 2, \cdots, n.
\end{aligned}
$$

$\mathbf{I}_m^i$ and $\mathbf{I}_n^i$ represent unit row vectors $(1,1,\cdots,1)_{1\times m}$ and $(1,1,\cdots,1)_{1\times n}$, respectively, and V_A^i V_D^i represent the expected utilities of attacker and defender in Nash equilibrium.

In the programming problem above, constraint conditions come from Definition 4. In the problem, $\mathbf{U}_A^i (p_D^i)^T \le V_A^i \mathbf{I}_m^T$ means that in the equilibrium situation, the expected utility obtained by the attacker by adopting any pure strategy is not more than equilibrium utility V_A^i, that is, $\forall t_{A_j}^i \in T_A^i, V_A^i(t_{A_j}^i, p_D^i) \le V_A^i$; $(p_A^i \mathbf{U}_D^i)^T \le V_D^i \mathbf{I}_n^T$ means that in the equilibrium state, the expected utility obtained by the defender by adopting any pure strategy is not more than equilibrium utility V_D^i, that is, $\forall t_{D_k}^i \in T_D^i, V_D^i(t_{D_k}^i, p_A^i) \le V_D^i$. In the objective function, $p_A^i \mathbf{U}_A^i (p_D^i)^T - V_A^i$ and $p_A^i \mathbf{U}_D^i (p_D^i)^T - V_D^i$ mean that if the attacker and the defender do not choose the equilibrium strategy, the objective function will never arrive at the optimal. Figure 1 shows the network security optimal attack-defense decision algorithm flow.

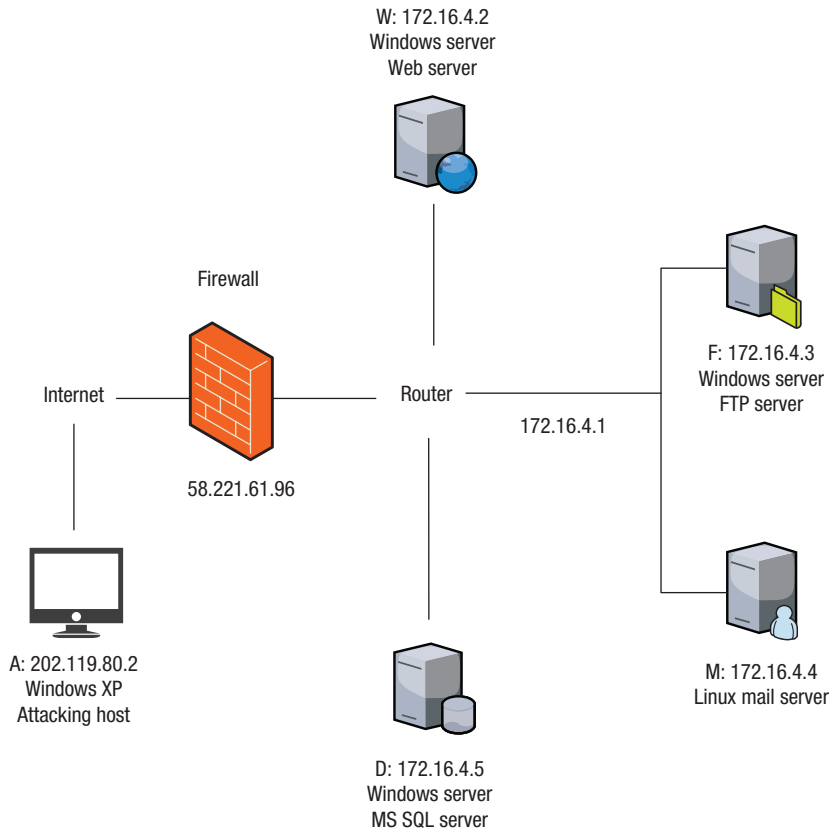


FIGURE 2. Network topological graph.

EXPERIMENT AND ANALYSIS

The present study uses a network attack scenario for a simulation experiment to verify the effectiveness of the network security game mode and optimal attack-defense strategy generation algorithm. Figure 2 shows the topological structure of the experimental network. The attacking host is in an external network, and the target network is a switched network. The internal and external networks are separated by a firewall. In the internal network, there is a public Web server, an FTP server, an MS SQL database server, and a mail server, which are represented by W, F, D, and M, respectively.

The network firewall only allows external hosts to access the services of the Web server with the permissions of normal users, and prevents any other external access. In the internal network, server nodes can access each other with the permissions of normal users. The vulnerable points in the experiment are all permission-escalation vulnerable points. The attacker has root permission in the attacking host and launches attacks with the objective of elevating the access permissions in server nodes in the internal network. According to the reliability vector orthogonal

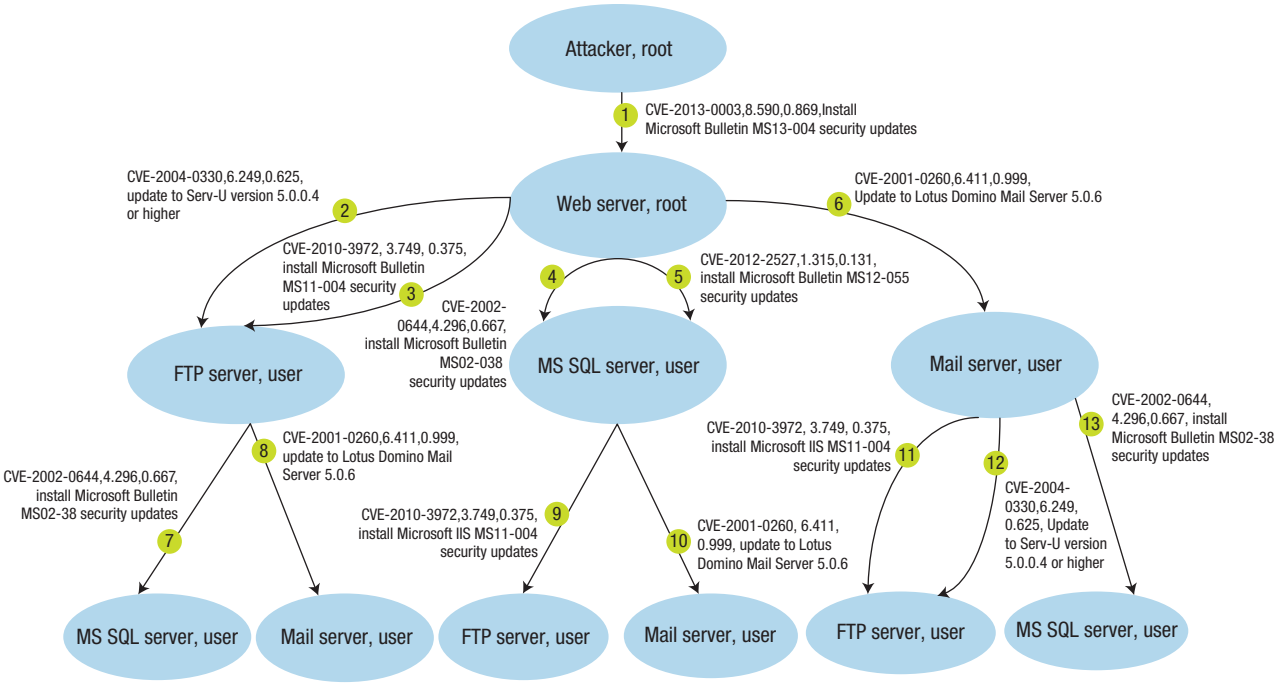


FIGURE 3. State attack–defense graph.

projection decomposition algorithm, the credibility of vulnerable point CVE-2013-0003 in the Web server is 1; the credibility of vulnerable points CVE-2004-0330 and CVE-2010-3972 in the FTP server is 0.625 and 0.375, respectively; the credibility of vulnerable points CVE-2002-0644 and CVE-2012-2527 in the MS SQL server is 0.667 and 0.333, respectively; and the credibility of vulnerable point CVE-2001-0260 in the mail server is 1. Initially, the attacker has user permissions to access the Web server and no permission to access other servers. The attacker uses the vulnerable points of the host nodes (mentioned above) to launch remote attacks following the attack rules in Table 1 to elevate permissions in server nodes. Figure 3 shows the SADG generated in the experimental network. Table 2 shows the cost, profit, and utility value of the attacker and the defender in each attack.

TABLE 1. Rules for using vulnerable points in an attack.

	Precondition				
Attack	**src_priv**	**dst_priv**	**vul_id**	**connection**	**Postcondition**
Microsoft .NET Framework remote permission escalation	Root	User	CVE-2013-0003	A→W	Root
Serv-U FTP server MDTM order remote buffer overflow	User	None	CVE-2004-0330	W→F M→F	User
IIS FTP server buffer overflow	User	None	CVE-2010-3972	W→F D→F M→F	User
Microsoft SQL Server Database buffer overflow	User	None	CVE-2002-0644	W→D F→D M→D	User
Microsoft Windows local permission escalation	User	None	CVE-2012-2527	W→D	User
Lotus Domino mail server strategy buffer overflow	User	None	CVE-2001-0260	W→M F→M D→M	User

A is attacker, D is MS SQL database server, F is FTP server, M is mailer server, and W is Web server.

Traditional attack-defense strategies

The specific security state of the network system is arrived at through a series of transformations, that is, by implementing a series of attacks. Figure 3 shows the attack path and the implementation steps of attack. Only when each attack in the attack path is implemented successfully is the whole attack path successful. According

TABLE 2. Cost, profit, and utility value of attacker and defender in each attack.

Vulnerable point used in attack	Attack	Attacker			Defender		
		Cost	Profit	Utility	Cost	Profit	Utility
CVE-2013-0003	Microsoft .NET Framework remote permission escalation	1.151	8.590	7.439	8.690	8.590	–0.100
CVE-2004-0330	Serv-U FTP server MDTM order remote buffer overflow	1.600	6.249	4.649	6.250	6.249	–0.001
CVE-2010-3972	IIS FTP server buffer overflow	2.667	3.749	1.082	3.750	3.749	–0.001
CVE-2002-0644	Microsoft SQL Server Database buffer overflow	1.499	4.296	2.797	6.670	4.296	–2.374
CVE-2012-2527	Microsoft Windows local permission escalation	7.634	1.315	–6.319	1.310	1.315	0.005
CVE-2001-0260	Lotus Domino mail server strategy buffer overflow	1.001	6.441	5.440	9.990	6.441	–3.549

to the multiplication principle of probability, the success probability of the attack path is the product of success probabilities of all attacks; each attack produces certain harm to the network system, so the harm index of the entire attack path is the sum of harm indices of all attacks. Therefore, starting from the initial security state of network, multiply the success probabilities of all attacks comprising the attack path and then get the success probability of the attack path; add the harm indices of all attacks comprising the attack path, and then get the harm index of the attack path.

Optimal attack and defense strategies

Traditional attack and defense strategies are generally made according to the difficulty of implementation and harm degree, and both attacker and defender pay more attention to the path attacked most easily and the path with the biggest harm index. However, the strategy plan that starts from the perspective of a player alone without considering the cost of attacker and defender is not rational. The following section uses the example of an FTP server node to describe the optimal attack-defense strategy-generation algorithm based on network security game mode, which is a strategy-generation method for both attacker and defense starting from the attacker and arriving at network security state (FTP server, user). A total of six attack paths from network security state (attacker, root) to network security state (FTP server, user) were used (Table 3).

The optimal attack-defense strategy generation algorithm and the mixed-game Nash equilibrium computing method were used and combined with the attack and defense utility matrices for network security state (FTP server, user) to get a mixed-strategy Nash equilibrium ((0,0,0,0,0,1), (0.7317,0.2683,0,0,0,0)). The optimal attack strategy of the attacker is to choose attack strategy 1→6→12; the optimal defense strategy of the defender is to choose path 1→2 for defense with a probability of 0.7317 and to choose path 1→3 for defense with a probability of 0.2683. According to the expected utility calculation formula of attacker and defender, the expected utility of the attacker is 4.366 and the expected utility of the defender is –1.107.

A pure-strategy Nash equilibrium ((0,0,0,0,1), (0,1,0,0,0)) can be obtained accordingly for attacker and defender in network security state (MS SQL server, user; Table 3). The optimal attack strategy of the attacker is to choose attack strategy 1→6→13, while the optimal defense strategy of the defender is to choose path 1→5 for defense. According to the expected utility calculation formula of attacker and defender, the expected utility of the attacker is 7.086 and the expected utility of the defender is –1.41.

A pure-strategy Nash equilibrium ((0,1,0,0,0),(1,0,0,0,0)) can be obtained accordingly for attacker and defender in network security state (mail server, user; Table 3). The optimal attack strategy of the attacker is to choose attack strategy 1→2→8; the optimal defense strategy of the defender is to choose path 1→6 for defense. According to the expected utility calculation formula of attacker and defender, the expected utility of the attacker is 2.497 and the expected utility of the defender is –3.649. Table 4 shows the optimal attack and defense strategies according to the calculation and analysis above; meanwhile, cost and profit of both attacker and defender and maximum reinforced network security were taken into consideration.

According to the comparison with the traditional way of generating attack and defense strategies, Figure 4 shows the utility curves of attacker and defender when adopting the strategy implemented most easily, the strategy with the biggest harm index, and the optimal cost–profit strategy. When using the optimal decision method, the attacker gets the biggest utility in all three network security states; the defender gets the biggest utility in states (MS SQL server, user) and (mail server, user), but in state (FTP server, user), the defender's utility is smaller than that of the strategy implemented most easily (Figure 4). The reason is that in that state, both attacker and defender get a mixed-strategy Nash

TABLE 3. Attack and defense strategies.

		Defender						
FTP server, user		Strategy	1→2	1→3	1→4→9	1→5→9	1→6→11	1→6→12
Attacker	Strategy	Mixed strategy	y1	y2	y3	y4	y5	y6
	1→2	x1	–2.751/–0.101	3.498/–3.850	3.498/–10.520	3.498/–5.160	3.498/–13.840	–2.751/–10.091
	1→3	x2	–0.069/–6.350	–3.818/–0.101	–3.818/–6.771	–3.818/–1.411	–3.818/–10.0910	–0.069/–16.340
	1→4→9	x3	2.728/–6.350	–1.021/–0.101	–5.317/–2.475	–1.021/–1.411	–1.021/–10.091	2.728/–16.340
	1→5→9	x4	–6.388/–6.350	–9.070/–0.101	–9.070/–6.771	–11.452/–0.096	–10.137/–10.091	–6.388/–16.340
	1→6→11	x5	5.371/–3.650	1.622/–0.101	1.622/–6.771	1.622/–1.411	–4.819/–3.650	–1.070/–9.899
	1→6→12	x6	2.689/–0.101	8.938/–3.850	8.938/–10.520	8.938/–5.160	2.497/–7.399	–3.752/–3.650

		Defender					
MS SQL server, user		Strategy	1→4	1→5	1→2→7	1→3→7	1→6→13
Attacker	Strategy	Mixed strategy	y1	y2	y3	y4	y5
	1→4	x1	–2.650/–2.474	1.646/–1.410	–2.650/–8.724	–2.650/–6.224	–2.650/–16.760
	1→5	x2	–7.470/–6.770	–8.785/–0.095	–7.470/–13.020	–7.470/–10.520	–7.470/–16.760
	1→2→7	x3	1.999/–2.474	6.295/–1.410	–4.250/–2.475	1.999/–6.224	1.999/–12.464
	1→3→7	x4	–1.568/–2.474	2.728/–1.410	–1.568/–8.724	–5.317/–2.475	–1.568/–12.464
	1→6→13	x5	2.790/–2.474	7.086/–1.410	2.790/–8.724	2.790/–6.224	–3.651/–6.023

		Defender					
Mail Server, User		Strategy	1→6	1→2→8	1→3→8	1→4→10	1→5→10
Attacker	Strategy	Mixed strategy	y1	y2	y3	y4	y5
	1→6	x1	–2.152/–3.649	–2.152/–9.899	–2.152/–7.399	–2.152/–10.319	–2.152/–4.959
	1→2→8	x2	2.497/–3.649	–3.752/–3.650	2.497/–7.399	2.497/–10.319	2.497/–4.959
	1→3→8	x3	–1.070/–3.649	–1.070/–9.899	–4.819/–3.650	–1.070/–10.319	–1.070/–4.959
	1→4→10	x4	0.645/–3.649	0.645/–9.899	0.645/–7.399	–3.651/–6.023	0.645/–4.959
	1→5→10	x5	–8.471/–3.649	–8.471/–9.899	–8.471/–7.399	–8.471/–10.319	–9.786/–3.644

equilibrium with the biggest expected utility. If the defender adopts defense strategy 1→2 for the pursuit of maximum utility unilaterally, the attacker will change the attack strategy, for example, adopting 1→6→11, to realize a bigger utility; the defender then gets the smallest utility.

Taking attack-and-defense cost and profit into consideration when making attack and defense strategies can help attackers and defenders find an equilibrium between cost and profit. In addition, blind input of a network administrator in network security risk control can be avoided and network security risk minimized using limited resources and capabilities.

Game theory was introduced in the field of network security risk control to avoid the blind and excessive output of network security risk control. We studied the complexity of choosing the optimum attack-defense strategy of network security in risk control. During network security risk control,

TABLE 4. Optimal attack and defense decisions.

Network security state	Optimal attack strategy	Optimal defense strategy
(Web server, root)	Launch Microsoft .NET Framework remote permission escalation attack to Web server using vulnerable point CVE-2013-0003.	Install Microsoft Bulletin MS13-004 security updates in Web server aiming at vulnerable point CVE-2013-0003.
(FTP server, user)	1. Launch Microsoft .NET Framework remote permission escalation attack to Web server using vulnerable point CVE-2013-0003. 2. Launch strategy buffer overflow attack to Lotus Domino mail server in mail server using vulnerable point CVE-2001-0260. 3. Launch MDTM order remote buffer overflow attack to Serv-U FTP service in FIP server using vulnerable point CVE-2004-0330.	1. Install Microsoft Bulletin MS13-004 security updates in Web server aiming at vulnerable point CVE-2013-0003. 2. Upgrade Serv-U FTP software in FTP server to Serv-U 5.0.0.4 or a higher version with a probability of 0.7317, or install Microsoft IIS MS11-004 security updates in FTP server aiming at vulnerable point CVE-2010-3972 with a probability of 0.2683.
(MS SQL server, user)	1. Launch Microsoft .NET Framework remote permission escalation attack to Web server using vulnerable point CVE-2013-0003. 2. Launch strategy buffer overflow attack to Lotus Domino mail server in Mail server using vulnerable point CVE-2001-0260. 3. Launch buffer overflow attack to SQL Server Database in MS SQL server using vulnerable point CVE-2002-0644.	1. Install Microsoft Bulletin MS13-004 security updates in Web server aiming at vulnerable point CVE-2013-0003. 2. Install Bulletin MS12-055 security updates in MS SQL server aiming at vulnerable point CVE-2012-2527.
(Mail server, user)	1. Launch Microsoft .NET Framework remote permission escalation attack to Web server using vulnerable point CVE-2013-0003. 2. Launch MDTM order remote buffer overflow attack to Serv-U FTP service in FIP server using vulnerable point CVE-2004-0330. 3. Launch strategy buffer overflow attack to Lotus Domino mail server in mail server using vulnerable point CVE-2001-0260.	1. Install Microsoft Bulletin MS13-004 security updates in Web server aiming at vulnerable point CVE-2013-0003. 2. Upgrade Lotus Domino mail server suite in Mail server to Lotus Domino Mail Server 5.0.6 or a higher version.

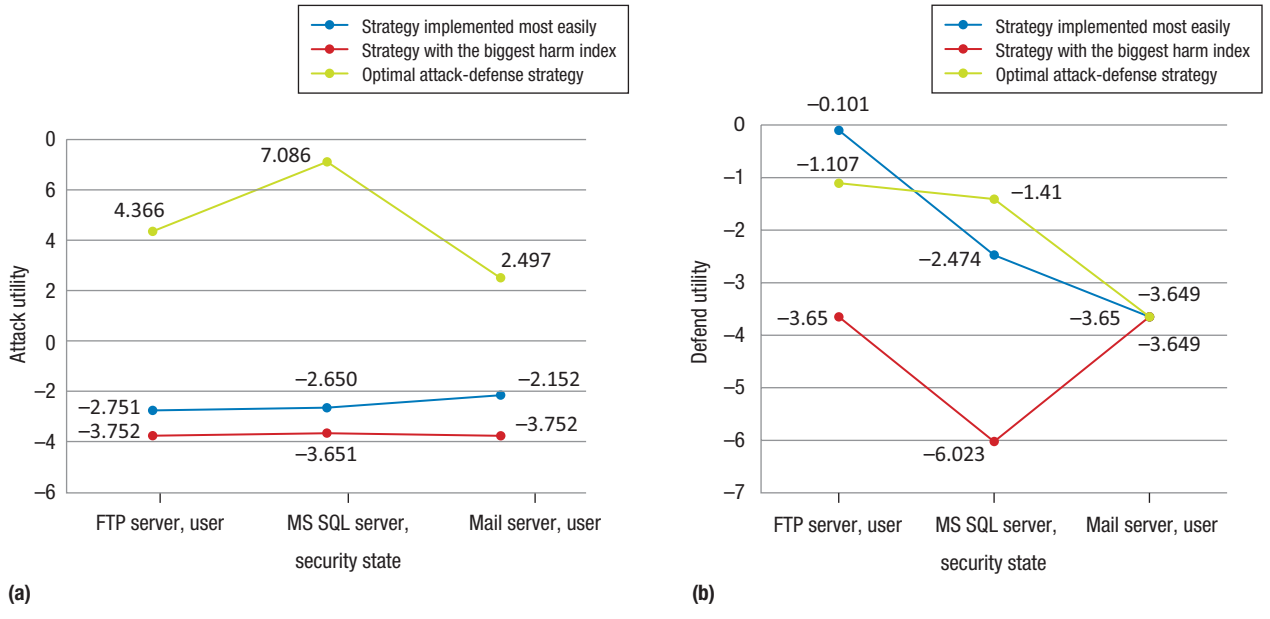


FIGURE 4. Utility of (a) attacker and (b) defender in different security states.

quantization of the costs and benefits for the attack and defense will greatly impact the final result of problem solving. Therefore, future studies should establish a cost/benefit index system and improve the quantization model.

ACKNOWLEDGMENTS

This work was supported by the Fundamental Research Funds for the Central Universities (no. 30916015104), the National Natural Science Foundation of China (no. 61272419), and the Perspective Project of Jiangsu Province (no. BY2014089).

REFERENCES

1. W. Jiang et al., "Evaluating Network Security and Optimal Active Defense based on Attack-Defense Game Model," *Chinese J. Computers*, vol. 32, no. 4, 2009, pp. 817–827.
2. G. Liu, Q. Li, and H. Zhang, "Defense Strategy Generation Method for Network Security based on State Attack-Defense Graph," *J. Computer Applications*, 2013, pp. S1.
3. J. Nash, *Non-cooperative Games*, Annals of Mathematics, 1951, pp. 286–295.
4. P.F. Syverson, "A Different Look at Secure Distributed Computation," *Proc. 10th Computer Security Foundations Workshop*, 1997; https://doi.org/10.1109/CSFW.1997.596797.
5. D.A. Burke, *Towards a Game Theory Model of Information Warfare*, master's thesis, Air Force Institute of Technology, DTIC document AFIT/GSS/LAL/99D-1, 1999; http://www.dtic.mil/dtic/tr/fulltext/u2/a374162.pdf.
6. K. Lye and J.M. Wing, "Game Strategies in Network Security," *Int'l J. Information Security*, vol. 4, nos. 1–2, 2005, pp. 71–86.
7. J. Xu and W. Lee, "Sustaining Availability of Web Services under Distributed Denial of Service Attacks," *IEEE Trans. Computers*, vol. 52, no. 2, 2003, pp. 195–208.
8. P. Liu, W. Zang, and M. Yu, "Incentive-based Modeling and Inference of Attacker Intent, Objectives, and Strategies," *ACM Trans. Information and System Security*, vol. 8, no. 1, 2005, pp. 78–118.
9. L. Carin, G. Cybenko, and J. Hughes, "Cybersecurity Strategies: The QUERIES Methodology," *Computer*, vol. 41, no. 8, 2008, pp. 20–26.
10. Y. Liu, C. Comaniciu, and H. Man, "A Bayesian Game Approach for Intrusion Detection in Wireless Ad Hoc Networks," *Proc. Workshop on Game Theory for Communications and Networks* (GameNets 06), 2006, article no. 4; https://doi.org/10.1145/1190195.1190198.
11. H. Otrok et al., "A Game-Theoretic Intrusion Detection Model for Mobile Ad Hoc Networks," *Computer Comm.*, vol. 31, no. 4, 2008, pp. 708–721.
12. Y. Wang et al., "Stochastic Game Net and Applications in Security Analysis for Enterprise Network," *Int'l J. Information Security*, vol. 11, no. 1, 2012, pp. 41–52.
13. G. Yan et al., "Towards a Bayesian Network Game Framework for Evaluating DDOS Attacks and Defense," *Proc. ACM Conf. Computer and Communications Security* (CCS 12), 2012, pp. 553–566.

ABOUT THE AUTHORS

ZHEN NI is a PhD candidate in the School of Computer Science and Engineering, at Nanjing University of Science and Technology. His research interests include software engineering, data mining, and information security. Contact him at nizhen0523@189.cn.

QIANMU LI is a professor in the School of Computer Science and Engineering, at Nanjing University of Science and Technology. His research interests include software engineering and information security. Li received a PhD in computer application from Nanjing University of Science and Technology. Contact him at liqianmu@126.com.

GANG LIU is an assistant professor in the School of Computer Engineering, at Suzhou Vocational University. His research interests include software engineering and information security. Liu received a PhD in computer applications and technology from Suzhou Vocational University. Contact him at 280986899@qq.com.

Detecting Code Reuse Attacks with Branch Prediction

Yongsuk Lee and Gyungho Lee, Korea University

Code reuse attacks (CRAs) allow attackers to produce malicious results by using legitimate code binaries in memory. The authors propose incorporating control-flow validation into the processor's instruction execution pipeline, along with a mis-prediction validation unit and a branch prediction unit, to help identify attacks.

Computers are exposed to constant threats from software attacks. Researchers have explored many different ways for ensuring trustworthy software behavior, yet attacks continue to become more sophisticated and often outpace the protections. There have been numerous tactics that successfully block compromised program control-flows based on specific attributes of the attack itself; however, these have led attackers to launch new, more sophisticated types of attacks, which bypass the protections.

A code injection attack attempts to inject malicious code into memory.[1] This strategy is less often used due to new, effective protection mechanisms such as DEP (data execution prevention), which prevents code injected into the data area from starting its execution. Attackers have responded to this advance in protection methods with code reuse attacks (CRAs). CRAs can perform malicious actions by reusing the existing code binaries already in memory. For example, the return-to-libc attack modifies a return address saved in the memory so the program

 0018-9162/18/$33.00 © 2018 IEEE

will "call" an existing C library function, instead of returning to the caller. A more evolved example is return-oriented programming (ROP) and its variants.[2-5] The attack forms "gadgets," each of which is a short sequence of instructions that ends with an indirect branch instruction such as call, return, and jump with register, from legitimate code already in memory. Then the attack payload is a sequence of addresses for the gadgets to compose a desired functionality. A sufficient number of gadgets can form an attack program with arbitrary functionality ("Turing complete").

Maintaining control-flow integrity (CFI), that is, ensuring at runtime that the program control-flow is following the control-flow information implied at the machine instruction level in the program, can be a basic principle for providing sound protection against most CRAs.[6,7] CFI implementations instrument the program to validate the control-flow transfer of an indirect branch instruction instance using a reference control-flow graph (CFG) generated statically or dynamically from the program. Ensuring that each indirect branch instance follows the CFG involves validating the target address for the given indirect branch instance.

A CFG can be viewed as a table with entries consisting of an indirect branch instruction location and its target address, called an indirect branch pair (IBP). Following the view, it is possible to see the similarity between the CFI per the CFG and the branch prediction found in almost all modern processors; the branch prediction verifies the predicted target address for a branch instruction instance. One key observation is that the branch prediction, especially for indirect branch instructions, is already doing control-flow validation: the target addresses stored in the branch prediction unit (BPU) provide recently encountered and validated IBPs.

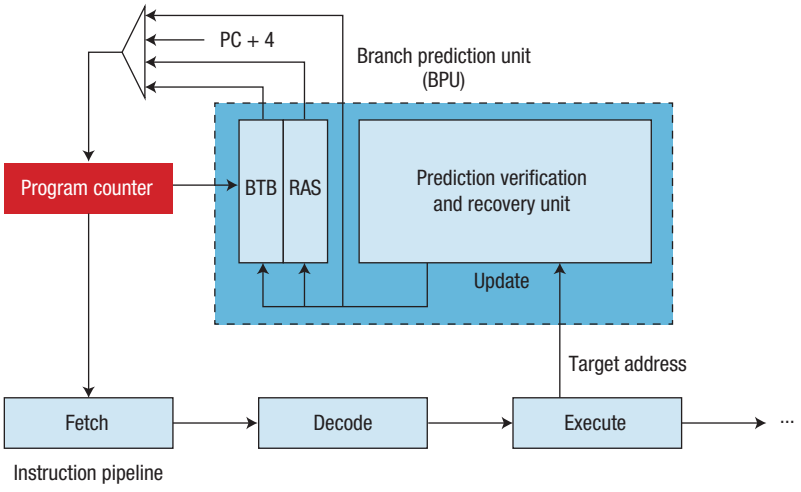


FIGURE 1. Branch prediction allows the processor to move on with the predicted target address from the branch target buffer (BTB) or return address stack (RAS). If the predicted target address turns out to be incorrect later in the instruction pipeline, the processor re-fetches the instruction with the correct target address (available usually at the execute stage of the branch instruction) and nullifies the instructions fetched with the predicted target address.

We propose incorporating the control-flow validation into the processor's instruction execution pipeline. Instead of adding instrumentation for CFI implementation, control-flow validation becomes an integral part of the indirect branch instruction execution, which will provide tighter protection and, at the same time, less performance overhead. This article introduces a mis-prediction validation unit (MVU) alongside the BPU for control-flow validation. An MVU validates the mis-predicted target address, whether it is caused by a legitimate target address for a legitimate control-flow or a compromised target address from an attack. Since an MVU works in tandem with a BPU, it allows us to narrow the control-flow validation scope within the mis-predicted branches and to incorporate an MVU into the instruction execution pipeline with little overhead. An MVU allows control-flow validation to be integrated into the processor's instruction execution pipeline like a memory management unit (MMU) enabling the virtual-to-physical address translation to be an integrated part of the instruction execution pipeline.

CODE REUSE ATTACKS AND BRANCH PREDICTION

When fetching an instruction for execution, modern pipelined processors use branch prediction to fetch the next instruction without waiting for the target address of the branch instruction. In doing so, the BPU stores a record of previous target addresses. For example, the

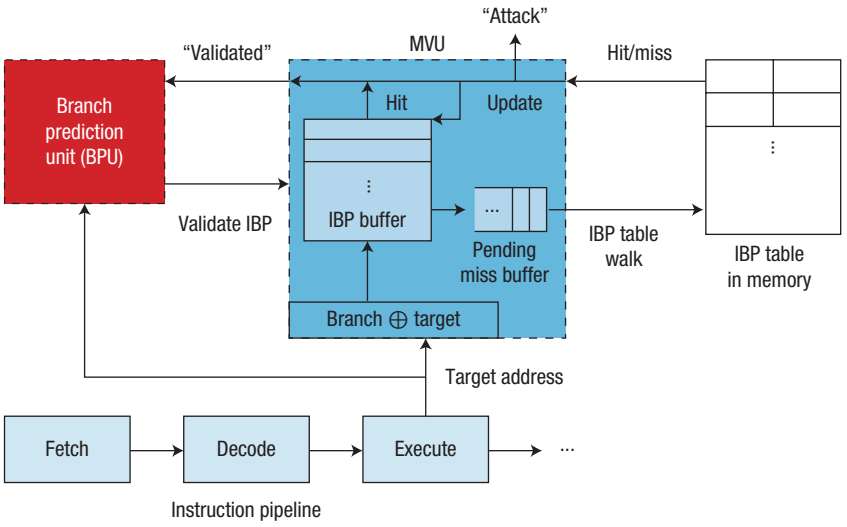


FIGURE 2. Mis-prediction validation unit (MVU). While the branch prediction unit (BPU) verifies whether the predicted target is correct, the MVU validates the indirect branch pair (IBP) of the mis-predicted indirect branch instruction to be sure it is not compromised.

last encountered target address of a branch instruction is stored in a small buffer called the branch target buffer (BTB), and the return address for the last call is stored in a fast hardware stack, called the return address stack (RAS).[8] For the current program counter (PC) value, the BPU checks if the corresponding entry exists in the record of previous target addresses. If so, the target address found in the record becomes the predicted target address. Otherwise, the next physical instruction is predicted.

The processor uses the predicted address to fetch the next instruction for execution without waiting for the branch instruction to provide the correct target address. While the processor moves on with the predicted target address, the BPU waits to verify if the predicted target is correct (branch prediction verification) until the target address from executing the branch instruction becomes available, which is usually at the execute stage of the processor's instruction pipeline. If the predicted target address turns out to be incorrect, that is, it is a mis-prediction, the processor rolls back the PC and re-fetches the instruction with the correct target address (mis-prediction recovery) (see Figure 1).

With no control-flow compromises, a mis-prediction occurs if an indirect branch instruction instance takes a different target address, out of several legitimate target addresses, from the previous instances. Under a CRA, a branch mis-prediction occurs when the control-flow transfer is hijacked to an unexpected position in the existing code because it is not a part of the program's legitimate control-flow and has not been previously executed. A branch target used by the attack might be legitimate, but it causes a branch mis-prediction because the target address is not the one that results from a normal execution of the branch instruction at the location. Alternatively, the branch location could be legitimate while the target address is not. In either case, the IBP is not legitimate. Note that from the control-flow validation perspective, the target addresses from branch prediction represent uncompromised addresses, because every past target address is already validated if the validation process is in place. We can use this fact to narrow the validation scope for detecting CRAs down to mis-predicted indirect branch instances.

MIS-PREDICTION VALIDATION UNIT

To detect a CRA, the mis-prediction validation unit (MVU), along with the branch prediction (see Figure 2), distinguishes whether the cause for a branch mis-prediction is from a legitimate control-flow transfer or from a compromised control-flow transfer by checking the reference CFG.

Validating Branch Mis-prediction

To see how a CRA changes the indirect branch pair (IBP)—that is, the information on the branch location and its target address—let us consider a return-to-libc attack as an example. A return-to-libc attack compromises the return address saved in the runtime stack to cause an illegitimate control-flow to a C library function. When the PC points to the return instruction, the processor fetches its target instruction with the predicted return address from the RAS without waiting to see the return address saved in the runtime stack. When the return address from the runtime stack becomes available later as the return instruction is executed, the prediction verification compares it with the predicted one from the RAS.

Under the attack, the return address from the runtime stack in memory is the one compromised, while the one from the RAS is not. The branch verification concludes it is a mis-prediction because the return address from the runtime stack and the predicted one do not concur. Unfortunately, when the recovery process for the branch mis-prediction rolls back the processor's progress with the predicted return address and re-fetches the instructions, it uses the compromised return address from the runtime stack. To avoid reverting the legitimate flow to the compromised flow of the attack, an MVU validates the IBP for the mis-predicted branch, that is, whether the mis-prediction is due to a compromised control-flow from the attack.

This paper assumes that the IBP table—namely, the set of IBPs collected from the profiled traces—is the reference CFG in memory. One might also generate the IBP table statically as in most CFI implementations; however, we have used the profiled traces for ease of experimentation, including obtaining the IBP information for dynamically linked routines. We assume that the IBP table is loaded along with the code binary.

Note that each IBP provides a unique identifier for each indirect branch with each of its target addresses. Assuming the code text area in memory is read only, the IBP set does not include direct branch instructions including conditional branches, because their target addresses are fixed and described as a part of the instruction bit pattern itself. The IBP table makes an ideal fine-grained CFG in terms of identifying each indirect branch instance uniquely.[6,7,9]

For IBP table access, we provide a small cache, called the IBP buffer.

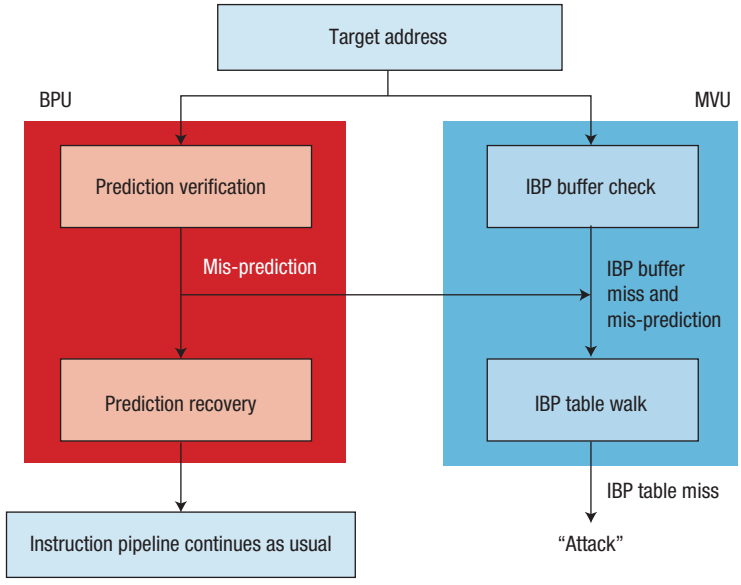


FIGURE 3. Overall mis-prediction validation flow.

Instead of loading the IBP table into the data cache, a separate software transparent buffer provides the isolation to secure the IBP information. Note that the IBP buffer is not read or written directly by software. It is read and updated by the MVU as an integral part of the indirect branch instruction execution. When the BPU goes through the branch verification and recovery, the MVU validates the IBP for the mis-predicted branch instance by accessing the IBP buffer (see Figure 3). If the MVU finds the IBP in the IBP buffer, it needs no action. As long as the IBP buffer access takes the same or less time for the branch prediction verification and recovery, the control-flow validation by the MVU is integrated into the existing instruction pipeline with no extra delay.

Since the MVU can only start checking the IBP buffer after the target address from the branch instruction is available, one might think that the MVU's access to the IBP buffer will add a delay to the instruction execution pipeline. When the target address is available from the branch instruction, it is provided to the BPU for prediction verification. If it turns out to be a mis-prediction, the recovery process rolls back the instruction fetches with the verified target address. This verification and recovery process adds a mis-prediction penalty of a few cycles to the processor's instruction pipeline. Since the target address from the branch instruction is available to the MVU at the same time as to the BPU, an MVU can finish the validation early enough to not cause additional delay if it finds the IBP in the IBP buffer.

If the MVU experiences an IBP buffer miss, it needs to access the IBP table in memory (IBP table walk). The MVU needs to signal an attack if the IBP table walk fails: an IBP table search failure means the IBP is not legitimate, that is, there is a compromised control-flow

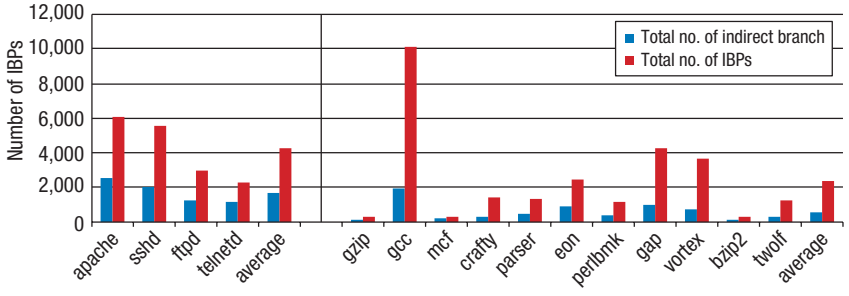


FIGURE 4: Number of IBPs. Indirect branch profiling for SPEC2000 CPUint and four server programs: apache, sshd, ftpd, and telnetd.

from the attack. Since the IBP table walk may take a long time because of the memory access, it will take longer than the BPU's branch verification and recovery. However, the IBP table walk for handling an IBP buffer miss does not need to interfere with the program execution; the instruction pipeline can move on without waiting for the MVU's validation result.

In normal situations, namely, when the control-flow is not compromised, the MVU does not interfere with the program execution. With a compromised IBP, the program execution needs to be blocked anyway as the MVU's IBP table walk failure prevents the attack from progressing. The MVU's control-flow validation can be decoupled from the instruction pipeline by introducing a small buffer to hold incoming mis-predicted indirect branches when an IBP table walk is pending (see "pending miss buffer" in Figure 2). Without the buffer, an MVU would need to stall the instruction pipeline until the IBP table walk finishes, which introduces a performance penalty.

IBP Buffer

To see how large the IBP table might become, we collected IBPs from SPEC2000 CPUint benchmark programs and also from four popular server programs: apache, sshd, ftpd, and telnetd. For the server programs, we collected IBPs from actual daily use and also with synthetic input data. We ran the SPEC benchmark programs with the provided regular input datasets from start to finish without any skipping. We used the Intel VTune Analyzer to collect the IBPs. We found the indirect branch instructions to be about 1.5 percent of all the instructions executed.

The profiling results show that the IBP tables are quite modest in size: the IBP table size for all the programs except *gcc* is less than 6,000, whereas it is slightly above 10,000 IBP's for *gcc* (see Figure 4). Although we know from the programs used in our experiments that IBP table size is usually small, it is still desirable to have a cache for the IBP tables to assist the MVU to validate branch mis-predictions with a shorter delay than the BPU's mis-prediction penalty.

Multiple target addresses are possible for a given indirect branch instruction. Although it varies from program to program, each indirect branch in the programs we tested has two to five different targets on average (see Figure 3). To reduce the IBP buffer miss rate, it is desirable to allow multiple IBPs to be stored in the buffer for a given PC value at the same time. Instead of indexing the IBP buffer via PC only, doing an exclusive-or (XOR) of the branch address and its target address will distribute its IBPs over the buffer address space. We might want to improve the IBP buffer hit rate with a set-associative structure to store multiple target addresses for each indirect branch in a more explicit way. However, we did not explore further IBP buffer refinement because even a small buffer can provide a very high hit rate with the simple XOR indexing.

PERFORMANCE OVERHEAD

To evaluate the potential performance overhead from the MVU's control-flow validation, we studied SPEC2000 CPUint benchmark programs with a Simplescalar-3.0 simulator, simulating a 4-wide issue out-of-order 9-stage pipeline core with 64 KB L1 data and instruction caches. We simulated each benchmark for 1 billion committed instructions after fast-forwarding for the first 100 million instructions. We used the following assumptions for the evaluation environment, the processor, and the MVU.

Processor:

- Pipeline—4-issue, 9-stage
- Issue queue size—16
- Reorder buffer size—64
- Branch predictor—gshare predictor with 4,096 counters and 16-entry RAS
- Mis-prediction penalty—5 cycles
- L1 caches—64 KB instructions/64 KB data, 4-way, 2-cycle hit latency
- Memory—dual ported with 100-cycle latency

Mis-prediction Validation Unit (MVU):

- IBP buffer—16 KB, 2K entry of IBP (PC, its target address) with 1-cycle hit latency
- IBP buffer indexing—XORing of PC and its target address
- IBP buffer miss penalty—200 cycles or 500 cycles (for IBP table walk)

We considered several IBP buffer sizes, from 512 entries to 8K entries. We charged 200 cycles for the MVU's access of the IBP table on an IBP buffer miss. An IBP table walk on an IBP buffer miss is simpler than the page table walk by MMU on a translation lookaside buffer (TLB) miss for virtual address translation, which is known to take three to five memory accesses because of the hierarchical structure of the page table.

Note that with a 32-bit address space, the page table for a single process has a million entries with a 4-KB page size, whereas the IBP table size is usually on the order of thousands. However, for the sake of generality, we also did our experiments with a 500-cycle penalty for the IBP table access. Recall that MVU accessing the IBP table in memory does not necessarily cause a performance penalty. The program execution will be blocked only if there is no matching IBP, namely, that an attack is detected. However, to have a conservative performance estimate, we have the processor wait in our simulation until the MVU's access of the IBP table in memory finishes, issuing the "validated" signal (as shown in Figure 2).

We observed that with buffer sizes of 1K entries and larger, the IBP buffer provides over 99 percent hit rates for all the benchmark programs. With an IBP buffer of 8K entries, the miss rate is 0.075 percent on average, ranging from 0.001 percent for *eon* to 0.16 percent for *gcc*. Our simulations suggest that the 2K-entry IBP buffer would be the most cost-effective with an average miss rate of 0.097 percent, ranging from 0.001 percent for *eon* to 0.297 percent for *gcc*. We selected the 2K-entry IBP buffer for our performance evaluation experiments.

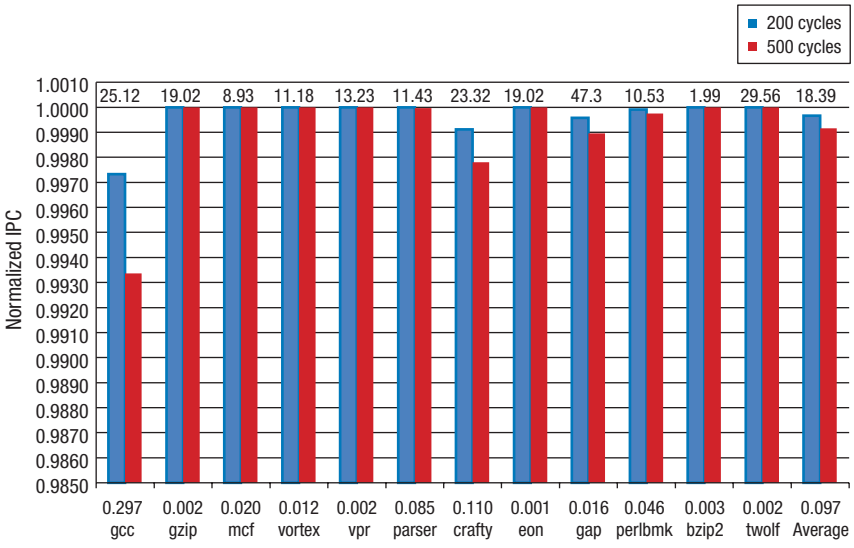


FIGURE 5. MVU performance overhead in terms of normalized instructions per cycle (IPC) over the case without an MVU's control validation. Above the bars for each program is the mis-prediction rate (%) for indirect branches including return, and below the bars for each program is the IBP buffer-miss rate (%).

Three factors determine the MVU performance overhead. The first is the ratio of indirect branch instructions to all instructions. The second is the mis-prediction rate for indirect branch instructions. The third is the IBP buffer miss rate. With an IBP buffer miss, MVU needs to access the IBP table in memory and replace the corresponding IBP buffer entry. Recall that an access of the IBP buffer happens only on a branch mis-prediction for indirect branches. With relatively fewer occasions of encountering indirect branches along with a decent branch prediction success rate, the low IBP buffer miss rates suggest little performance impact by the MVU.

Even with our conservative assumption of stalling the instruction pipeline at every IBP buffer miss, the MVU is shown to be an extremely efficient mechanism (see Figure 5). The performance overhead in terms of normalized instructions per cycle (IPC) is, with a 200-cycle IBP buffer-miss penalty, 0.034 percent on average with the highest overhead of 0.26 for *gcc*. This is an order of magnitude or two less performance overhead than the CFI implementations with added instrumentation for control-flow validation. For example, an early CFI implementation with a static CFG[6] reported the average overhead of 21 percent, while recent coarse grain CFI implementations have reported the average

overhead of 1 percent[10] and 2.14 percent.[11] The low performance overhead of the recent CFI implementations comes from using heuristics on indirect branch behavior along with hardware support. However, such heuristics expose vulnerabilities for CRAs.[7] The main contributor to the efficiency of the MVU is pruning the number of indirect branch instruction instances to validate. On average, about 82 percent of the control-flow validation in our experiments was done by the BPU without accessing the IBP table.

In our experiments, the last value prediction scheme was used—storing a single last-seen target address for an indirect branch in the BTB. Moreover, the RAS assumed was a relatively small straightforward design with 16 entries. Branch prediction for indirect branches can improve significantly by providing a separate jump target cache or storing multiple targets in the BTB for an indirect branch.[12] With the improved prediction success rate, the already very low performance overhead will decrease even further.

Our experimental results are in a single process environment and do not consider context-switching effect. Our experiments with a higher IBP buffer-miss penalty of 500 cycles are somewhat reflective of an environment running multiple processes together. However, even with the conservative assumption of stalling the instruction pipeline at every IBP buffer miss, while also using the higher IBP buffer-miss penalty, the performance overhead remains low: less than 0.1 percent on average.

DETECTING CODE REUSE ATTACKS

A code reuse attack based on ROP or its variants will mount the attack by linking the gadgets. Each gadget is a short consecutive portion of a function residing in memory that ends with an indirect branch instruction. With an MVU in place for control-flow validation, a CRA cannot proceed unless only legitimate control-flow transfers are used. Further, the MVU's control-flow validation integrated into the instruction execution pipeline provides less chance for a CRA to bypass the validation than the added instrumentation for existing CFI implementations. For example, in the Intel x86 instruction set architecture (ISA) with variable instruction length, an attack can "create" indirect branches by fetching the instruction byte from the middle of multi-byte instructions.[5] These unintended branch instructions are not protected by the inline code instrumentation for CFI because it is difficult, if not impossible, to guess where to insert the inline code instrumentation to check them.

Most CFG representations suffer from the lack of fully accurate and complete context information.[9] For example, if two returns for two different legitimate call sites for a shared procedure are mixed up in a sequence different from the original program, it will be deemed legitimate because the two IBPs for the two returns are legitimate per the CFG. A CRA that uses only legitimate control-flow transfers per the CFG, but uses a different control-flow transfer sequence from the uncompromised original program, can evade the protection because each control-transfer is considered independent of the other control-transfers.

CRAs have evolved into more sophisticated types, and will continue to do so. With more sophisticated CRAs, more complex fine-grained CFGs are needed. Although incorporating a more complex fine-grained CFG with the MVU needs further investigation, we can note that the context information—such as the conditional branch history or execution path information, in addition to the IBPs—is readily available in BPUs. Using branch prediction in an MVU generally benefits CFI implementations by reducing the need for control-transfer validation, and the benefit will be greater with more complex and context-sensitive CFGs.

In this article, we have considered validating each control-transfer instance as an integral part of indirect branch instruction execution. An MVU working in tandem with the BPU found in most processors validates every control-flow transfer instance in terms of IBP, that is, its indirect branch location and its target address. Since recently encountered and validated IBPs are readily available in the BPU, BPUs do most of the validations, about 82 percent in our experiment, whereas the MVU does the validation for the mis-predicted indirect branches. To enable an MVU to validate the control-flow without delaying the instruction execution pipeline, we have introduced a small buffer (IBP buffer) that holds the IBPs for the indirect branches mis-predicted by the BPU.

Our performance simulations of the SPEC CPU integer benchmark programs via the SimpleScalar simulator show that the MVU incurs a 0.034 percent performance overhead (in terms of the IPC) on average. Moreover, we can disregard this negligible MVU performance overhead because it comes from our conservative assumption that every IBP buffer miss adds a delay to the program execution. However, an MVU's handling of IBP buffer misses

does not need to interfere with the program execution in normal situations with no attack—namely, when the control-flow is not compromised. Further, an MVU's integration of control-flow validation into the instruction pipeline is software-transparent, making it more difficult for the attacks to bypass.

ACKNOWLEDGMENTS

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science, and Technology (2015R1A2A2A01).

REFERENCES

1. Y.-J. Ahn et al., "Monitoring Translation Lookahead Buffers to Detect Code Injection Attacks," *Computer*, vol. 47, no. 7, pp. 66–72.
2. E. Buchanan et al., "When Good Instructions Go Bad: Generalizing Return-Oriented Programming to RISC," *Proc. 15th ACM Conf. Computer and Communications Security* (CCS 08), Oct. 2008, pp. 27–38.
3. S. Checkoway et al., "Return-Oriented Programming without Returns," *Proc. 17th ACM Conf. Computer and Communications Security* (CCS 10), Oct. 2010, pp. 559–572.
4. F. Schuster et al., "Counterfeit Object-Oriented Programming: On the Difficulty of Preventing Code Reuse Attacks in C++ Applications," *Proc. IEEE Symp. Security and Privacy* (S&P 15), 2015, pp. 745–762.
5. H. Shacham, "The Geometry of Innocent Flesh on the Bone: Return-into-libc without Function Calls (on the x86)," *Proc. 14th ACM Conf. Computer and Communications Security* (CCS 07), Oct. 2007, pp. 552–561.
6. M. Abadi et al., "Control Flow Integrity Principles, Implementations, and Applications," *ACM Transactions on Information and System Security* (*TISSEC*), vol. 13, no. 1 (article no. 4), Oct. 2009.
7. E. Goktas et al., "Out of Control: Overcoming Control-Flow Integrity," *Proc. IEEE Symp. Security and Privacy* (S&P 14), 2014, pp. 575–589.
8. J. Hennessy and D. Patterson, *Computer Architecture: A Quantitative Approach*, 2012, Morgan Kaufmann.
9. N. Carlini et al., "Control-Flow Bending: On the Effectiveness of Control-Flow Integrity," *Proc. 24th USENIX Conf. on Security Symp.*, 2015, pp. 161–176.
10. V. Pappas, M. Polychronakis, and A.D. Keromytis, "Transparent ROP Exploit Mitigation Using Indirect Branch Tracing," Proc. 22nd USENIX Conf. Security Symp., 2013, pp. 447–462.
11. M. Kayaalp et al., "Branch Regulation: Low-Overhead Protection from Code Reuse Attacks," *Proc. Int'l Symp. Computer Architecture* (ISCA 12), 2012, pp. 94–105.
12. H. Kim et al., "Virtual Program Counter (VPC) Prediction: Very Low Cost Indirect Branch Prediction Using Conditional Branch Prediction Hardware," *IEEE Transactions on Computers*, Vol. 58, No. 9, 2009, pp. 1153–1170.

ABOUT THE AUTHORS

YONGSUK LEE is currently a PhD student in the Department of Computer Science & Engineering at Korea University. His research interests include computer architecture, trusted computing, and system security. Lee received his MS in computer science and engineering from Korea University, Seoul. Contact him at duchi@korea.ac.kr.

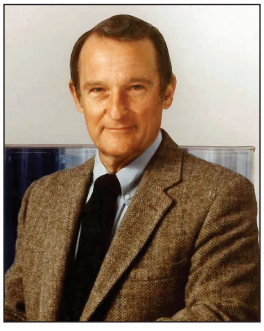
GYUNGHO LEE is a professor in the Department of Computer Science & Engineering at Korea University. His research and teaching interests include computer architecture, specifically in the areas of microprocessor architecture, system security, and code optimization. Lee received his PhD in computer science from the University of Illinois at Urbana-Champaign in 1986. He is a fellow of the American Association for the Advancement of Science (AAAS). Contact him at ghlee@korea.ac.kr.

Recognizing Excellence in High Performance Computing

Nominations are Solicited for the

SEYMOUR CRAY SIDNEY FERNBACH & KEN KENNEDY AWARDS

SEYMOUR CRAY COMPUTER ENGINEERING AWARD



Established in late 1997 in memory of Seymour Cray, the Seymour Cray Award is awarded to recognize innovative contributions to high performance computing systems that best exemplify the creative spirit demonstrated by Seymour Cray. The award consists of a crystal memento and honorarium of US$10,000. **This award requires 3 endorsements.**

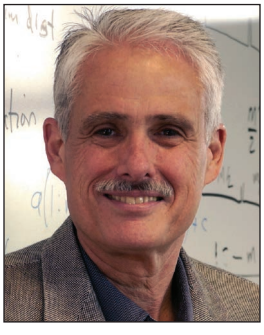
Sponsored by: IEEE computer society

SIDNEY FERNBACH MEMORIAL AWARD

Established in 1992 by the Board of Governors of the IEEE Computer Society. It honors the memory of the late Dr. Sidney Fernbach, one of the pioneers on the development and application of high performance computers for the solution of large computational problems. The award, which consists of a certificate and a US$2,000 honorarium, is presented annually to an individual for "an outstanding contribution in the application of high performance computers using innova-tive approaches." **This award requires 3 endorsements.**

Sponsored by: IEEE computer society

ACM/IEEE-CS KEN KENNEDY AWARD



Established in memory of Ken Kennedy, the founder of Rice University's nationally ranked computer science program and one of the world's foremost experts on high-performance computing. A certificate and US$5,000 honorarium are awarded jointly by the ACM and the IEEE Computer Society for outstanding contributions to programmability or productivity in high performance computing together with significant community service or mentoring contributions. **This award requires 2 endorsements.**

Cosponsored by: IEEE computer society acm Association for Computing Machinery

Deadline: 1 July 2018

All nomination details available at http://awards.computer.org

IEEE

International Neuroscience Initiatives through the Lens of High-Performance Computing

Kristofer E. Bouchard, Lawrence Berkeley National Laboratory and UC Berkeley
James B. Aimone, Sandia National Laboratories
Miyoung Chun, Kavli Foundation
Thomas Dean, Google and Stanford University
Michael Denker and Markus Diesmann, Jülich Research Center
David D. Donofrio, Lawrence Berkeley National Laboratory
Loren M. Frank, UC San Francisco and Howard Hughes Medical Institute
Narayanan Kasthuri, Argonne National Labs and University of Chicago
Christof Koch, Allen Institute for Brain Science
Oliver Rübel and Horst D. Simon, Lawrence Berkeley National Laboratory
F.T. Sommer, UC Berkeley
Prabhat, Lawrence Berkeley National Laboratory

Neuroscience initiatives aim to develop new technologies and tools to measure and manipulate neuronal circuits. To deal with the massive amounts of data generated by these tools, the authors envision the co-location of open data repositories in standardized formats together with high-performance computing hardware utilizing open source optimized analysis codes.

Many international neuroscience initiatives are in different stages of progression—although they have different goals, they will all produce large amounts of data. Much attention has been focused on the technological challenges of measuring and manipulating neural activity from large numbers of sites for long periods, but much less attention has been paid to the computing challenges associated with the vast amounts of data these technologies will generate. As a result, potential advances offered by neurotechnologies are threatened by a lack of computing tools. The neuroscience community is not alone in this challenge, as other science fields are being transformed by advanced analytics being applied to an ever-increasing volume of experimental data. Co-location of massive datasets hosted in open repositories with high-performance computing (HPC) will allow for community-driven exploratory analysis and integration with simulations. This is required to extract universal design principles of biological computation, which might

0018-9162/18/$33.00 © 2018 IEEE

provide insight into and inspiration for new models of in silico computation.

GRAND CHALLENGE PROBLEMS IN NEUROSCIENCE

To understand the brain is to know how its structure (wiring diagram) and function (activation dynamics) give rise to specific computations and behaviors. Following the goals of the Brain Research through Advancing Innovative Neurotechnologies (BRAIN) Initiative and the EU Human Brain Project (HBP), we propose four grand challenge problems in neuroscience for which HPC will likely play an important role (see Figure 1): neuroanatomy and structural connectomics; neural population dynamics and functional connectomics; linking sensations, brains, and behaviors; and synthesis through simulations. While each challenge would provide useful information by itself, integrating them will result in synergistic understanding. Taken together, the results of these challenges will deepen our understanding of network mechanics that generate complex behaviors and the transformation of sensory inputs into neural representations of sensations. Furthermore, the results will provide insight into how brains achieve near-optimal computing capabilities and link structure to function across many spatiotemporal scales.

Neuroanatomy and structural connectomics

Across species, brains consist of hundreds to billions of individual neurons that are connected by thousands to trillions of synapses (see Figure 2). These anatomical features are the structural backbone from which all neuronal function is generated. In its most microscopic form, *structural connectomics* refers to the reconstruction of tissue at sufficient resolution to trace the finest neuronal processes and identify synaptic connections. Currently, the only approaches that offer the required resolution over large volumes are based on automated serial electron microscopy.

Advances in microscope design have improved resolution and acquisition time so that segmentation and annotation is the rate-limiting step.[1] The result of such anatomical reconstruction could be a full 3D representation of each neuron or a graph of the resulting structural connectivity matrix with some measure of synaptic strength (the matrix C_s: N neurons × N neurons). A structural connectome would provide a compact summary sufficient for some analyses, and is required to link structure to function in the nervous system.

Neural population dynamics and functional connectomics

Although neuroanatomy defines the possible interactions among neurons, it is the dynamically modulated spatiotemporal patterns of activations across neurons that give rise to sensations, actions, cognition, and consciousness. New technologies enable increasingly large numbers of brain signals to be recorded simultaneously, and these signals can be derived from diverse recording modalities. Furthermore, the duration of recordings is concurrently increasing, and it will soon be possible to record continuously for weeks to months. Thus, it is becoming increasingly important to develop data-analysis methods capable of revealing structure from heterogeneous, nonstationary time-series measurements at scale.

Challenges	
Structural connectomics	
Scaling:	neurons^3
Dataset:	serial electron microscopy of brains
Product:	circuit wiring diagram
Functional connectomics	
Scaling:	neurons^2 × time
Dataset:	in vivo activation levels of neurons
Product:	circuit dynamics
Sensations, brains, and behaviors	
Scaling:	modalities × time
Dataset:	sensory and/or behavioral measurements
Product:	computations performed by circuits
Biophysically detailed simulations	
Scaling:	synapses × time
Dataset:	connectivity and biophysics of neurons
Product:	bridge across spatiotemporal scales

FIGURE 1. Grand challenge problems in neuroscience. We pose four grand challenge problems in neuroscience, which, at scale, will require high-performance computing (HPC). This figure summarizes how problems approximately scale with key features and provides example inputs (data types) and outputs (insights gained) associated with each problem. Note that each of these problems scales approximately as the product of at least two key features of the dataset (for example, neurons^2 × time). (Source: Christian Swinehart, Samizdat Drafting Co.)

Two complementary approaches to this problem are dimensionality reduction and functional connectomics. Dimensionality reduction methods aim to find low-dimensional spaces that concisely summarize high-dimensional spatiotemporal patterns of activity, and can be used to gain insight into network dynamics. Functional connectomics aims to determine time-resolved causal influences among spatially distributed neural recordings (for example, the data array C_f: N neurons × N neurons × time for cellular-level data). Together, these complementary methods will provide insight into dynamic interactions among individual neurons and neural populations that can be linked to underlying structural connectomics and behavior.

Linking sensations, brains, and behaviors

Brains have evolved to produce behaviors in response to sensory events that

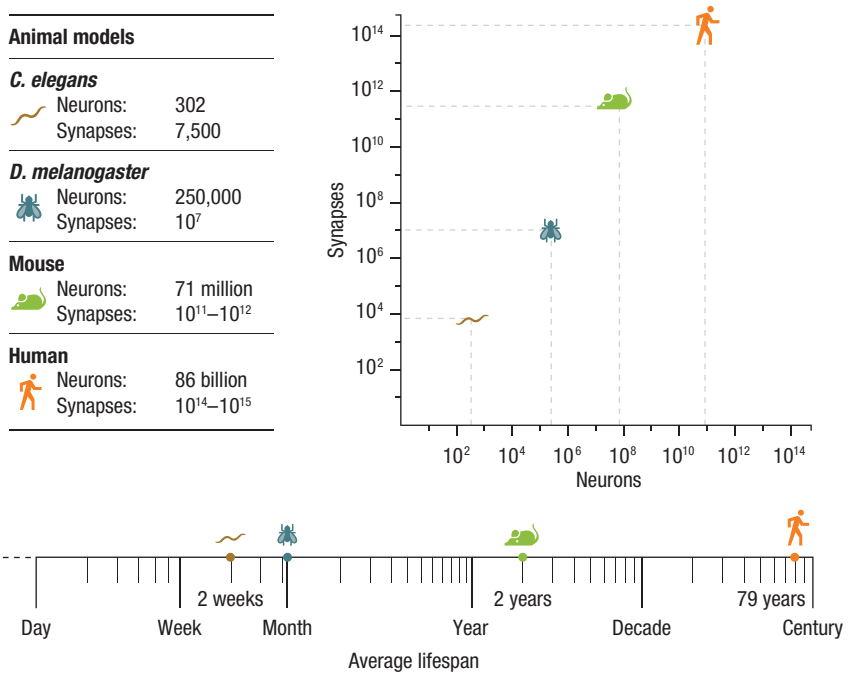


FIGURE 2. Neuroscience datasets will scale exponentially across species. Neuroscientists study brains across several species of varying complexity. This figure depicts the exponential grown in brain size (number of neurons and synapses) and lifespan across *C.elegans* (worms), *D. melanogaster* (flies), mice, and humans. (Source: Christian Swinehart, Samizdat Drafting Co.)

increase the probability of organismal survival and reproductive fitness. As such, our ability to infer brain function depends on linking brain measurements to events and objects in the external world. In many cases, the stimuli and behaviors used in neuroscience studies have been relatively simple (low-dimensional), which eases data analyses but elicits neural activity far removed from activity underlying naturalistic sensations and behavior. Indeed, it has recently been noted that simple tasks result in simple neural activity patterns.[2] Thus, it is insufficient to record from more neurons without simultaneously monitoring behavior during increasingly complex sensory, motor, and cognitive tasks. This brings with it challenges of acquiring, analyzing, and integrating such multimodal data (for example, visual, audio, haptic, and movement) with the brain data.

A causal understanding of the brain requires combining measurements with closed-loop manipulations of neural circuits triggered by neuronal and behavioral observations during complex tasks. However, this is very challenging because the activity of many neurons is rapidly modulated (tens of milliseconds) by sensations, cognitive tasks, and behavior. Given this complexity, manipulations need to be targeted to specific neurons when they are engaged in specific types of information processing, requiring real-time analysis of neural signals.

Synthesis through simulations

Ultimately, the goal of neuroscience is to achieve a deeper, broader understanding of the brain that extends across spatial and temporal scales. However, simply acquiring data without simultaneously developing guiding (theoretical) principles will impede extracting understanding from the data, and runs the risk of misguided investment into costly experiments. As other fields have shown, common computational and theoretical frameworks should permeate research directions while scaling up data acquisition and analysis to reduce the challenge of integrating information from very different levels of system granularity.

Many research domains focused on highly complex, multiscale problems (for example, climate, high-energy physics, and cosmology) have effectively leveraged HPC capabilities to integrate data and analysis into evolving theoretical frameworks through the use of simulations. This has been useful in precisely those conditions where extensive experimentation is intractable due to either cost or feasibility of data acquisition. While a general "theory of brain" seems a distant goal, using HPC for large-scale simulation of neural circuits and networks has a long history. Continued scaling (both in number and accuracy) of neural circuit simulations, as well as tighter integration with experimental data, is critical to connect spatiotemporal scales.

DIVERSE COMPUTING PLATFORMS FOR A DIVERSE COMMUNITY

Modern computing solutions are as diverse as the needs of the neuroscience community, and it is unlikely that there will be a one-size-fits-all solution to all needs. Indeed, there are several important tradeoffs in performance, cost, and accessibility associated with different computing platforms. Neuroscientists should be aware of these when deciding on current solutions and planning long-term investments.

While many neuroscientists are familiar with computing capabilities contained within a single laptop or by a shared cluster, there is less familiarity with the resources available or what problems may be tackled by cloud computing or supercomputing facilities.

Cloud computing infrastructure fundamentally relies on commodity hardware with somewhat better network interconnects (10 Gbytes) than typical university clusters. The cloud is popular because of the ease with which resources can be provisioned and software services can be utilized, without exposing users to fault-prone hardware. HPC provides well-balanced CPU and memory subsystems, tightly coupled with high-performance interconnects, and data is typically read over massively parallel file systems capable of terabyte-per-second read/write performance. In aggregate, state-of-the-art HPC centers are capable of holding hundreds of terabytes of data in memory, and can calculate at the rate of petaflops. In contrast to cloud computing, where productivity is the norm, the software stack on HPC resources is tuned for performance, and it does take some degree of expertise and familiarity to fully utilize such systems. As the rate of improvement in computing processors decreases from a slowing of Moore's law, commodity computers (and clusters of them) will not be able to address the ever-increasing volumes of neuroscientific data.

In neuroscience, there also exists a need for experimenters to rapidly interrogate their data to receive timely feedback on results of experiments (for example, to evaluate the position of a recording device). There are other experiments (such as closed-loop perturbations) that require real-time analysis (<5–10 ms). These latencies can be achieved with carefully designed PC-based systems or in a more cost-effective fashion through specialized hardware utilizing field-programmable gate arrays (FPGAs). In this context, the power of FPGAs comes from their flexible and high bandwidth (terabytes per second) connections and their ability to manipulate data from multiple sources. However, programming FPGAs can be challenging, and while familiar to many in the computing world, it is a skill rarely found in the neuroscience community. Finally, for on-sensor processing of massive datastreams from large-scale experimental equipment for which the processing algorithm has been established and agreed upon, application-specific integrated circuits (ASICs) can provide a critical filter, dramatically reducing the amount of data that needs to be moved and written to disk.

DEFINITIONS OF TERMS

Brain signals: There are many diverse signals that can be used as measures of brain function, including intracellular and extracellular electrical recordings, optical imaging of neuronal voltage/calcium, electrocorticography (ECoG), magnetoencephalography (MEG), and functional magnetic resonance imaging (fMRI). The spatial and temporal resolution of different signals is generally proportional to the invasiveness of the methods used to sense the signal.

Closed loop: Used here to describe an experiment that uses the signals generated by a system (such as the brain) to trigger a perturbation of that system.

Electrophysiological recordings: The recording of electrical activity generated by the biophysical processes involved in neuronal signaling. Electrophysiology provides the highest temporal resolution measurements of brain signals, capable of resolving the precise timing of individual action potentials ("the speed of thought").

Neuronal skeletons: The external structure of single neurons, including the cell body, dendrite, axon, and associated synapses.

Performance: A metric typically correlated with obtaining a quick turnaround time to solution. A well-balanced computational system will optimize for storage, network, memory, and compute performance.

Productivity: A metric typically associated with effort spent by developers and programmers in developing code and utilizing computational infrastructure.

Spike sorting: The problem of assigning recordings of action potentials ("spikes") to individual neurons from extracellular electrophysiological data, which are typically composed of a superposition of electrical signals from multiple neurons.

DATA MANAGEMENT CHALLENGES FOR NEUROSCIENCE

Advanced data management and sharing is required to accommodate the increasing complexity, data rates and acquisition times, and multimodality of neuroscience data. Similar problems occur with neuronal circuit simulations, where standardization of formats and data is an ongoing endeavor. The data-management requirements correspond to specific needs of experimentalists and analysts. Experimentalist requirements include fast write and efficient storage of large data volumes, resilience to corruption to minimize loss, extensible data standards, and collection of metadata with the raw data for reproducibility. Analyst requirements include: common standards to enhance portability and usability for storage, sharing and access; fast read for efficient analysis; integration of distributed, multimodal data sources; and provenance for interpretation and reuse. Standardization of data formats and data sharing are critical to maximize the return on investment into acquisition of experimental data, and will require close interactions among neuroscientists, data model designers, and data analysts.

Data standardization requires convergence of file storage formats and organization, as well as metadata standards and ontologies, and is an unresolved challenge in the field. File format and metadata standards have to be fully supported and well-integrated with acquisition, ideally through automation. To enable efficient analysis, standards also need to be well-integrated with the analysis pipelines, requiring advanced APIs. The need to interpret data in place, combined with efficient discoverability across hardware resources, means metadata should be centrally accessible and machine readable. Integration of multiple modalities requires effective modeling of complex semantic and structural relationships among data. Together, these capabilities will enable the neuroscience community to effectively store, analyze, and share data, accelerating discovery and enhancing reproducibility.

Sharing and reuse of data is essential to enable validation of neuroscience results, which will enhance reliable and unbiased scientific interpretation. The close integration of computing resources with data will enable effective data-driven discovery. This includes co-location of hardware resources to enable efficient processing while reducing costs for large transfers, as well as integrated management and analysis software stacks for analysis at scale. To enable the utilization of shared resources, centralized science gateways/portals that collect data and make it searchable and accessible are needed. Ultimately, data analyses result in commodities that become shared and reused. Similar to the role of metadata for raw measurements, data provenance (including the denotation of methods, parameters, and so on) is required for reliable interpretation of analyses. Meeting all these needs requires advanced, high-performance infrastructure that computer science and HPC centers are ideally positioned to provide.

COMPUTING CHALLENGES FOR NEUROSCIENCE

Repositories hosting data in standardized formats co-located with HPC resources will present the opportunity for creation of automated analysis pipelines at scale. The extraction of information from most modern neuroscience experiments and simulations requires application of sophisticated data-analysis methods. Indeed, each grand challenge problem has data processing (such as segmentation and spike sorting) and analytics challenges associated with it. As there is a cost associated with all computing resources, optimized analysis codes developed by experts in an open source community are required for efficient resource utilization. Building workflows and frameworks for distributed computing, and embedding them in a collaborative setting, requires expertise that is well outside that of typical neuroscientists. Here, although we describe specific analysis issues in the context of a single grand challenge problem, many of the issues apply to all of the problems. Figure 3 schematizes the computing resource requirements, using different species as anchor points, and emphasizes the need for advanced computing solutions at scale.

Neuroanatomy and structural connectomics

The current fastest approach for acquiring structural connectomics data involves scanning multiple electron beams over a brain sample in parallel and is already producing approximately 50 terabytes per day. The major time-limiting step in the analysis pipeline is the stitching, alignment, segmentation, and annotation required to obtain an accurate reconstruction of the brain.[1] The segmentation challenge is compounded by two facts: the task requires tracing most

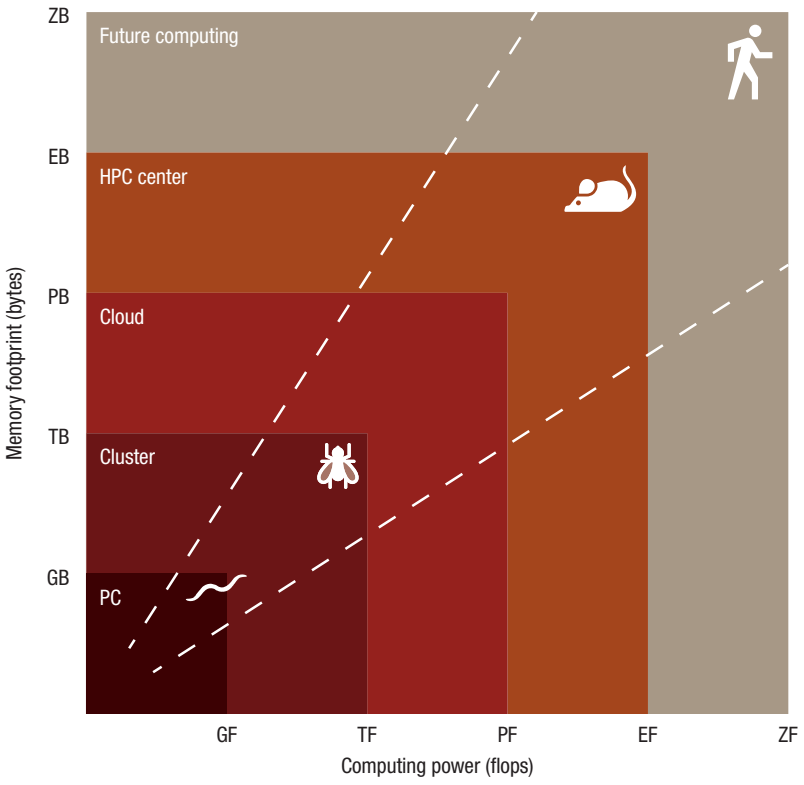


FIGURE 3. Grand challenge problems in neuroscience will push the boundaries of computing. Schematic of the computational demands (computing power [flops] and memory footprint [bytes]) of the grand challenge problems associated with four species. We project that these problems will scale approximately within the boundaries outlined by the dashed line. GB: gigabytes, TB: terabytes, PB: petabytes, EB: exabytes, ZB: zettabytes, GF: gigaflops, TF: teraflops, PF: petaflops, EF: exaflops, ZF: zettaflops. (Source: Christian Swinehart, Samizdat Drafting Co.)

objects in the field of view, and accuracies must be good because even small mistakes tracing an axon could result in thousands of synapses being incorrectly assigned.

The best algorithms for segmentation use recurrent 3D convolutional neural networks to automatically segment raw data.[3] This method performs 60 Mflops per voxel, which for a 100 μm^3 cube of cortex requires 200,000 GPU hours. This is relatively slow when one considers the total amount of computation required to render any reasonably sized circuit. However, once the necessary precision has been achieved, it is reasonable to assume that code optimization will allow processing of cubic millimeter scale samples.

While a cubic millimeter generates about a petabyte (PB) of data, an entire mouse brain will require close to an exabyte (EB) of data. This implies the need to avoid storing raw data long term, and instead processing data as it is generated by co-locating the imaging and computing hardware. This will require special-purpose hardware (such as ASICs and FPGAs) optimized for structural connectomic pipelines. Much of the computational challenge of structural connectomics is image processing, for which HPC systems have long been used. In addition to the 3D reconstruction of all neuropil in the sample, morphological analysis of neuronal skeletons is another important and computationally demanding task. Finally, once raw data has been processed and the resultant connectivity matrix has been extracted, analyses will need to be performed. Rigorous analytics on graphs consisting of 100 billion nodes (number of neurons in the human brain) and 10 billion edges (10 percent average connectivity) requires numerical linear algebra methods that can both exploit the structure of the graph (for example, sparse vs. dense and local vs. global connectivity) and are tailored to the available computing resources.

Neural population dynamics and functional connectomics

In 5 to 10 years, technologies will allow electrophysiological recordings in individual animals from (approximately) 10^6 neurons from brain networks of intermediate size, continuously (temporal resolution of 1 kHz) for many days (for example, 10 days: 8.64×10^8 ms), corresponding to approximately 3.5 PB of time-series data for a single animal (multiply by 25 for the high sampling rate raw data). Much of the analysis for large-scale neural population recording is focused on data-driven discovery, where our knowledge of the ground truth is vague at best. The accuracy of many machine learning/statistical data-analysis methods will be greatly enhanced by increased recording durations (and hence increased numbers of samples) afforded by next-generation recording technologies. Application of state-of-the-art analysis methods to large-scale functional datasets will benefit from efficient implementations in HPC systems.

Neural population activity and dynamics have been examined with dimensionality reduction methods for more than 15 years, and this approach has recently experienced a resurgence. Owing primarily to its computational ease, the most prevalent method in neuroscience is principal components analysis (PCA), the core calculation of

which is the singular-value decomposition (SVD). Performing very-large-scale SVDs is nontrivial, but has recently been performed on terabyte-sized matrices using HPC systems with techniques from randomized linear algebra.[4] Moving forward, convolutional-based methods (for example, convolutional non-negative matrix factorizations [NMF], which can result in interpretable "parts-based" decomposition) allow for the simultaneous extraction of spatiotemporal basis. Convolution-based methods can require significant computational resources and so implementations on GPUs and HPC architectures could greatly reduce compute time.

A complementary approach to dimensionality reduction for understanding spatiotemporal structure of neural populations is to use temporally directed analysis methods to infer causal influences among neurons. Such "functional connectomes" explicitly represent the influences between individual physical recordings and are thus potentially more interpretable than PCA. Open challenges in the application of such methods (such as generalized vector autoregressive models) to large-scale neurophysiology datasets include improving scalability of implementations, increasing the sparsity of estimated connectivity while retaining temporal contiguity, scaling of sparse identification of nonlinear dynamical systems to large dimensions (neurons), and accounting for the nonstationarity of data obtained in complex behavioral experiments. Other scalable methods to identify recurring spatiotemporal activity patterns in neuronal data (such as recurrent neural networks) should also be investigated.

Analysis of time-series data, while ubiquitous across the physical sciences and central to neural recordings, has not been performed in many biological fields. For example, as the cost of genetic sequencing and proteomics continues to fall, it is likely that biologists (and clinicians) will not only collect more samples from larger populations, but also more samples from the same individual over time. Thus, continued investigation in data analytics for time-series data, coupled with implementations in HPC systems (for example, to distribute the optimization calculation underlying many of these methods) are likely to become important for interpreting biomedical data in the near future.

Linking sensations, brains, and behaviors

In natural environments, animals process many high-dimensional sensory signals that are co-modulated over diverse time scales to produce complex sequences of behaviors toward achieving goals. Neuroscientists traditionally use hand-engineered features to characterize sensory and behavioral events, but the inherent arbitrariness of this selection impedes insight into neural coding principles. Sparse coding methods extract features from naturalistic sensory datasets (primarily visual and auditory) that can provide a principled account of response properties in primary sensory areas.[5] Characterizing the joint high-order statistics of natural sensory signals is challenging, as it requires the analysis of large amounts of multimodal data collected over long periods.

Neuronal activity is modulated on rapid time scales by many factors, including attention, reward, variation in arousal, and so on. Disentangling the contributions of different factors requires causal perturbations of neural activity during task engagement. Methods for rapid, cell-type-specific manipulations of neural activity in awake, behaving animals provide the capability to perform high-resolution closed-loop experiments. To be effective, this requires incoming brain signals to be processed in real time and analyzed to identify specific patterns, and then to trigger the manipulation in milliseconds (<10 ms), necessitating sufficient computational power and fast interconnects. Modern approaches to distributed computation in local computer clusters can likely solve current problems, but scaling to future data volumes might be challenging. Alternatively, FPGAs are ideally suited for real-time processing of massive datastreams, but programming such hardware is outside the purview of most neuroscientists.

Synthesis through simulations

While simulations have long been a tool of neuroscience, until recently most have been downscaled in size. Downscaled networks can preserve first-order statistics of neuronal activity (means) but higher-order statistics (cross-correlations) are generally not preserved. Mesoscale measures of activity, which are key tools for understanding the human brain, are driven by the fluctuations of neuronal populations, which are dominated by correlations. Thus, if a simulation is inaccurate in the second-order statistics of the microscopic activity, predictions of the mesoscopic activity obtained by forward modeling might be misleading.

HPC systems have recently enabled researchers to construct anatomically detailed models of local cortical circuits and perform full-scale simulations of neurons with all their synapses.[6] While work on cortical microcircuits is progressing, current implementations have fundamental shortcomings. First, many brain functions are distributed over several areas, and thus cannot be understood by studying an isolated microcircuit. Second, each neuron receives about half of its excitatory inputs from distant sources; thus, isolated models of cortical microcircuits are severely underconstrained. These challenges might be addressed by increasing the scale of neural circuit simulations—a task well-suited to HPC.

While detailed biophysical simulations are important, other approaches have targeted more abstract simulations with less biological fidelity.[7] These modeling approaches have different goals: "bottom-up" models aim to emulate high-level phenomena by constraining low-level parameters, while "top-down" models aim to replicate specific computations in a region. Linking the biophysical reality of bottom-up models with the well-defined computations of top-down models could reveal the biophysical mechanisms of neural computations. More effort in this direction is required.

A central challenge facing very-large-scale neural circuit simulations is understanding how best to evaluate the quality of results of models underconstrained by relatively limited amounts of experimental data. Conversely, it should be possible to examine the accuracy of a data-analysis algorithm (such as spike sorting and functional connectivity estimation) on data from simulations for which the ground truth is known. Techniques for understanding the uncertainty of model outputs (uncertainty quantification) and overall sensitivity of models relative to input parameterization are areas that need to be further explored.

Now that major obstacles in memory usage have been removed, reduction of the simulation time becomes the relevant target. Next-generation supercomputers (the so-called exascale systems) will be able to represent major parts of the human brain at microscopic resolution, and could be used to make predictions of the effects of pharmaceuticals testable in humans with mesoscale measurements.

THERE IS A GROWING REQUIREMENT FOR HPC ARCHITECTURES TO BE LESS SIMULATION FOCUSED AND MORE DATA INTENSIVE.

FROM BIOLOGICAL BRAINS TO DIGITAL BRAINS AND BACK

Exponential increases in computational capabilities have fueled the establishment of simulation as the third leg of modern science (complementing classic theory and experiments). Importantly, because Moore's law is slowing down, HPC has found itself at a crossroads, as illustrated by the National Strategic Computing Initiative (NSCI). The challenge of making HPC systems bigger and faster is joined by a change in their desired use.

Specifically, simulations have been the driving application behind HPC's growth, and development of these systems has been focused on their requirements. However, there is a growing requirement for HPC architectures to be less simulation focused (higher flops) and more data intensive (higher-performance I/O bandwidth to memory and between nodes).[8] We expect that neuroscience, with its special demands on the balance between memory access and compute power, will further drive this shift in supercomputing.

Many neuroscientists are envisioning *interactive supercomputing*—using a supercomputer like a superworkstation for exploring large datasets. This requires a supercomputer to be managed more like a telescope: individual research groups would be assigned time, as opposed to a scheme that executes a job when it optimally fits on the system. The price HPC centers will pay for this is a decline in the overall system utilization metric, but the benefit will be a much broader scientific user base.

As Moore's law draws to end, alternatives to von Neumann's model of

computing are being explored. Neuromorphic hardware is being developed, which might enable large-scale brain models and potentially advance more brain-like computing systems. Hardware implementations must make engineering tradeoffs to achieve specific computational advantages.

For instance, IBM's TrueNorth is a specialized chip achieving large-scale computation at low power, but with relatively restricted connectivity, low-precision synapses, and no on-chip learning. In contrast, the Spinnaker system modifies commercially available ARM cores that are directly programmable, and thus permits flexible implementations of neural circuits, albeit without energy savings. Finally, the BrainScales system uses analog approaches for accelerating simulations of neural circuits to study long-time scale processes like learning and development, but these approaches are nontrivial to implement and challenging to program.

Innovations in neuromorphic hardware might inspire new technology for classical systems. Perhaps design concepts that enable low-power computation in the brain might be used in next-generation supercomputing facilities to satiate their voracious power consumption. Additionally, computer scientists might find continued inspiration for stable, adaptive computing systems from the brain's synaptic and cellular learning processes.

Collaboration between brain scientists and computer scientists has a long history. Shortly before his death, von Neumann started writing about the similarities and differences between computers and brains.[9] Today, many burgeoning collaborations benefit both fields. The droves of data produced by the world's neuroscience initiatives could be an application area that ushers in a new age for HPC focused on experimental and observational data. Drawing inspiration from brain circuitry has enabled the development of low-power computing chips. Modern computer systems and algorithms are able to leverage massive datasets to train deep neural networks to effectively solve many problems for which progress had essentially plateaued. Together, these collaborative ventures have revived interest in artificial intelligence—the true nexus of the two fields.

There are many opportunities for neuroscience to benefit from HPC and computer science. Co-location of open neuroscience data repositories with HPC hardware will greatly support neuroscience efforts to reveal universal design features of species' brains and to understand what makes each individual unique—a central concept of precision medicine. Relating the structural and functional connectomes is necessary to deepen biophysical understanding of neural computations by mapping anatomical wiring diagrams to functional properties. Quantitative methods for understanding structure–function relationships is a ubiquitous problem in many fields (structural biology and material science, for example). In this context, developing a theoretical framework for understanding learning in deep neural networks (where we have precise knowledge of the structural connectivity, the activation of every unit, the objective function, input statistics, and learning dynamics) is a prerequisite to a normative "theory of brain" that links structure and function (this observation has been made by Stanford University physicist Surya Ganguli, among others). However, we speculate that a "theory of brain" would require much more than deep learning and would likely build on concepts from nonequilibrium statistical mechanics, information theory, optimal control and decision theory, (deep) learning theory, sparse coding, Bayesian inference, and sensor fusion.

We believe that addressing the challenges described here would create infrastructure to enable the neuroscience community to utilize HPC systems, and would provide a prototype for long-term strategic engagements between HPC centers and other scientific communities in the age of data-driven discovery. This would impact scientific return from major federal investments across multiple initiatives—immediate and sustained investment is required.

REFERENCES

1. J.W. Lichtman, H. Pfister, and N. Shavit, "The Big Data Challenges of Connectomics," *Nature Neuroscience*, vol. 17, no. 11, 2014, pp. 1448–1454.
2. P. Gao and S. Ganguli, "On Simplicity and Complexity in the Brave New World of Large-Scale Neuroscience," *Current Opinion in Neurobiology*, vol. 32, 2015, pp. 148–155.
3. M. Januszewski et al., "Flood-Filling Networks," arXiv:1611.00421, 2016; https://arxiv.org/abs/1611.00421.
4. A. Gittens et al., "Matrix Factorization at Scale: A Comparison of Scientific Data Analytics in Spark and C+MPI Using Three Case Studies," arXiv:1607.01335, 2016; https://arxiv

ABOUT THE AUTHORS

KRISTOFER E. BOUCHARD is principal investigator of the Neural Systems and Engineering Lab at Lawrence Berkeley National Laboratory (LBNL) and the Helen Wills Neuroscience Institute at UC Berkeley. He received a PhD in systems neuroscience from UC San Francisco. Contact him at kebouchard@lbl.gov.

JAMES B. AIMONE is a principal member of technical staff in the Center for Computing Research at Sandia National Laboratories. He received a PhD in neurosciences from UC San Diego. Contact him at jbaimon@sandia.gov.

MIYOUNG CHUN is executive vice president of science programs at the Kavli Foundation. She received a PhD in molecular genetics from the Ohio State University. Contact her at chun@kavlifoundation.org.

THOMAS DEAN is a research scientist at Google and teaches computational neuroscience at Stanford University. He received a PhD in computer science from Yale University. Contact him at tld@google.com.

MICHAEL DENKER is a research scientist at the Institute of Neuroscience and Medicine (INM-6), Computational and Systems Neuroscience at the Jülich Research Center. He received a PhD in biology from the Free University of Berlin. Contact him at m.denker@fz-juelich.de.

MARKUS DIESMANN is director of the Institute of Neuroscience and Medicine (INM-6), Computational and Systems Neuroscience, and director of the Institute for Advanced Simulation (IAS-6), Theoretical Neuroscience at the Jülich Research Center. He is also a professor of computational neuroscience at RWTH Aachen University. He received a PhD in physics from Ruhr University Bochum. Contact him at diesmann@fz-juelich.de.

DAVID D. DONOFRIO leads the Computer Architecture Group at LBNL. He received a bachelor's degree in computer engineering from Virginia Tech. Contact him at ddonofrio@lbl.gov.

LOREN M. FRANK is a professor in the Department of Physiology at UC San Francisco and an investigator at the Howard Hughes Medical Institute. He received a PhD in systems neuroscience from MIT. Contact him at loren@phy.ucsf.edu.

NARAYANAN ("BOBBY") KASTHURI is a neuroscience researcher at Argonne National Labs and an assistant professor in the Department of Neurobiology at the University of Chicago. He received a PhD in neurophysiology from Oxford University (Rhodes Scholar). Contact him at bobbykasthuri@anl.gov.

CHRISTOF KOCH is the president and chief scientific officer at the Allen Institute for Brain Science. He received a PhD in physics from the Max Planck Institute for Biological Cybernetics. Contact him at christofk@alleninstitute.org.

OLIVER RÜBEL is a research scientist in the Computational Research Division at LBNL. He received a PhD in computer science from the University of Kaiserslautern. Contact him at oruebel@lbl.gov.

HORST D. SIMON is deputy director and chief research officer at LBNL. He received a PhD in mathematics from UC Berkeley. Contact him at hdsimon@lbl.gov.

FRIEDRICH T. SOMMER is an adjunct professor at the Redwood Center for Theoretical Neuroscience and the Helen Wills Neuroscience Institute at UC Berkeley. He received a PhD in physics from the universities of Düsseldorf and Tübingen. Contact him at fsommer@berkeley.edu.

PRABHAT leads the data and analytics services group at the National Energy Research Scientific Computing Center at LBNL. He received an MS in computer science from Brown University. Contact him at prabhat@lbl.gov.

.org/abs/1607.01335.

5. B.A. Olshausen and D.J. Field, "Emergence of Simple-Cell Receptive Field Properties by Learning a Sparse Code for Natural Images," *Nature*, vol. 381, no. 6583, 1996, pp. 607–609.
6. S. Kunkel et al., "Spiking Network Simulation Code for Petascale Computers," *Front Neuroinform*, vol. 8, 2014, p. 78.
7. J.J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *Proc. Nat'l Academy of Sciences USA*, vol. 79, no. 8, 1982, pp. 2554–2558.
8. E. Bethel et al., "Management, Analysis, and Visualization of Experimental and Observational Data—The Convergence of Data and Computing," *Proc. IEEE 12th Int'l Conf. e-Science*, 2016; doi: 10.1109/eScience.2016.7870902.
9. J. von Neumann, *The Computer and the Brain*, Yale Univ. Press, 1958.

RESEARCH FEATURE

What Are We Missing When Testing Our Android Apps?

Konstantin Rubinov and Luciano Baresi, Politecnico di Milano

Android's broad adoption drives the development of millions of new apps. Apps on this OS are not just trivial games; many of them handle sensitive information, exhibit complex structure, and require high reliability and trustworthiness. The authors discuss the problem of testing Android apps—the results achieved with current approaches, and what is still missing and requires fresh solutions.

Today's mobile apps are not merely cheap addictive games or a means to interact with our different social networks. These apps enable mobile users to carry out complex tasks that handle sensitive data and require high reliability and trustworthiness (for example, for managing our bank accounts or making our travel reservations). These activities rely on quality apps; so the code-and-fix approach, with frequent updates and changes, is no longer sufficient. Effective quality assessment requires proper planning, and apps must be carefully tested before release. Some peculiar characteristics of devices, operating systems, and the apps themselves necessitate new approaches to mobile software testing. To avoid ambiguity, we focused on the problem of testing Android applications, which are run on 80 percent of devices. In addition, there are many more readily-available testing solutions for Android than for iOS.

A vast majority of Android apps are written in Java, and thus some might consider reusing well-known solutions developed for testing software written in this language. Examples of these include JUnit (http://junit.org) for implementing and running unit tests, Findbugs (http://findbugs.sourceforge.net) and Evosuite (www.evosuite.org) for automated testing and static analyses, Randoop (https://randoop.github.io/randoop) for automated random testing, and MuJava (http://cs.gmu.edu/~offutt/mujava) for mutation testing.

In addition, given the prominent role of the graphical user interface of these apps, some might consider borrowing ideas and solutions from testing web- and GUI-based applications. Because many apps are clients of more complex systems, the solutions for testing distributed systems might also be useful. We are not suggesting these solutions should be disregarded, but we aim to provide a comprehensive analysis of the problem of testing Android applications by clearly summarizing what has already been done and by identifying what is missing.

A thorough study of the current practice revealed that the research has concentrated so far on several distinct areas: GUI model definition and extraction for exploration and testing, test input generation, capture-and-replay

0018-9162/18/$33.00 © 2018 IEEE

solutions, data-flow analysis and symbolic execution, and race detection. Overall this research is based on legacy solutions developed in other contexts that, to a large extent, combine existing solutions into hybrid approaches. Although such solutions solve part of the testing problem, more work is still needed to properly test how apps support multiple diverse devices, how they exploit fragments (https://developer.android.com/guide/components/fragments.html), and how they use the indirect and implicit mechanisms for invoking activities (https://developer.android.com/guide/components/activities.html). Furthermore, testing context-awareness capabilities and the possible interactions with multiple sensors requires additional effort.

ANDROID IN A NUTSHELL

Mobile apps occupy a special niche in today's software landscape. Their architecture and design combine and extend concepts inherited from multiple application domains, including embedded, web-based, distributed, and desktop software. Mobile apps are instances of reactive, non-terminating software. They provide adaptive GUIs and share dynamic resource management with their operating system. In addition, apps can embody both native and web content and be context-aware, meaning that they can react to changes in the environment.

More specifically, the Android framework defines an application model for structuring apps and provides different types of components for building them: activities and fragments for user interaction through graphical interfaces, services for background processing, content providers for data management, and broadcast receivers for system-wide messages. App components are implemented by extending the Java classes provided by the Android framework and must be registered in the app's manifest to become visible to the OS. The framework also provides an Inter-Component Communication (ICC) messaging mechanism, based on intents, for the communication between components. Apps can specify explicit and implicit intents to request an action from another app component or components of the other apps on the same device (subject to permissions and constraints). Explicit intents identify the target component by name, while implicit ones declare a general action to be performed by the unknown target. The OS finds the proper match for the declared action and mediates the actual interaction between components.

SIMILAR TO WEB APPS COMPOSED OF PAGES, ANDROID APPS ARE ORGANIZED AROUND SCREENS, CALLED ACTIVITIES.

Similar to web apps composed of pages, Android apps are organized around screens, called activities. Each activity defines both the graphical elements the user can interact with and the business logic executed upon interaction with these elements (such as a click of a button). Fragments enable the modularization of activities because they define self-contained assemblies of components and can be instantiated dynamically within activities, tied to the different device configurations, user interactions, and contexts. The OS controls the lifecycle of activities and fragments according to its policies (that is, it can destroy a background component if resources are needed), which makes app logic tightly coupled with that of the OS.

Android realizes a single-UI–threaded, event-based model: the main thread of the app processes the events on UI elements. This model simplifies the synchronization between the different concurrent processes on the device: it aligns and sequentially executes user interactions, responses to user actions (such as communications with external services), context changes (including, for example, new network availability), and sensor management processes. The downside of the model is that a UI thread can only allow brief computations to guarantee the responsiveness of the UI. This forces the developer to design multi-threaded apps in which the main thread is used almost exclusively for spawning new threads to carry out computations in parallel.

CURRENT PRACTICE

The UI and business logic of Android apps are tightly integrated by design

and difficult to isolate, therefore the most convenient way to test them is through the direct manipulation of the UI. This method is currently adopted by the majority of the manual and automated testing approaches. In addition, a number of static and dynamic analysis techniques facilitate automated testing. These include app-specific taint analysis, race detection, and symbolic execution, whereas capture-and-replay approaches naturally integrate the exploration of UI and partial contextual input data.

> PROGRAM ANALYSIS IS BECOMING INCREASINGLY MORE RELEVANT AS APPS GROW IN SIZE AND COMPLEXITY.

Functional testing and direct UI manipulation

Functional testing is largely performed by directly interacting with the UI to check various functional properties. These can include, for example, visualized data and charts, data processing, storage, application flow across activities, expected sounds, settings, and time zones.

Since apps are usually devoid of specifications, developers must derive test cases manually by relying only on informal specifications. The generation of test cases is supported by several testing frameworks, like Espresso, Robotium, Appium, and others. In these frameworks, test case generation is a combination of unit test case design à la JUnit and GUI-based web application testing à la Selenium (http://seleniumhq.org). Testing amounts to developing scripts for exercising the basic interactions with the app's UI elements, such as clicking on a button, entering text in a text field, or checking the status of an element (for example, if a button is visible or not).

Various research solutions address the issue of automated test case generation relying on direct UI manipulation. To adequately exercise app behavior, these solutions extend existing desktop GUI testing approaches and derive test cases predominantly from the GUI models.[1] However, for realistic app testing, GUI exploration must be combined with the synthesis of system events (for example, no network available) and contextual inputs (for example, current GPS location). Nevertheless, complex GUI interactions are not always amenable to automated execution or they can become inconsistent across different devices and vendor-customized Android versions.

More importantly, these solutions lack comprehensive test oracles. To date, few approaches generate test oracles, for instance, for the detection of user-interaction issues,[2] whereas the majority of automated solutions rely on runtime exceptions to detect faults. Since these exceptions only constitute a subset of the possible fault manifestations, many faults could remain undetected.

Program analysis and Android framework

Program analysis is becoming increasingly more relevant as apps grow in size and complexity, because it facilitates a multitude of tasks that help assure the quality of apps. Examples include code coverage analysis, mutation analysis, symbolic execution, automated test case generation, application structure modeling, memory leak and race condition detection, and network usage profiling.

Historically, analyses for Android apps have largely been developed for malware detection and security analysis, rather than for testing or test automation. However, novel testing approaches incorporate static control and data flow analyses for deriving test cases automatically. For instance, A3E (http://spruce.cs.ucr.edu/a3e) relies on static analysis to construct the activity transition graph of an app and uses it at runtime for automated exploration and testing.

The major challenge in transplanting existing analyses to Android stems from the structural dependence of the apps on the Android framework (the OS's top layer that provides services and resources for managing and executing apps). This necessitates modeling the framework to faithfully model the apps themselves. In particular, one has to model the app's lifecycle management, a mechanism that creates complex control- and data-flows to handle the cases when the OS or

the user interrupts the app. Consequently, one should extend the analyses for working with the ICC messaging mechanism and consider that the required, external app components might not always be available.

The problem of modeling the Android framework has been tackled by a number of approaches. Stubdroid[3] models the Android runtime library to enable precise data-flow analysis. Yang and colleagues[4] propose a new program representation to model event callbacks. Symbolic and concolic exploration approaches rely on models of the framework for exercising apps.[5] The problem has also been tackled by data race and memory leak detection approaches.[6] However, none of these approaches faithfully models dynamically instantiated app components, therefore precluding the testing and analysis of fragment-based apps.

OPEN ISSUES

Despite the many approaches that adapt existing solutions to the new context, there are still some app-specific dimensions in testing Android applications that require additional effort and new solutions.

Multiple devices

The first challenge refers to so-called "universal" apps, that is, a single codebase of an app is supposed to run properly on any device. Accommodating this requirement to the great variety of existing Android devices (in terms of screen sizes/resolutions and available resources) is a difficult problem and requires that many different devices be available for app development and testing. Moreover, the manifold screen sizes could force developers to design completely different UIs; that is, different layouts of the (possibly different) graphical elements. This situation calls for the use of fragments to structure and modularize reusable assemblies of the user interface, but adds further complexity. Fragments are active, independent entities that must be managed properly with respect to the activities that contain them.

Android apps also face the fragmentation of both available devices and versions of the OS. The former is caused by the variety of physical components available on the different devices. The latter stems from the many coexisting versions of the OS and from the use of the Android Support Library to make old versions support some of the latest features.

Numerous public and private cloud testing infrastructures have been established for large-scale mobile app testing (popular examples include Xamarin [www.xamarin.com/test-cloud], and PerfectoMobile [www.perfectomobile.com/continuous-quality-lab]). Nevertheless, Choudhary and colleagues[1] highlight the relevance of the fragmentation issue since Android developers have to constantly make an effort to overhaul app compatibility with older versions of Android and many different devices. An example of a simple problem is when graphics created for high-resolution screens are not displayed properly on mobile devices with extra-high or low screen resolutions.

There is a combinatorially intractable number of possible device and OS version configurations. Moreover, the number can be magnified by the possible interactions with different versions of external apps on the same device. To mitigate the problem, several approaches have started to address the optimal combinatorial selection of mobile devices for testing. Vilkomir and Amstutz[7] highlight the lack of coverage metrics and a conflict in widespread recommendations: testing on most popular devices vs. testing on most diverse devices. The authors propose the use of combinatorial methods to merge the two recommendations. Khalid and colleagues[8] propose the prioritization of devices that have the highest impact on app ratings. By analyzing user reviews, the authors found that "on average, 33 percent of all devices account for 80 percent of reviews given to free game apps," and a similar trend applies to paid apps. Finally, Lu and colleagues[9] propose the prioritization of devices by mining usage data such as browsing time.

These approaches focus on external or indirect app characteristics, and might not apply when allocating resources to test entirely new apps. These approaches also implicitly accept the risk of poor software quality on less-popular devices. In contrast, we argue that instead of making these trade-offs, the focus must shift toward understanding the internal structure of the apps to optimally select devices for testing. This process can be supported by the generation of differential test oracles for checking the differences in app behavior and in the visualization of the UI that arise across the configuration space.

Differential oracles would be especially relevant for testing the different layouts of the UI based on the dynamic instantiation of components (fragments). Configuration files give an indication of data placement, but

the actual content might not always be known before runtime. Often the layout must be changed at runtime to accommodate the content with respect to the used devices. Consequently, testing on actual devices is relevant, but without comprehensive test oracles it would require considerable effort and entail high costs. Novel techniques attempt to address the issue by comparing screenshots for localized UI testing across devices (for instance, the Facebook screenshot-tests-for-android tool http://facebook.github.io/screenshot-tests-for-android), however test oracle generation approaches have yet to be developed.

Lifecycle and system interactions

Android manages the lifecycle of apps to help respond to context changes and facilitate dynamic resource management. Context changes can interrupt the execution of an app because of an incoming call, low battery, or a rotation of the device. These are examples of highly common events that can happen in different combinations and orders. It is then imperative that the lifecycle of apps be tested to assess their behavior with respect to these events and to their interactions with the system.

Several known techniques incorporate system events into automated app testing. The Android testing framework provides Monkey for the random synthesis of UI and system events. Building on Monkey, Dynodroid[10] synthesizes relevant UI and system events that include incoming text messages, audio focus requests, changes in the GPS coordinates, or modifications to the battery level. The approach identifies registered event listeners and uses them to create "relevant" events. Finally, Appdoctor[11] tests apps against user actions, lifecycle events, and the subset of system events related to changes in storage capabilities and in connectivity.

Despite the existence of program analyses (especially for resource management) that explicitly address the issues triggered by the lifecycle management of app components, only a few approaches focus on testing how apps react to changes in their lifecycle state. For instance, one may be interested in assessing the behavior of an app that is moved from being in the foreground, and thus being running, to the background, and thus becoming suspended or stopped. Approaches like, for example, Quantum[2] target checking an app's adaptation to changes in its lifecycle state, but they focus on sequences of neutral lifecycle events (for example, pausing and resuming an app by simulating the notification of an incoming text message). Furthermore, lifecycle state changes are currently not supported by the standard Android testing infrastructure provided by Google.

The dynamic instantiation of components (fragments) adds another layer to lifecycle-related changes. Changes in the lifecycle of fragments depend on system events and user interactions, but they also depend on the state of their container activities. Yet no approach tests the changes in the lifecycle state of fragments.

Even though lifecycle management is properly supported by the Android framework and is integral to the development of Android apps, many lifecycle-related problems escape testing and can crash the apps. New self-healing solutions for Android apps[12] are now being proposed in an attempt to cope with these issues.

Context-awareness and sensor data

Mobile apps are context-aware in a unique way because they can integrate contextual data from a multitude of diverse sensors. For instance, modern smart alarms or fitness apps continuously collect data from the device's GPS antenna, microphone, and accelerometer, interact with its NFC reader, vibrator, and flashlight, and possibly dialogue over Bluetooth with external devices (such as with a smart watch or a wearable activity tracker, for heart rate monitoring). Apps are also routinely exposed to network and connectivity changes, which can be seen as another type of context awareness.

Testing context awareness requires that realistic and extreme sensor data be synthesized. These data, along with the event patterns behind them, are complex, can be imprecise (because of disturbances or faulty sensors), and depend on multiple parameters. The task is even more complex since apps are governed through touch screens and complex gestures, and the number of possible combinations of diverse contextual data and user inputs (gestures) tends to explode.

Existing frameworks have strong limitations when dealing with contextual data (provided by the device's GPS antenna, accelerometer, and gyroscope), rotations of the device, and changes in network connectivity. Single system events can be triggered manually by using the Android emulator (https://developer.android.com/studio/run/emulator.html), whereas

low-level interfaces like adb shell (https://developer.android.com/studio/command-line/adb.html) and hardware emulators can inject events automatically. However, this solution is cumbersome and is only available in the simulated environment.

Several network conditioning tools exist and can be used to manually control and vary the network conditions (for example, Network Link Conditioner for macOS or Augmented Traffic Control http://facebook.github.io/augmented-traffic-control). They allow one to emulate packet loss and network latency while testing apps on real devices. The Android emulator also provides limited network-tuning capabilities, and the tool support for network-related testing is improving. One can manually specify test cases for web-specific app components, such as WebViews (https://developer.android.com/guide/webapps/webview.html), to simulate varying network conditions. Automated systematic testing approaches, as well as solutions that help specify realistic network change patterns, are to be developed.

Several testing approaches support the systematic and automated synthesis of complex contextual data, but most of these approaches use either simplified single events or sets of user-generated events obtained by means of capture-and-replay tools. As for the former approach, Amalfitano and colleagues[13] present different proposals for developing techniques that leverage patterns of system events such as "network instability" or "loss and recovery of GPS signal." The authors show that although the support the Android framework offers for event injection is limited, the synthesis of contextual events is technically feasible. The latter requires that capture-and-replay tools be able to generalize observed context-dependent events to different devices with different configurations. Another general approach is the simulation of generally occurring conditions: for example, Crashscope simulates adverse conditions by disabling network connectivity or setting unexpected sensor values.[14]

Finally, the manual definition of sensor events is not trivial. A few approaches have been proposed to simplify the definition of test cases, enriched with manually supplied contextual data. For example, Griebe and colleagues[15,16] propose to either use annotated UML activity diagrams to specify application segments that depend on contextual data and to derive test cases automatically, or to specify sensor data in test cases by means of a domain-specific language called Gherkin. These test cases can mimic device movement, and embed data from a variety of sensors (for example, the accelerometer and the sensors in charge of measuring magnetic field, proximity, ambient temperature, and network availability). The authors highlight the need for easy-to-manipulate human-readable representations of sensor data.

Dynamic UIs

The apps that embed dynamically created UI elements require special attention. Their structure and behavior (application flow) can be modified dynamically in response to user interactions and to changes in the device's configuration (such as a change in orientation). Moreover, these structural and behavioral changes vary on the different devices since larger devices may have a starkly different UI from that of the smaller ones. Modeling and testing dynamic UIs is a recent challenge that has not been tackled yet. Fragments are essential to implement dynamic and versatile UIs (for example most tabbed interfaces are based on fragments), but automated testing and analysis approaches do not model them and their respective lifecycle.

The problem of testing apps with fragments stems from their versatility. Fragments are not simple composable containers of graphical elements, but they are active entities whose lifecycle oversees the behavior of contained elements. Fragments must be contained in activities, the same fragments can be reused in different activities, and they can also be embedded (composed) in different ways given the device (screen size) on which the app is run. The fragments in an activity can interact, and they must interact with the host activity.

The problem is similar to testing apps on different devices. Existing techniques focus on checking the behavior and appearance of an app on a given device; no technique examines the differences among UIs that may differ significantly, especially when the UIs are based on fragments.

Complex UI interactions

Many effective techniques and models can navigate UIs using basic, unimodal input actions like point and click, scroll, and swipe, whereas the support to multimodal inputs is still problematic. Event synthesis and oracle generation are challenging for user interactions that involve multi-touch

gestures, and become even more complex when they are combined with sensor data (gyroscope, acceleration, and so on) and voice commands. For example, many apps use annotated charts to visualize data and allow the user to manipulate them by means of multi-touch gestures, or require that the user draws a particular pattern on the screen. The generation of tests in these two cases would be cumbersome.

Another problem is the correctness of visual and kinesthetic feedback. Apps extensively rely on animated transitions and dynamic visual feedback (namely that a user input triggers animation of some elements). No standard automated mechanism exists to verify the correctness of this kind of feedback.

Capture-and-replay tools like RERAN (www.androidreran.com) and VALERA (http://spruce.cs.ucr.edu/valera) can help, but they are limited to the generation of test cases for single, individual device configurations. To be useful, identified test cases must be generalizable to a large number of devices and cover multiple configurations (display size, pixel density, orientation, version of the operating system, and so on). Currently, captured test cases with custom gestures are not parametrizable for different device configurations. Initial research in this direction only attempts to make captured test oracles (screenshots) reusable across devices.

Moreover, realistic gesture-based interactions must take into account the non-uniformity of human gestures. Hesenius and colleagues[17] formalize the description of virtual, parametric touch gestures that can be instantiated properly in the different test cases. This allows one to extrapolate the coordinates and scale the gestures for different configurations. However this approach is not automated and lacks oracles to check the success of applied gestures: it can only check the results indirectly, for example, by observing a state change in some UI elements (such as when a button becomes visible) as side-effect of mimicking a correct gesture.

Several commercial tools analyze user experience by recording how the users interact with apps. In particular, one could capture and analyze touch heatmaps, for instance, with Appsee (https://www.appsee.com/features/touch-heatmaps) or Appanalytics (http://appanalytics.io). This type of analysis detects bugs related to obstructions in the user interface or, for instance, user gestures that do not trigger a response in the app. An example of such a bug is when the user consistently taps on an UI element that looks like a button, but it is actually an image or a label.

Inter-app interactions

Apps extensively rely on other apps and services for extending their functionality and capabilities by means of Inter-Component Communication (ICC). Any external app the user selects can provide its services and carry out some tasks on behalf of the original app. For example, apps commonly use other "standard" apps for taking pictures or scanning QR codes. In this case, Android allows an activity to ask the OS—through implicit intents—to interact with another activity to carry out a given action (for example, to visualize a webpage). The activities an app would like to interact with are usually not known a priori and are external to the app itself. As a consequence, the same app when executed on different devices may produce different outputs, and different results, because of the external components it interacts with. ICC challenges integration testing by presenting a combinatorially intractable number of possible app interactions.

Standard testing frameworks, like Espresso (https://google.github.io/android-testing-support-library), allow one to specify the intents for interacting with external apps and to mock their responses. System level testing tools, like UI Automator, allow the user to specify test cases for navigating across apps and thus explore the possible combinations. However, these test cases are laborious to design, and are slow and flaky during execution.

Apps that respond to external intents can benefit from intent fuzzing to assess their reactions to specifically crafted intents.[18] Fuzzers check the robustness of apps, rather than covering their behavior or exercising combinations of inter-app dependences. Recent research introduces data-flow tracing across app sets. FlowDroid (https://blogs.uni-paderborn.de/sse/tools/flowdroid), which is the state-of-the-art tool for static data-flow analysis for Android apps, has been extended to track the data-flow across apps.[19] These analyses can potentially act as starting point for automated test-case generation.

Another related challenge is the selection of all the relevant combinations of components for integration. Octeau and colleagues[20] tackle the combinatorial explosion of app combinations through ICC. A probabilistic model helps triage static analysis

results and eliminate combinations that have extremely low probability of occurrence. The model is trained using domain knowledge and is then applied on top of static analysis results. The authors show that over 95.1 percent of potential combinations are associated with probability values below 0.01 and can thus be eliminated.

CONCLUSIONS

This paper shows that there has been significant progress in testing Android apps. On the other hand, it also claims that testing dynamically instantiated components is still overlooked by the existing techniques. These components require that one uses more complex models, extends existing structural analyses, and checks their behavior on many different device configurations.

Another interesting observation refers to the testing of context-awareness and complex user inputs. Despite the fact that apps are becoming increasingly more context-aware, testing this feature is still a laborious process with little or no automation. Similar concerns apply to testing apps that require complex input gestures. The automated generation of such test cases is still in its infancy. Finally, the fragmentation issue contributes substantially to the cost and scalability of the different solutions.

Even if this paper does not provide new solutions, it offers a comprehensive analysis of the state of the art and identifies a set of significant challenges. New, fresh solutions are required to address these challenges properly, whereas the advancement of artificial and virtual reality apps will likely magnify these challenges.

ABOUT THE AUTHORS

KONSTANTIN RUBINOV is an independent researcher. While performing the work reported in this article, he was a postdoctoral researcher at the Politecnico di Milano. His research interests include software engineering and the mobile domain with a particular focus on software testing and analysis. Rubinov received his PhD in computer science from the Università della Svizzera italiana (USI), Switzerland. Contact him at rubinov.konst@gmail.com and http://futurezoom.in.

LUCIANO BARESI is a full professor in the Dipartimento di Elettronica, Informazione e Bioingegneria at the Politecnico di Milano. His research interests include software engineering, particularly in pervasive, service-based, mobile, and self-adaptive software systems. Baresi received his PhD in computer science from Politecnico di Milano. Contact him at luciano.baresi@polimi.it.

ACKNOWLEDGMENTS

The work presented in this article has been partially supported by project PSC - Piattaforma dei servizi nel settore della giustizia civile (MIUR-SCN 00356).

REFERENCES

1. S.R. Choudhary, A. Gorla, and A. Orso, "Automated Test Input Generation for Android: Are We There Yet?," *Proc. 30th IEEE/ACM Int'l Conf. Automated Software Engineering* (ASE 15), 2015. pp. 429–440.
2. R. Zaeem, M. Prasad, and S. Khurshid, "Automated Generation of Oracles for Testing User-Interaction Features of Mobile Apps," *Proc. IEEE 7th Int'l Conf. Software Testing, Verification and Validation* (ICST 14), 2014, pp. 183–192.
3. S. Arzt and E. Bodden, "Stubdroid: Automatic Inference of Precise Data-Flow Summaries for the Android Framework," *Proc. 38th Int'l Conf. Software Engineering* (ICSE 16), 2016, pp. 725–735.
4. S. Yang, et al., "Static Control-Flow Analysis of User-Driven Callbacks in Android Applications," *Proc. IEEE/ACM 37th Int'l Conf. Software Engineering* (ICSE 15), 2015, pp. 89–99.
5. J. Jeon, et al., "Synthesizing Framework Models for Symbolic Execution," *Proc. 38th Int'l Conf. Software Engineering* (ICSE 16), 2016, pp. 156–167.
6. P. Bielik, V. Raychev, and M. Vechev, "Scalable Race Detection for Android Applications," *Proc. 2015 ACM SIGPLAN Int'l Conf. Object-Oriented Programming, Systems, Languages, and Applications* (OOPSLA 15), 2015, pp. 332–348.
7. S. Vilkomir and B. Amstutz, "Using Combinatorial Approaches for Testing Mobile Applications," *Proc. IEEE 7th Int'l Conf. Software Testing, Verification and Validation Workshops* (ICSTW 14), 2014, pp. 78–83.

8. H. Khalid, et al., "Prioritizing the Devices to Test Your App on: A Case Study of Android Game Apps," *Proc. 22nd ACM SIGSOFT Int'l Symposium on Foundations of Software Engineering* (FSE 14), 2014, pp. 610–620.
9. X. Lu, et al., "Prada: Prioritizing Android Devices for Apps by Mining Large-Scale Usage Data," *Proc. 38th Int'l Conf. Software Engineering* (ICSE 16), 2016, pp. 3–13.
10. A. Machiry, R. Tahiliani, and M. Naik, "Dynodroid: An Input Generation System for Android Apps," *Proc. 9th Joint Meeting on Foundations of Software Engineering* (ESEC/FSE 13), 2013, pp. 224–234.
11. G. Hu et al., "Efficiently, Effectively Detecting Mobile App Bugs with Appdoctor," *Proc. 9th Eur. Conf. Computer Systems* (EuroSys 14), 2014, pp. 18:1–18:15.
12. M.T. Azim, I. Neamtiu, and L.M. Marvel, "Towards Self-Healing Smartphone Software Via Automated Patching," *Proc. 29th ACM/IEEE Int'l Conf. Automated Software Engineering* (ASE 14), 2014, pp. 623–628.
13. D. Amalfitano, et al., "Considering Context Events in Event-Based Testing Of Mobile Applications," *Proc. IEEE 6th Int'l Conf. Software Testing, Verification and Validation Workshops* (ICSTW 13), 2013, pp. 126–133.
14. K. Moran, et al., "Automatically Discovering, Reporting and Reproducing Android Application Crashes," *Proc. IEEE Int'l Conf. Software Testing, Verification and Validation* (ICST 16), 2016, pp. 33-44.
15. T. Griebe and V. Gruhn, "A Model-Based Approach to Test Automation for Context-Aware Mobile Applications," *Proc. 29th Ann. ACM Symp. Applied Computing* (SAC 14), 2014, pp. 420–427.
16. T. Griebe, M. Hesenius, and V. Gruhn, "Towards Automated UI-Tests for Sensor-Based Mobile Applications," *Proc. 14th Int'l Conf. Intelligent Software Methodologies, Tools and Techniques* (SoMet 15), 2015, pp. 3-17.
17. M. Hesenius, et al., "Automating UI Tests for Mobile Applications with Formal Gesture Descriptions," *Proc. 16th Int'l Conf. Human-Computer Interaction with Mobile Devices & Services* (MobileHCI 14), 2014, pp. 213–222.
18. R. Sasnauskas and J. Regehr, "Intent Fuzzer: Crafting Intents of Death," *Proc. Joint Int'l Workshop on Dynamic Analysis (WODA) and Software and System Performance Testing, Debugging, and Analytics (PERTEA)* (WODA+PERTEA 2014), 2014, pp. 1–5.
19. W. Klieber, et al., "Android Taint Flow Analysis for App Sets," *Proc. 3rd ACM SIGPLAN International Workshop on the State of the Art in Java Program Analysis* (SOAP 14), 2014, pp. 1–6.
20. D. Octeau, et al., "Combining Static Analysis With Probabilistic Models to Enable Market-Scale Android Inter-Component Analysis," *Proc. 43rd Ann. ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (POPL 16), 2016, pp. 469–484.

SUBMIT TODAY

IEEE TRANSACTIONS ON BIG DATA

SCOPE

The *IEEE Transactions on Big Data* (*TBD*) publishes peer reviewed articles with big data as the main focus. The articles provide cross disciplinary innovative research ideas and applications results for big data including novel theory, algorithms and applications. Research areas for big data include, but are not restricted to, big data analytics, big data visualization, big data curation and management, big data semantics, big data infrastructure, big data standards, big data performance analyses, intelligence from big data, scientific discovery from big data security, privacy, and legal issues specific to big data. Applications of big data in the fields of endeavor where massive data is generated are of particular interest.

SUBSCRIBE AND SUBMIT

For more information on paper submission, featured articles, calls for papers, and subscription links visit:

www.computer.org/tbd

IEEE computer society | IEEE COMMUNICATIONS SOCIETY | IEEE Computational Intelligence Society | IEEE Sensors Council | IEEE Consumer Electronics Society | IEEE Signal Processing Society | Systems, Man, & Cybernetics Society

IEEE Systems Council | IEEE VTS | CSS | IEEE photonics SOCIETY | EMB | IEEE PES Power & Energy Society | IEEE Biometrics Council | IEEE

Weaponizing Twitter Litter: Abuse-Forming Networks and Social Media

Hal Berghel, University of Nevada, Las Vegas

Instead of liberating us from the biases of the educated among us, the Internet has saddled us with the biases of the unreasoned among us.

Many of us have serious reservations against creating user accounts with, or even using online services from, tech companies with onerous privacy policies. Google, for one, is especially aggressive harvesting personal data on users (www.google.com/policies/privacy). However, as Eli Pariser shows in his recent book *The Filter Bubble*,[1] we have more to fear from online services than invasions of privacy: many of these services also manipulate the online information spaces that shape our decisions. In a very real sense, Pariser anticipated the problems with fake news and partisan trolling that befell the 2016 US national elections.

It's easy enough to avoid using Google services like Gmail and its search engine. Protonmail (https://protonmail.com) and DuckDuckGo (https://duckduckgo.com), for example, are both viable, independent, privacy-respecting substitutes. In addition, there are large corporate offerings like Microsoft Outlook and Bing that appear to be less invasive of personal privacy than Google (www.privacy.microsoft.com/en-us/privacystatement). However, as Pariser demonstrates, there's no way to insulate ourselves from the censorship and rectified flow of information that takes place without our knowledge and consent. Even if we once accepted the premise that surrendering some privacy is the price we paid for free online services, it's gone too far: we're not only losing privacy to services we use, we're losing it to services we *avoid* because more and more of them are digitally interlocked through information sharing. Further, these same services are placing us in an information cocoon.

ABUSE-FORMING NETWORKS

(Robert) Metcalf's law[2] holds that the connectivity value (aka utility) of a network is proportional to the square of

0018-9162/18/$33.00 © 2018 IEEE

EDITOR **HAL BERGHEL**
University of Nevada, Las Vegas; hlb@computer.org

the number of nodes, n^2. It assumes that there's some measurable value that increases with the number of possible pairwise connections. David P. Reed[3] extends this reasoning to claim that there's also some measureable value to the number of potential groups that can form within a network, and that this number grows exponentially by the number of nodes, 2^n—what he calls group-forming networking. Andrew Odlyzko and Benjamin Tilly[4] seek to moderate Reed's formula by taking into account the law of diminishing returns within group formation, arguing that the value of the number of groups is $n\log(n)$.

All of these are plausible measures of some sort of value for networks,[2,5] but they miss one important factor that has become critical in the past few decades: the potential aggregate "cost" or "penalty" to the users/participants through the leaking of information about them, the surveillance of their daily lives and actions, the loss of their time due to unnecessary or unwanted distractions, the potential loss of their personal sovereignty and liberty, and the potential for their social manipulation by tyrants, demagogues, dictators, and other manifestations of the power elite.[6] I refer to this phenomenon as *abuse-forming networking*.

We take Reed's law as our starting point. As with Reed, we note that if group-forming networks are integrated, their aggregate value is proportional to the product of their individual values, $2^m \times 2^n \times \ldots$, that is, to the product of the number of power sets. Something very similar happens to the degree of possible abuse of the users/participants. This all starts from the fact that the individual connections are detectable—at least by those who provide the networking. While this in itself might seem harmless, after the revelations of Edward Snowden we all recognize that the potential for abuse goes way beyond the knowledge of who's connected to what network. In fact, this was well known long before Snowden was born.[7,8] As I've written, Snowden's real legacy is showing that many of our suspicions were justified.[9]

But the abuse just begins with the identifiability of the connections. Add to that the traffic metadata: how many times node i received traffic from nodes j and k, when and to whom node l communicated, how often nodes associated with event x communicated with nodes associated with organization y, and so on. This is the stuff of which the NSA's Section 215 bulk metadata collection program is made.[10] The number of different identifiable signatures from this metadata far exceeds the size of the number of groups that might share information.

Now let's add some object-level data. Modern integrated marketing firms collect thousands of individual pieces of information on all of us: first and last name, Social Security number, mother's maiden name, known associates, family history, birth/death dates, income history, credit history, current and past addresses, employers, driver's license and other ID numbers, email addresses, bank card data, transaction records from many merchants, name/type of pets, associated IP addresses, arrest record, voting registration information and party affiliation, potential inheritance, medical conditions/needs, time/date/duration/SMS routing information types and destinations of all electronic communications (telephony, email, faxes), computer and mobile device information (such as MAC addresses, serial numbers, OS versions, and browser versions), geolocation information (GPS coordinates), biometric data (including DNA data, voiceprints, and face images captured by satellites, drones, and surveillance cameras), cookies and contents from application caches, clickstream data, associated ISPs and telcos, what you watch on YouTube, Netflix, Amazon, and so on. This is just a partial list of the information routinely collected by businesses and doesn't include the much more invasive data collected by private and government security agencies, all of which can be used as "selectors" to search through yobibytes ($1{,}024^8$) of global, digital stored data on all of us. For more details, visit the websites of the Electronic Privacy Information Center (www.epic.org) and Privacy International (www.privacyinternational.org).

> Our online world has introduced two new forms of information corruption: source displacement and decontextualization.

My point is that if we consider the abuse of individual rights as related to the use of information about n individuals without their expressed permission, the potential for abuse derived from the combined object and meta-level data from the networks has to be significantly larger than 2^n. So let's estimate the upper bound at $2^n \times 2^k$, which is Reed's number of potential subgroups of network users times the power set of the number of different collections of attributes that can be defined over all groups of users. This has some intuitive justification, for 2^k is the number of all possible associations that that one might make

of all of the groups based on the individual data elements that correspond to the network members/users such as those mentioned in the preceding paragraph. That is, the latter set would include such subsets as all members with brown dogs, the set of all groups with at least one member with a brown dog and a subset of members whose name is Phil, the number of dog owners that subscribed to white supremacist literature, and so on. With k in the thousands (a realistic assumption), the ways in which the threads of association can be created is exceedingly large. In general the public isn't appraised of how, by whom, and for what purposes these threads are created, but it would be a mistake to underestimate their use by political organizations and operatives, intelligence and investigative agencies, law enforcement, criminal organizations, phishers, scammers, spammers, NGOs, marketing companies, and so forth—almost all of which instances are without the user's knowledge or consent.

Thus, Reed's law actually vastly understates the aggregate cost of abuse to network users in terms of the loss of privacy, misuse of personal data through identity theft or financial fraud (think Equifax hack), or downstream negative externalities from aggressive data harvesting. Such numbers, too, should be estimated, but under the rubric of abuse formation. We note that because the abuse is largely externally imposed rather than self-organizing, no constraining cognitive limit applies. Put simply, while there's a limit to the number of stable, cohesive groups with whom an individual can associate,[11] there's no limit to the number of external groups that can heap abuse on the individual by means of data mining automation. As an aside, an excellent history of the origins of the American version of the surveillance state can be found in Alfred W. McCoy's recent book, *Policing America's Empire*.[12]

BEYOND ABUSE BUILDING TO TRUTH FABRICATION

We extend our analysis to more subtle variations of online abuse, most importantly through the manipulation of the public through a constant stream of lies, prevarications, untruths, distortions, and so forth derived from fake news sources, trolls, propaganda channels, partisan media, and the like, for which we have no known protection and few working models.[13] According to Pariser, we all live in a filter bubble where information flow is carefully regulated by upstream manipulators under the banner of "personalization." He points out how the online "people-powered news" that many of us anticipated has been corrupted by merchants of faux news. Where 100 years ago major news sources "had a sense of ethics and public responsibility baked in however imperfectly, ... [today's] filter bubble does not."

> We have now entered the era of "lock-on" news feeds that nourish the addiction to misinformation.

Of course, censorship has been a constant companion to democracies—albeit in softer, self-induced forms than is found in dictatorships. But our online world has introduced two new forms of information corruption: source displacement and decontextualization. A century ago, fake news for the most part had to be orchestrated by corporate mass media within view of many critical eyes. The online revolution makes it possible for any individual or group of -ists (racists, anarchists, tribalists, and so on), no matter how small, to launch their own fake news service with an inexpensive computer and an Internet connection. While the disintermediation of the editor/publisher disempowered them to be sure, it also empowered delusionists, narcisissists, and sociopaths whose presence looms large over networked neophytes. Our educational system simply failed to anticipate this possibility and underemphasized the criticality of individual fact checking of all information sources.

Instead of a panel of professional journalists filtering news, we now have propagandists and prevaricators filling the role. Instead of liberating us from the biases of the reasonable among us, the Internet has saddled us with the biases of the unreasonable among us—at least when it comes to fake news. Pariser credits Columbia Law School professor and *New York Times* op-ed writer Tim Wu with a particularly apropos remark: "The rise of networking did not eliminate intermediaries, but rather changed who they are." Indeed, the disreputable Internet disintermediaries as a group form a paradigmatic case of an untrusted system—there's precious little journalism or scholarship involved. Unfortunately, too many minds are attentive to the vacuous content.

What's more, even the prevarications and propaganda are filtered and bundled for us—we don't get our misinformation unadulterated either. This activity falls under the category of "personalized information services," which is a euphemism for filtered information with manipulative potential. The key to this personalization lies in external forces pushing information toward you that primarily serves *their* interests. Whether it serves yours is of secondary concern.

The key to the successful spread of misinformation and false reporting is decontextualization, as it removes the contextual links necessary for confirmation/disconfirmation. You won't find extensive reference lists to

academic papers in tribalists' online resources: objectivity is an enemy to the tribe! If an online news service sees that you like chocolate-covered marshmallows, information about that will be delivered to you. The same goes for perceived interests in white nationalist or anarchist literature, partisan politics, and hate groups. What these personalized news services do is drive consumer opinion to the extremes, as they tend to reinforce existing biases and stereotypes rather than provide other points of view. This just further polarizes the polity. It should be remembered that the cause of the tribalist rejection of mainstream news was never that it was demonstrably false, but that it was inconvenient—it didn't comport with the preferred opinion.

We've now entered the era of "lock-on" news feeds that nourish the addiction to misinformation. Instead of looking for counterexamples to our worldview, we allow others to filter them, thereby ensuring the growth of collective ignorance and prejudice. This surfaces in subtle ways these days. Publishers hire selectivity readers to ensure that readers aren't accidentally offended. Amazon has review Nazis to limit reviewer bias: claiming a product is far superior to one product but inferior to another is verboten. These companies have not only diminished the value of free expression, they've lost sight of the criticality of the First Amendment to free societies.

There's a striking parallel between abuse-forming networking and phishing: both involve technical subterfuge (antisocial use of networking technology), perception management (manipulation of the public by getting them to think that they don't see something they do, or do see something they don't), and social engineering (motivating people to do something that they probably wouldn't have done otherwise, such as subscribe to a controversial blog).[14] Ruth Alexander's recent installment of the BBC series *The Inquiry* [15] also extends Pariser's work on filter bubbles to include data mining for psychometric profiling. I'll return to these topics in future columns.

REFERENCES

1. E. Pariser, *The Filter Bubble: How the New Personalized Web Is Changing What We Read and How We Think*, Penguin Books, 2011.
2. R. Metcalf, "Metcalf's Law after 40 Years of Ethernet," *Computer*, vol. 46, no. 12, 2013, pp. 26–31.
3. D.P. Reed, "That Sneaky Exponential—Beyond Metcalfe's Law to the Power of Community Building," 1999; www.deepplum.com/dpr/locus/gfn/reedslaw.html.
4. A. Odlyzko and B. Tilly, "A Refutation of Metcalfe's Law and a Better Estimate for the Value of Networks and Network Interconnections," 2005; www.dtc.umn.edu/~odlyzko/doc/metcalfe.pdf.
5. R. Tongia and E.J. Wilson III, "The Flip Side of Metcalfe's Law: Multiple and Growing Costs of Network Exclusion," *Int'l J. Communication*, vol. 5, 2011, pp. 665–681.
6. C.W. Mills, *The Power Elite*, 2nd ed., Oxford Univ. Press, 2000.
7. J. Bamford, *The Puzzle Palace: Inside the National Security Agency, America's Most Secret Intelligence Organization*, Penguin Books, 1983.
8. T. Weiner, *Enemies: A History of the FBI*, Random House, 2013.
9. H. Berghel, "Mr. Snowden's Legacy," *Computer*, vol. 47, no. 4, 2014, pp. 66–70.
10. D. Kravets, "Court Says It's Legal for NSA to Spy on You Because Congress Says It's OK," *Ars Technica*, 29 Oct. 2015; https://arstechnica.com/tech-policy/2015/10/court-says-its-again-legal-for-nsa-to-spy-on-you-because-congress-says-its-ok.
11. R.I.M. Dunbar, "Neocortex Size as a Constraint on Group Size in Primates," *J. Human Evolution*, vol. 22, no. 6, 1992, pp. 469–493.
12. A.W. McCoy, *Policing America's Empire: The United States, the Philippines, and the Rise of the Surveillance State*, Univ. of Wisconsin Press, 2009.
13. H. Berghel, "Disinformatics: The Discipline behind Grand Deceptions," *Computer*, vol. 51, no. 1, 2018, pp. 89-93.
14. H. Berghel, J. Carpinter and J.-Y. Jo, "Phish Phactors: Offensive and Defensive Strategies," *Advances in Computers*, vol. 70, 2007, pp. 223–268.
15. R. Alexander, "How Powerful Is Facebook's Algorithm?," *The Inquiry*, BBC World Service, 23 Apr. 2017; www.bbc.co.uk/programmes/p04zvqtx.

HAL BERGHEL is an IEEE and ACM Fellow and a professor of computer science at the University of Nevada, Las Vegas. Contact him at hlb@computer.org.

Rebooting Computers to Avoid Meltdown and Spectre

Thomas M. Conte, Georgia Tech
Erik P. DeBenedictis, Sandia National Laboratories
Avi Mendelson, Technion
Dejan Milojičić, Hewlett Packard Labs

Security vulnerabilities such as Meltdown and Spectre demonstrate how chip complexity grew faster than our ability to manage unintended consequences. Attention to security from the outset should be part of the remedy, yet complexity must be controlled at a more fundamental level.

Although the Meltdown[1] and Spectre[1,2] vulnerabilities were the result of specific designs ignoring the impact of speculative execution on security, the rapid rise in processor complexity over time, a key to achieving higher speed, probably made these types of flaws inevitable. If computing is to be "rebooted," complexity and security need to be a top-level concerns.

COMPLEXITY AND ABSTRACTION

Let's propose a holistic view of computing that accepts the need for both security and performance. The 1980s vintage 16-bit Motorola 68000 microprocessor had user and supervisor modes that enabled it to securely run most of the applications we have today, albeit at an anemic clock rate of 10–20 MHz. It did this with 68,000 transistors, hence its name. Speed increased disproportionally once chips reached a few million transistors, or 20× the 68000, because of internal parallelism such as speculation and other out-of-order instruction processing.

However, the number of potential states in a logic circuit is exponential in the number of transistors. Although the computer architects that design logic, called microarchitects, successfully manage the states they're thinking about, there are exponentially more states that they're not thinking about. This widens the attack surface such that

0018-9162/18/$33.00 © 2018 IEEE

EDITOR **ERIK P. DEBENEDICTIS**
Sandia National Laboratories;
epdeben@sandia.gov

attackers need only find a single entry point out of an exponentially growing number of options.

The root cause of the Meltdown–Spectre issue is a failed abstraction. Early computers were imagined to be state machines, where the computer's current state is held in a register and logic computes the next state from the current one. When originally conceived, state machines changed state on the rising edge of a clock, which was usually driven by a precise oscillator. The 68000 fit this model pretty well, as shown by the three instructions—load, add, and store—in Figure 1a. Each 68000 instruction used a specific number of clock cycles, illustrated as two, and the clock was precise, so the rate of instruction processing was also precise. It was so obvious that the instruction processing time carried no information that the community stopped thinking about the issue, setting the stage for a surprise several decades later.

The state machine abstraction became more complicated as computer architecture advanced. Faster transistors allowed a processor's internal clock rate to increase substantially, but the memory bus couldn't keep up. The fix was a cache memory that stored some data on the processor chip, where it could be accessed quickly; going off-chip to access real memory was only necessary in cases of a "cache miss." This led to the situation shown in Figure 1b, where the length of time required to load data from memory depended on whether the data was in cache or not.

But do microprocessors still satisfy the state machine abstraction? In the historical rush to keep up with Moore's law, the answer was "absolutely yes." Caches led to two versions of the state machine abstraction, and gave architects a new degree of freedom to slow down one state machine to get more work from the other. In fact, the computer architecture community made

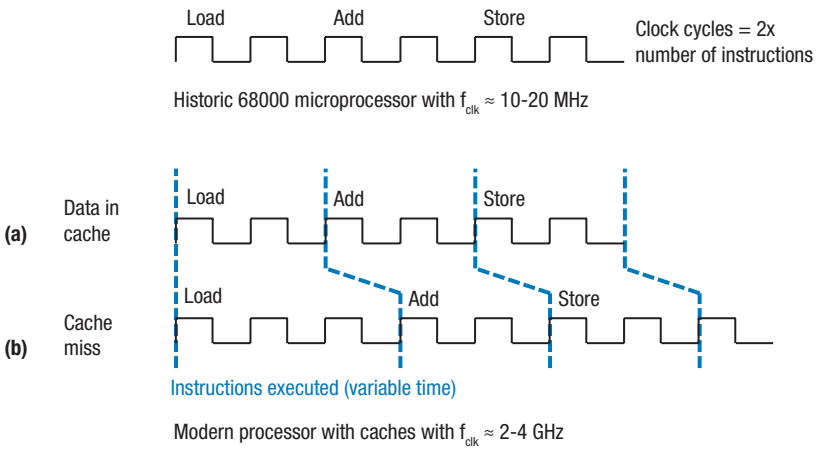


Figure 1. Microprocessors follow the state machine abstraction at two levels: (a) state machines at one level advance at the rate of the system clock, while (b) state machines at the other level advance at the rate of instruction processing. Before the advent of cache memory, the two state machines advanced in time at a fixed rate relative to each other; however, the rate varies in today's complex processors and becomes a side channel that can convey sensitive data to hackers.

a big deal out of the relative speeds of the two state machines. The cache miss rate could be reduced through improved hardware design, yielding more performance. While Figure 1b shows a load instruction taking either two or three clock cycles, computer architects ultimately represented load time as a real number comprised of the average of the two possibilities weighted by the cache hit-versus-miss rate.

Reanalysis of the issue in the last year led some out-of-the-box thinkers to take a different view. Although a single real number was convenient, in reality, it was a mixture of two discrete values, giving it attributes of a random number. A random number has a mean and standard deviation, making it look noisy if graphed over time. Further out-of-the-box thinking revealed that the purportedly random number is somewhat predictable, based on the layout of data in memory affecting whether a miss occurs. Once Meltdown and Spectre were disclosed to the public, computer engineers instantaneously changed their view of the amount of time required to execute an instruction on a complex processor. It's now a base time duration with a superimposed noisy signal that might carry information about internal processor state. This makes the hit-versus-miss ratio a covert "side channel," in computer security lingo.

Most modern processors use speculative execution to improve the cache's hit-versus-miss ratio and hence performance. Yet we now discover that most implementations of speculation allow data in otherwise inaccessible memory to change a subsequent cache hit into a miss—or vice versa—creating an opportunity for adversaries to craft malware that exports sensitive data to a lower-security environment.

The specific cause of Meltdown and Spectre was a timing side channel, but the root cause can be traced back to the state machine abstraction splitting into two abstractions without enough attention to unintended consequences. As we'll discuss below, Meltdown and

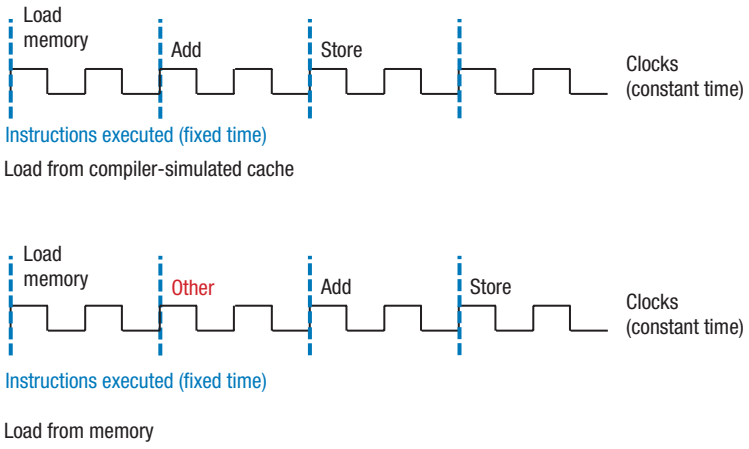


Figure 2. A potential remedy for Meltdown–Spectre class issues: replace today's automatically operating cache with one created by the compiler using temporary registers. Using the same load–add–store sequence from Figure 1, the compiler could put a value in a temporary register and (a) load it in one cycle or (b) reorder instructions so that the "other" instruction does useful work while leaving time for a load from memory. This approach should be about as fast as current methods, but the instruction rate won't depend on data because just one state machine abstraction is in use.

Spectre can be addressed in future chips through design changes; addressing the root cause in complexity will require deeper thinking.

FIXING MELTDOWN AND SPECTRE

Different groups around the world are busy finding the most efficient way to address the Meltdown and Spectre security bugs. For example, one of the authors, Tom Conte at Georgia Tech, suggests a deterministic VLIW approach. Today's caches decide whether to use on-chip cache or off-chip main memory at execution time. The current cache approach could be replaced by temporary registers and a more sophisticated compiler that essentially precomputes the cache misses. Because the compiler runs long before sensitive information is on the scene, the cache hit-versus-miss decision can't depend on it.

Fine-grained memory protection is another possibility. The discussion above showed how variance in the rate of instruction processing can carry information, but this only becomes a security problem if it allows access to sensitive information. Microarchitectural changes to the processor and additional memory tagging, called fine-grained memory protection, applied to both prefetching and speculative execution would allow the OS to block the side channel in places where it could convey sensitive data. Although microarchitectural changes are costly to implement, they would also make programs substantially more resilient to other bugs and exploits, such as stack or buffer overflows.

STEMMING THE GROWTH OF COMPLEXITY

The two methods just described might be good options that deserve to be implemented, but they have a common vulnerability: when a system that's too complex to understand develops undesired behavior, adding a slew of patches makes the system even more complex, risking the introduction of additional undesired behaviors due to the increased complexity.

It's possible to create computer systems with enormous numbers of transistors that still function securely through the mundane "air gap" method. The design of a data center illustrates this principle. Data centers typically have many servers with just one application per server, turning off all unnecessary TCP/IP ports. This might lead to one server for email, one for passwords, one for the database, and so on. The air gap comprises inches or feet of space between servers, isolating the servers on either side in a way that's not subject to the exponential complexity growth found in microprocessor architectures.

Air gaps aren't perfect barriers, but the method of detecting and eliminating them is very different from microarchitectural vulnerabilities. For example, an air gap was recently breached by malware flashing the activity light on a hard drive while the attacker stationed a drone carrying a video camera outside a window to record the flashing pattern.[3] These types of attacks often have simple remedies because of their inherently physical origin—like closing the curtain over the window.

Just like air gaps or other hardware barriers, we can partition systems in software by running an OS instance on individual cores and having them communicate with message passing, such as Barrelfish[4] did nine years ago. If performance is an issue, the message passing can be implemented using shared memory. The same is true for tight communication channels between user space and kernels, which is another choke point preventing information leaks.

Similar approaches are being adopted to partition OSs' memory into different degrees of trust. This approach has always been the basis of object-based systems, which, if supported by hardware similar to capabilities and call gates, can provide access to data only through well-defined interfaces while protecting the inner data of an object.

Partitioning is becoming popular, although not in computers readily accessible to most programmers. Mobile devices achieve high performance with long battery life by having many

chips specialized to specific functions separated by air gaps—just like the servers in a data center.

Software can also add noise to decrease the signal-to-noise ratio and make it more difficult for hackers to extract sensitive information. For example, OSs randomly map virtual pages to physical page locations to prevent adversaries from knowing where sensitive data resides. Random time delays can be effective at higher levels of software as well, such as was done for Spectre in JavaScript interpreters in Google's Chrome and Mozilla's Firefox browsers.

There's an expectation that computers should be secure, but is this realistic? We increasingly view computers as artificial intelligences, expecting them to keep secrets just like humans. But humans aren't particularly good at keeping secrets. Poker, for example, is based on players developing the skill to see into another person's mind by using the cues that people emit but don't pay attention to—just like the timing side channels exploited by Meltdown and Spectre. Furthermore, humans can be tortured, polygraphed, and subject to drugs, vulnerabilities analogous to those available for analyzing computers when physical access is possible. So while it's clear that security must be a top-level requirement when designing future architectures, computer security will remain an ongoing challenge.

ACKNOWLEDGMENTS

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the US Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

REFERENCES

1. M. Lipp et. al, "Meltdown," arXiv: 1801.01207; Meltdown and Spectre; https://meltdownattack.com/meltdown.pdf.
2. J. Horne, "Reading Privileged Memory with a Side-Channel," Project Zero, 3 Jan. 2018; https://googleprojectzero.blogspot.com/2018/01/reading-privileged-memory-with-side.html.
3. A. Greenberg, "Malware Lets a Drone Steal Data by Watching a Computer's Blinking Light," *WIRED*, 22 Feb. 2017; https://www.wired.com/2017/02/malware-sends-stolen-data-drone-just-pcs-blinking-led.
4. A. Baumann et. al., "The Multikernel: A New OS Architecture for Scalable Multicore Systems," *Proc. ACM SIGOPS 22nd Symp. Operating Systems Principles* (SOSP 09), 2009, pp. 29–44; https://doi.org/10.1145/1629575.1629579.

THOMAS M. CONTE is a professor at Georgia Tech and the director of its Center for Research into Novel Computing Hierarchies (CRNCH). Contact him at conte@gatech.edu.

ERIK P. DEBENEDICTIS is a technical staff member at Sandia National Laboratories Center for Computing Research. He's a member of IEEE, ACM, and APS. Contact him at epdeben@sandia.gov.

AVI MENDELSON is a professor in the computer science and electrical engineering departments at Technion Haifa. He's a member of IEEE and ACM. Contact him at avi.mendelson@technion.ac.il.

DEJAN MILOJIČIĆ is a distinguished technologist at Hewlett Packard Labs. He's a Fellow of IEEE, Distinguished Engineer of ACM, and member of USENIX. Contact him at dejan.milojicic@hpe.com.

Net Neutrality: A Brief Overview of the Policy and the FCC's Ruling to Upend It

Mina J. Hanna, Synopsys

In this first installment of Computer's *new column, the author gives an overview of the net neutrality policy and the FCC's recent rulings that threaten it.*

The debate over net neutrality has been contentious since the early days of the Internet. It has repeatedly been challenged in court by Internet service providers (ISPs) for years, culminating in the latest decision by the Federal Communications Commission (FCC) to upend the 2015 net neutrality ruling. Critics see this decision as a threat to Internet consumer freedoms, economic competition (imposing a significant market disadvantage on small businesses), and even the progress of scientific research and knowledge discovery. Seeing how important the Internet is to the advancement of innovation, development of human capital, and economic growth, it is imperative for technologists, researchers, and innovators to take heed of the policy debate and the latest developments.

On 18 May, 2017, the FCC voted to put forth the Restoring Internet Freedom Notice of Proposed Rulemaking (NPRM), seeking comments on a proposal to roll back the 2015 net neutrality regulations. In 2015, the Obama administration passed a rule to classify the Internet as a public utility under Title II of the 1934 Communications Act. The rule prohibits ISPs from preferentially allocating bandwidth for affiliated content, throttling bandwidth, or implementing preferential price structures for their customers. Several months later, on December 14, the FCC moved forward with their decision to repeal the rule after a 3–2 party-line vote.[1]

POLICY BACKGROUND

The US Constitution, through the speech and press clause of the First Amendment, established a foundational tenet that was unequivocally fundamental to its progress as a nation. The amendment—which reads, "Congress shall make no law abridging the freedom of speech, or of the press"—codified the freedom of expression as an essential

0018-9162/18/$33.00 © 2018 IEEE

EDITOR **MINA J. HANNA**
Synopsis; mhanna@synopsys.com

liberty guaranteed by US law. Interestingly, the amendment, which was conceived by James Madison and introduced in the House of Representatives in 1789, was originally worded as follows: "The people shall not be deprived or abridged of their right to speak, to write, or to publish their sentiments; and the freedom of the press, as one of the great bulwarks of liberty, shall be inviolable" (http://constitution.findlaw.com/amendment1/annotation06.html).

While James Madison and the rest of the Founding Fathers in their day would have never conceived of a system that remotely resembles the Internet, one could argue that the Internet has fulfilled their vision. It became the quintessence of the First Amendment as perhaps the most perfect embodiment of a democratic system. The independence of the Internet from any exclusive use by state actors, private interests, or a single political entity for their sole benefit has been pivotal in advancing humanity, our quest for knowledge, and human welfare.

Far beyond what Madison envisioned, the Internet has ushered in continuous waves of radical technological advances, scientific breakthroughs, and forces of novel and disruptive innovations that bolstered the economic growth and prosperity of all nations across the globe. From 2006 to 2011, the Internet accounted for 21 percent of total GDP growth of mature economies.[2] The Internet Association estimated that the Internet sector was responsible for close to $966.2 billion or 6 percent of the US real GDP in 2014,[3] surpassing several other established sectors. The Internet user penetration rate is at 92 percent in the US—the Internet is almost ubiquitous, and it still has unbounded potential to drive more economic growth and reimagine the landscape of technology, tools, and business models.

This progress has been enabled by the Internet functioning as an equitable platform promoting a free space for innovation, scientific research, economic competition, and free expression. Net neutrality became a standard to enshrine these principles and uphold the open architecture of the Internet. In this sense, net neutrality aimed to make the Internet a public good.

However, the Internet is not strictly a public good (like free-to-air television programming, radio transmission, or national defense) because it is not maintained by the government but by ISPs. This means the Internet is a nonrivalrous, potentially excludable good.

The Internet is almost ubiquitous, and it still has unbounded potential to drive more economic growth and reimagine the landscape of technology, tools, and business models.

- *Nonrivalry* in a market is where the consumption of any resource by an individual does not diminish the use of another individual of the same resource.
- *Excludability* applies to goods or services, the access to which is provisional upon payment to the provider of such products or services, wherefore the provider can exclude consumers from having access if they have not paid for it. Put differently, any given user can be prohibited from the use of the Internet with no consequences.

Technology, intellectual property, and proprietary content (like information, music, movies, and computer software, which are often protected by Digital Rights Management technologies and copyright laws), are considered nonrivalrous and excludable goods, since the producers of such content need to realize a profit to justify the cost of production. This is where the crux of the matter lies. Should the Internet be regarded as an information service (regulated under Title I of the 1934 Communications Act), in and of itself a nonrivalrous and excludable good, or as a telecommunication service (governed under Title II), a public good that merely connects the consumer to sources of information?

Private provision runs some risk of monopoly, and the Internet is not an exception. A critical question presents itself, then. Considering the monumental importance of the Internet, should there be any restrictions or regulations preventing ISPs from transforming the Internet into a monopoly, a market where ISPs can be price makers instead of price takers? Price takers in a perfectly competitive free market are constrained by the broader supply and demand requirements of the market; they can't raise the price of broadband access without reasonably expecting to lose customers in the process to their competitors. By contrast, in a perfect monopoly, due to lack of competition, market actors' decisions dictate market prices as they are not bound by supply and demand. Monopolistic ISPs can restrict output by blocking or throttling content, downgrade the quality of their product, offer paid prioritization, apply data caps and zero ratings, and prohibitively raise

prices and not risk any substantial loss to their profit margins whatsoever.

Internet activists have been arguing that the Internet market in the US is already a monopolistic competition, or worse yet, an oligopoly, where two or more firms control a majority of the market share. An example of an oligopoly is Canada, where only three companies—Rogers Communications Inc., Bell, and Telus Corp.—control approximately 90 percent of the market. In early 2014 (and again in 2016), all three companies raised the price for smartphone plans to $80 in most markets in tandem.[4]

In the US, antitrust practices like price gouging and price fixing of any product is a criminal violation under the Sherman Antitrust Act, a civil violation under the Federal Trade Commission (FTC), and an offense under state antitrust laws (www.ftc.gov/tips-advice/competition-guidance/guide-antitrust-laws/antitrust-laws). And yet, ISPs did in the past knowingly engage in unfair antitrust practices[5] and continue to do so.

FCC OPEN INTERNET ORDER OF 2010, NET NEUTRALITY RULE OF 2015, AND THE 2017 FCC RULING

In the US, regulating the Internet falls within the federal government's authority to regulate interstate commerce, specifically within the jurisdiction of the FCC. The FCC's legal power to regulate telecommunications (or common) carriers is granted by Title II of the Communications Act of 1934 and section 706 of the Telecommunications Act of 1996, which authorizes the FCC to facilitate the deployment of broadband to Americans. The FCC's purview—among other things—is to broaden the implementation of and encourage investments and innovation in broadband technologies, in addition to ensuring that similar regulatory treatment is applied for all competing broadband providers (www.fcc.gov/general/national-broadband-plan).

The point of contention in this decade-long debate is how ISPs are regulated under the law: as a telecommunications service (under Title II) or as an information service (under Title I). Title II explicitly gives the FCC the authority to protect consumers and online businesses against "unjust or unreasonable discrimination" by ISPs. Title I does not. And in the absence of a Congressional act, it is left up to the FCC to make a decision.

> The US Government Accountability Office (GAO) will investigate the possibility of fraud and identity theft during the FCC's process.

The FCC approved the Open Internet Order[6] of 2010, setting the following specific rules:

- *Transparency*. Fixed and mobile broadband providers must disclose the network management practices, performance characteristics, and terms and conditions of their broadband services.
- No blocking. Fixed broadband providers may not block lawful content, applications, services, or nonharmful devices; mobile broadband providers may not block lawful websites, or block applications that compete with their voice or video telephony services.
- *No unreasonable discrimination*. Fixed broadband providers may not unreasonably discriminate in transmitting lawful network traffic.

The Open Internet Order was strongly challenged in court (*Comcast Corp. v. FCC*, 600 F.3d 642 [DC Cir. 2010] and *Verizon v. FCC*, 740 F.3d 623 [DC Cir. 2014].[7,8] The Court of Appeals of the DC Circuit ruled in both cases against the FCC and vacated the "no blocking" and "no unreasonable discrimination" rules from the order, arguing that these rules only apply to common carriers.

Instead of appealing the court ruling on the Open Internet Order, the FCC passed the net neutrality rule of 2015, directly classifying the Internet as a telecommunications service, not as an information service. Therefore, the Internet can be considered a public good and ISPs can be regarded under the law as common carriers that grant access to broadband service to consumers under Title II of the Communications Act of 1934.

In *USTA v. FCC* (DC Cir. 2016),[9] the court eventually ruled in favor of treating the Internet as a utility, not a luxury, and it upheld the FCC's classification of broadband providers as common carriers—thus, the net neutrality rule prevailed.

Fast forward to 2017, under a new administration. On 18 May, the FCC put forward a NPRM intended to repeal the net neutrality rule and return to light-touch Internet regulations. And on December 14, the FCC voted in favor of ISPs to repeal the net neutrality policy.

Current Development

In response to the FCC repeal of the net neutrality ruling, Senate minority leader Chuck Schumer (D-NY) stated he would force a vote to undo the FCC's decision with a Congressional resolution of disapproval. Congress can overturn agency actions by invoking the Congressional Review Act (CRA). The FCC repeal takes effect 60 days after publication of the rule in the Federal Register, which in this case was published on 22 February, 2018. Fifty Democratic senators pledged to vote for the resolution, which only needs a simple majority to pass (51 votes) to undo the FCC regulation. But even if

the CRA passes both chambers (which is unlikely in the House), President Trump can veto it.

A coalition of 22 states and the District of Columbia are suing the FCC for preempting states from imposing their own net neutrality rules and adducing the bulk of fake comments submitted in response to the FCC's NPRM as a corrupting record on the FCC's net neutrality rulemaking process. The US Government Accountability Office (GAO) will investigate the possibility of fraud and identity theft during the FCC's process.

The Pew Research Center had previously estimated that close to 57 percent of the 21.7 million public comments submitted to the FCC appeared to include fraudulent information and fake names, with only 6 percent of submitted comments being unique. The rest had been sent thousands of times, including comments that appear to have originated from Russia.[10]

On the US House side, Representative Marsha Blackburn (R-TN) introduced HR 4682, the Open Internet Preservation Act (www.congress.gov/bill/115th-congress/house-bill/4682), in response to the FCC decision. The bill "amends the Communications Act of 1934 to ensure Internet openness, to prohibit blocking of lawful content, applications, services, and non-harmful devices, to prohibit impairment or degradation of lawful Internet traffic, to limit the authority of the Federal Communications Commission and to preempt State law with respect to Internet openness obligations, to provide that broadband Internet access service shall be considered an information service, and for other purposes."

The Open Internet Preservation Act has been widely criticized by Amazon, Google, Microsoft, and public interest groups for defining broadband as an information service because it would mean it couldn't be regulated more strictly as a Title II service, as it was under the repealed Open Internet Order. The bill would allow ISPs to offer paid access and content prioritizations, and would curb states' rights in making their own determination on this policy. The bill does exactly what the FCC intended in repealing the Obama-era regulation.[11]

Representative Sean Patrick Maloney (D-NY) introduced another bill, HR 4585 (the Save Net Neutrality Act of 2017; www.congress.gov/bill/115th-congress/house-bill/4585), "to prohibit the Federal Communications Commission from relying on the Notice of Proposed Rulemaking in the matter of restoring Internet freedom that was adopted by the Commission on May 18, 2017, to adopt, amend, revoke, or otherwise modify any rule of the Commission."

The net neutrality debate is far from settled, and it remains to be seen whether the Congressional resolution of disapproval will secure the remaining vote in the Senate before 23 April, when the FCC's Restoring Internet Freedom will take effect. Similarly, we will find out if any of the 22 states' court cases and the GAO's investigation will reveal any information about the FCC NPRM process.

REFERENCES

1. C. Kang, "F.C.C. Repeals Net Neutrality Rules," *The New York Times*, 14 December 2017; www.nytimes.com/2017/12/14/technology/net-neutrality-repeal-vote.html.
2. J. Manyika and C. Roxburgh, "The Great Transformer: The Impact of the Internet on Economic Growth and Prosperity," McKinsey Global Institute, Oct. 2011; www.mckinsey.com/industries/high-tech/our-insights/the-great-transformer.
3. S. Siwek, "Measuring the U.S. Internet Sector," Internet Association, Dec. 2015; https://cdn1.internetassociation.org/wp-content/uploads/2015/12/Internet-Association-Measuring-the-US-Internet-Sector-12-10-15.pdf.
4. T. Hopper, "Why Canadian Cell Phone Bills are among the Most Expensive on the Planet," *The National Post*, 18 Sept. 2017; http://nationalpost.com/news/canada/why-canadian-cell-phone-bills-are-among-the-most-expensive-on-the-planet.
5. S. Nichols, "Merger-Hungry AT&T Sued for Price Gouging by Texas ISP," The Register, 6 Dec. 2017, www.theregister.co.uk/2017/12/06/mergerhungry_att_sued_for_wait_for_it_price_gouging.
6. Federal Communications Commission, "In the Matter of Preserving the Open Internet Broadband Industry Practices," 23 Dec. 2010, https://apps.fcc.gov/edocs_public/attachmatch/FCC-10-201A1.pdf.
7. Comcast Corp. v. Federal Communications Commission, 2010 (US Court of Appeals for the District of Columbia Circuit).
8. Verizon v. Federal Communications Commission, 2014 (US Court of Appeals for the District of Columbia Circuit).
9. US Telecom Association v. Federal Communications Commission, 2015 (US Court of Appeals for the District of Columbia Circuit).
10. "Over Half of Public Comments to FCC on Net Neutrality Appear Fake: Study," Reuters, 29 Nov. 2017; www.reuters.com/article/us-usa-internet-pew/over-half-of-public-comments-to-fcc-on-net-neutrality-appear-fake-study-idUSKBN1DT297.
11. A. Robertson, "The Republican Net Neutrality Bill Doesn't Save Net Neutrality," The Verge, 19 Dec. 2017; www.theverge.com/2017/12/19/16797778/congress-open-internet-preservation-act-marsha-blackburn-net-neutrality-bill.

MINA J. HANNA is a senior software consultant at Synopsys, Inc. Contact him at mhanna@synopsys.com.

Penetration Testing in the IoT Age

Chung-Kuan Chen, Zhi-Kai Zhang, Shan-Hsin Lee, and Shiuhpyng Shieh,
National Chiao Tung University

Internet of Things (IoT) objects offer new services but also pose new security threats. Due to the heterogeneity, large number, and resource constraints of these objects, new penetration testing tools and techniques are needed to complement defensive mechanisms.

Internet of Things (IoT) devices and services are now integral to most daily activities. However, the IoT brings not only added convenience but, by connecting more and more objects to the Internet, new security threats.[1] Many applications in IoT ecosystems, from smart homes to customized healthcare, contain sensitive personal information that can become the targets of network attacks.

Unfortunately, ensuring the security of IoT objects is not straightforward for three major reasons. First, the IoT's heterogeneous nature makes it vulnerable to many kinds of attacks. Second, heavyweight protection mechanisms are infeasible for resource-constrained IoT devices. Third, many IoT objects are deployed only once and thereafter are rarely maintained or updated.

PENETRATION TESTING

Due to these challenges, penetration testing (PT), which employs offensive attack techniques to discover vulnerabilities, is often used to complement defensive security methods before IoT objects are deployed. Because malicious attacks need only a single exploit to be successful, improving PT coverage is crucial. To enhance manual PT, security researchers use automated tools to carry out three types of specialized PT: interface testing, transportation testing, and system testing.

Interface testing targets interfaces that interact with external users or devices. Major vulnerabilities can exist in an application if its input validation mechanisms are not in effect. In the Open Web Application Security Project (OWASP) tester guidelines for IoT applications (www.owasp.org/index.php/IoT_Testing_Guides), the categories "insecure web interface" and "insecure network services," among others, would be addressed by interface testing.

Transportation testing focuses on misuse issues and design flaws in communication protocols and weak cryptographic schemes. In the OWASP guidelines, "insufficient authentication/authorization," "lack of transport encryption/integrity verification," and "privacy concerns" fall into this type of testing.

System testing examines firmware, OSs, and system services for implementation flaws, insecure system settings,

0018-9162/18/$33.00 © 2018 IEEE

and other known vulnerabilities. In the OWASP guidelines, "insufficient security configurability" and "insecure software/firmware" are relevant for system testing.

To cope with the heterogeneity and large quantity of IoT objects, we propose modularization of test modules to scale up all three types of testing. At the same time, due to IoT devices' resource limitations, intelligent approaches are desirable for generating test plans based on available test modules to reduce wasted resources and redundant effort while extending test coverage.

INTERFACE TESTING

Many user-facing IoT objects have web-based interfaces, and these can have various vulnerabilities. Among the most common is input validation failure. Unlike traditional web interfaces, which are linked to operations closely coupled with data manipulation, IoT object interfaces can also be linked to code-oriented operations such as controlling system programs. Code-oriented attacks such as command injection and code injection could be even more severe than data-oriented attacks. Improving input validation testing is thus critical for the IoT. Although testing web-based interfaces is our focus, the same modularization and intelligence mechanisms described below can be applied to other types of IoT applications.

Modularized design

Testers employ various techniques for different input validation vulnerabilities. However, these methods are conceptually similar in that they all crawl to the entry points and submit the test payload. Modularizing interface testing would make it easier to create testing tools for specific vulnerabilities and install them on demand. Moreover, algorithms could be developed in a more systematic way—for example, to implement adaptive, prioritized, or automutation test strategies.

Intelligent payload mutation

Because IoT objects can lack comprehensive input validation mechanisms, extending the coverage of test payloads is desirable. A widely used method, fuzz testing, employs randomly generated payloads, but this is inefficient due to resources wasted on meaningless inputs. An alternative is to exhaustively or randomly generate syntax-correct inputs. This method provides better test coverage but is still inefficient, as the space of syntax-correct inputs is usually large.

Modularizing interface testing would make it easier to create testing tools for specific vulnerabilities and install them on demand.

Intelligently mutating known payloads is a compromise between manual testing and exhaustive/random testing. Combining existing evasion techniques provides greater ability to circumvent validation mechanisms. In this case, conflicting or overlapping techniques should be manipulated carefully to prune unnecessary test cases.[2] On the other hand, converting payloads to syntactically or semantically equivalent payloads is worthy of further investigation. Syntactic mutation generates payloads with slight changes at the syntax level. For example, SQL code "`'or 1 = 1`" can be mutated to "`'|| 1 = 1`". Semantic mutation converts the whole payload to functional equivalent ones. For instance, "`id = 1 or 1`" is semantically equivalent to "`id = id xor 0`".

Intelligent entry-point crawling

Entry-point discovery in an IoT environment with numerous network services is difficult and time-consuming. Because service entry points can be dynamically generated, the links between them can be complex, and loops might be produced across IoT objects. In addition, a dispatcher might be built into an IoT application to manage entry points. As the dispatcher can be in either a centralized or distributed structure, a crawler should be able to discover as many entry points as possible in both types of structures to locate more test targets. A proof-of-concept vulnerability scanner that does this, VulScan,[2] has been developed to complement manual PT.

TRANSPORTATION TESTING

Transportation testing is performed both on the network infrastructure interconnecting IoT objects as well as the associated cryptographic schemes and communication protocols used to protect messages.

New network infrastructures

Messages between IoT objects traverse heterogeneous networks such as TCP/IP, Zigbee, and 6LoWPAN. To allow more efficient object communication, new infrastructures such as FIA (www.nets-fia.net), HUB4NGI (www.hub4ngi.eu), and PNS[3] have been proposed. New PT tools are needed to test these infrastructures, the protocols, and the gateways or converters between the infrastructures and protocols. Because network heterogeneity is a key issue in IoT communication, transportation testing should be modularized to provide better flexibility.

Cryptographic issues

In general, the cryptographic algorithms that protect network communication are believed to be secure due to theoretical proofs. When vulnerabilities are discovered, they're generally attributable to misuse, implementation failures, and bad protocol design. However, resource-constrained IoT objects can't afford heavyweight cryptographic mechanisms. Moreover, messages between devices usually are well formatted and lack entropy. The combination of these factors could make differential cryptanalysis or statistical attacks possible.

Trusted platform modules (TPMs) enable new applications but also raise new threats. For example, the ROCA vulnerability[4] is caused by a weak prime-number generator in the RSA library within TPMs. This vulnerability affects many vendors including Microsoft, Google, and HP. Another example is KRACK attacks,[5] which exploit a flaw in Wi-Fi's WPA2 encryption and affects all major software platforms. As cryptographic operations are rarely computed in cleartext, developing PT methods to discover such vulnerabilities in the IoT is challenging.

SYSTEM TESTING

In contrast to interface testing, which focuses on commonly used technologies such as web interfaces, proprietary programs are the main targets of system testing. Without having knowledge of such systems, testers often resort to black-box methods, such as fuzz testing. Given the large number of IoT objects to be tested, exhausting all test cases is infeasible. It's therefore helpful generating test cases through automatic reverse-engineering, what is termed grey-box PT.

In conventional computing environments the x86/x64 instruction-set architecture (ISA) dominates, but other ISAs such as ARM, MIPS, and PPC are also used in the IoT. OSs vary among IoT objects as well, with general-purpose OSs such as Linux, Windows, and Android often customized. The diversity of IoT objects makes automated reverse-engineering challenging.

Encapsulation

To mitigate the impacts of system diversity, encapsulation can enable cross-platform analysis. Encapsulation involves using an abstract language such as LLVM (http://llvm.org) or VEX (http://valgrind.org) to create an intermediate representation (IR) of different machine languages to emulate ISAs. Hardware-assisted emulators are used to test programs running on specific ISAs, but software-based emulators such as QEMU (www.qemu.org) and Bochs (http://bochs.sourceforge.net) can leverage multiple ISAs and are more suitable for IoT objects. Another method for building an IR is symbolic execution, which translates a program to mathematical constraints and evaluates whether certain properties can be satisfied. With these constraints, developing an intelligent PT method with a more formalized foundation is possible.

To mitigate the impacts of system diversity, encapsulation can enable cross-platform analysis.

Virtual machine introspection

While symbolic execution mostly deals with per-process information, system-wide runtime information is also important for PT. However, runtime analysis tools might not be available for IoT objects. Due to resource constraints, object diversity, and proprietary architectures, developing debugging and analysis tools for different objects is usually infeasible. An alternative approach is virtual machine introspection (VMI), which monitors VM execution in the hypervisor outside the VM.[6,7] Because VMI doesn't modify the guest OS, IoT objects are easier to deploy. Through VMI, the emulator's out-of-box monitoring, memory forensics, and debugging features can be developed more easily to enable both manual and automatic PT.

Intelligent grey-box testing

As the boundary of grey-box testing is more obscure than white- and black-box testing, a systematic division of testing phases enables the development of future testing techniques. Intelligent grey-box PT can be divided into four phases: vulnerability model construction, execution path exploration, vulnerability path searching, and vulnerability path verification. To discover vulnerabilities, the model of abnormal behaviors is first constructed. Next, control flows are analyzed to find each execution path. The vulnerability risk for each path is then estimated using information from the IR and VMI to prioritize testing order. Once the path with highest risk is identified, the symbolic execution resolves inputs to the path. During the final phase, if the resolved input is available, a verifier can monitor the program with the input to check whether the vulnerability model can be satisfied. Using this systematic approach, intelligent grey-box PT can discover system-level vulnerabilities.

To cope with the heterogeneity, large number, and resource constraints of IoT objects, PT tools and techniques should apply the principles of modularization and intelligence. Modularization provides the flexibility to test various targets, and intelligence enlarges test coverage and improves accuracy. In interface testing, input validation mechanisms should be tested using an intelligent mutation engine and entry-point

discovery automated. Transportation testing must address the problem of messages between IoT objects traversing heterogeneous networks. To deal with emerging IoT network infrastructures, PT tools should be compatible with the overlay networks. Cryptographic misuse issues and implementation flaws must also be considered. In system testing, the challenge is IoT objects with various ISAs and OSs. If encapsulation and related translation modules are available, cross-platform analysis becomes feasible. VMI and symbolic execution can be applied on top of encapsulation. In this way, intelligent analysis methods can be used to discover vulnerabilities in variant platforms.

REFERENCES

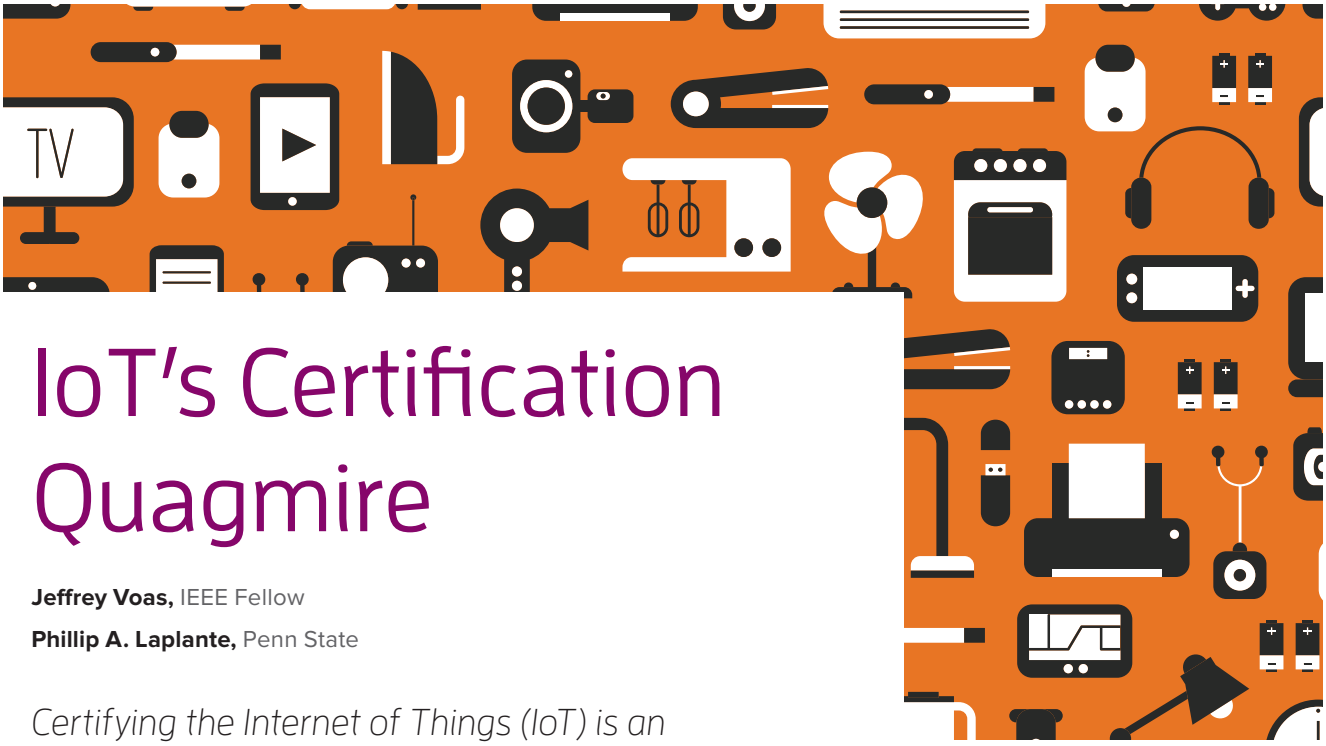
1. Z.-K. Zhang et al., "IoT Security: Ongoing Challenges and Research Opportunities," *Proc. IEEE 7th Int'l Conf. Service-Oriented Computing and Applications* (SOCA 14), 2014, pp. 230–234.
2. H.-C. Huang et al., "Web Application Security: Threats, Countermeasures, and Pitfalls," *Computer*, vol. 50, no. 6, 2017, pp. 81–85.
3. Z.-K. Zhang et al., "Identifying and Authenticating IoT Objects in a Natural Context," *Computer*, vol. 48, no. 8, 2015, pp. 81–83.
4. M. Nemec et al., "The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli," *Proc. 2017 ACM SIGSAC Conf. Computer and Communications Security* (CCS 17), 2017, pp. 1631–1648.
5. M. Vanhoef and F. Piessens, "Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2," *Proc. 2017 ACM SIGSAC Conf. Computer and Communications Security* (CCS 17), 2017, pp. 1313–1328.
6. K. Nance, M. Bishop, and B. Hay, "Virtual Machine Introspection: Observation or Interference?," *IEEE Security & Privacy*, vol. 6, no. 5, 2008, pp. 32–37.
7. C.-W. Wang et al., "Cloudebug: A Programmable Online Malware Testbed," *Computer*, vol. 47, no. 7, 2014, pp. 90–92.

CHUNG-KUAN CHEN is a PhD candidate in the Department of Computer Science at National Chiao Tung University (NCTU). Contact him at ckchen@cs.nctu.edu.tw.

ZHI-KAI ZHANG is a PhD candidate in the Department of Computer Science at NCTU. Contact him at skyzhang.cs99g@g2.nctu.edu.tw.

SHAN-HSIN LEE is a PhD student in the Department of Computer Science at NCTU. Contact him at shlee.cs06g@nctu.edu.tw.

SHIUHPYNG WINSTON SHIEH is a university chair professor and past chair of the Department of Computer Science at NCTU. Contact him at ssp@cs.nctu.edu.tw.



IoT's Certification Quagmire

Jeffrey Voas, IEEE Fellow
Phillip A. Laplante, Penn State

Certifying the Internet of Things (IoT) is an important but unachieved goal. Challenges include who does the certification and what to certify. The authors address these challenges and propose a way forward that applies to the IoT.

FROM THE EDITOR

The Internet of Things (IoT) ecosystem is still a nascent area. When you purchase an IoT product, it's hard to know what kind of system you are buying. For example, what is the security and privacy story? Have the manufacturers thought through the safety issues? Without independent certification, this will remain a deficiency of the industry. This article examines those issues in detail, the challenges that certification for IoT faces, and what needs to be done to have reasonable coverage without impeding the industry. —*Roy Want*

Certification has been a goal of software and hardware professionals for decades. However, this goal generally remains elusive because the quality and integrity of the certification criteria and certification authorities can be suspect.

Examples of certification success can be found in the safety-critical avionics community, in part due to regulation. Designated Engineering Representatives handle these tasks on behalf of the Federal Aviation Administration and sign off on airplane safety before planes are authorized to operate. Other US regulatory agencies such as the Food and Drug Administration and the Nuclear Regulatory Commission have similar goals but different processes. However, for general-purpose consumer products, slowing time to market (for the sake of better product quality) might be dismissed because "first to market" is often preferable and certification adds costs for vendors.

The proliferation of Internet of Things (IoT)-based systems, particularly in homes, has created new concerns for consumer safety, security, and privacy. Problems with insecure "things" such as appliances, toys, and even light bulbs can create new vulnerabilities. For example, intelligent

0018-9162/18/$33.00 © 2018 IEEE

EDITOR **ROY WANT**
Google; roywant@google.com

virtual assistants such as Amazon's Echo are widely used in homes but can be hacked, providing unwanted access to homes and private information.[1]

Another certification issue is the repeated misuse or misunderstanding of the IoT. Is it simply any noun we can stick "smart" in front of? Smart home, smart car, smartphone, smart appliance, smart grid, smart city, and so on. Without some agreed-upon standard definition as to what the IoT is, the goal of it being "certifiable" appears immature at best and foolish at worst.

Microsoft Regional Director Troy Hunt notes that often "we're taking everyday consumer goods and adding Internet for no apparent good reason." Hunt is so concerned about the vulnerabilities introduced by certain devices that he asks, "What would it look like if we put warnings on IoT devices like we do cigarette packets?"[2]

So how can we guarantee, ensure, or otherwise certify that IoT-based systems or individual devices satisfy certain properties, are safe to use, and protect privacy and/or security? Co-author Jeffrey Voas suggested that there are three distinct approaches to certification: certifying the product, certifying the processes that produce the product, and certifying the people who produced the product.[3] He calls the confluence of these three approaches "the software quality certification triangle."

Voas observed that "although you can approach software certification in any one of these ways (the third, product certification, most strongly correlates with the "goodness" of software), a combination of the three provides a more balanced approach. This [triple approach to certification] is preferable, since any one of the three can be inadvertently misapplied."[3] Although this model was intended for software certification, it also applies to IoT things and networks.

CERTIFICATION AUTHORITIES

Before we talk about certifying products, processes, and people, we need to mention certification authorities (CAs). What benefit is certification if CAs cannot be trusted? Daniele Miorandi and his colleagues agree: "Many open issues have to be addressed in order to develop IoT trust services. [But] first, the definition of globally accepted certification authorities should be addressed, together with a number of requirements that an IoT-compliant certification authority should respect."[4] But who certifies the certifier? And do we need to ask who certifies the certifier of the certifier?

Certification is a balancing act between vendor and certifier. Certifiers have liability, and the results from the certifier can project liability onto a vendor—this creates tension.

CERTIFYING PRODUCTS

By product, we mean individual devices, computational power, clouds, the network of things itself, and everything in between (for example, communication protocols). Devices get updates pushed to them continuously—will a prior certification still hold or does ongoing certification need to occur? Furthermore, what about issues such as mis-certification and de-certification?

The definition of "device" can get in the way here. Is a smartphone one thing or many things? It has a camera and apps with numerous permissions, and it records sounds and takes videos. Are each of these capabilities individual things that need vetting?

NIST's "Network of 'Things'" model defines five primitives (classes of things) for any IoT-based system: sensor, aggregator, communication channel, external utility, and decision trigger.[5] These are the basic building blocks that define the science of what is referred to as the IoT. Consider these building blocks as classes of things that can be certified as standalone entities. The issue, though, is that standalone certifications do not equate to a certification of the whole. Let's briefly look at a few certification issues for each of these classes of things.

> Without some agreed-upon standard definition as to what the IoT is, the goal of it being "certifiable" appears immature at best and foolish at worst.

Sensor

This represents the devices used to sense the environment of a network of things. Organizations such as Underwriters Laboratories (UL) will certify "electronic gadgets" containing sensors against electric shock, but the IoT will need more than that. For example, sensors need security and compatibility certifications.

Even if an IoT CA did exist, there are no IoT-specific databases of vulnerabilities from which to test against. Co-author Phillip Laplante created a nascent database of IoT sensor failures (https://iotfdb.laplante.io), but it has received little participation.

Economic realities can affect certification processes as well. For example, some IoT devices have shorter lifespans than the time it would take to certify them. IoT device vendors often don't have the time or resources

to focus on building in safety and privacy. Additionally, the cost to certify a sensor could be much greater than the cost of the sensor itself.

Aggregator

An aggregator collects and fuses data from various sensors or other data feeds, such as a cloud database. Aggregation can be done by hardware or software; both can be vetted for the efficacy of the aggregation implementation. Unfortunately, the enormous amount of data the sensors can produce makes the data integrity problem very computationally expensive. And if the data an aggregator receives is bad, certifying the aggregator could be meaningless.

Communication channel

A communication channel, whether wired or wireless, moves data among things. Certifying communication protocols is plausible, but vetting the implementations is a different problem because each implementation needs to be vetted separately. This problem is similar to that of vetting mobile apps for security and reliability.

External utility

An external utility includes black-box services and hardware devices that networks of things use for storage, aggregation, and decision making. Clouds are an example of an external utility. Certifying third-party public clouds is a near hopeless problem because of the scope, scale, and legal obstacles for the access needed for testing. However, for commercial clouds, there are workarounds. Consider leasing cloud storage and storing valid credit card information in it. To see if your data was leaked, just wait—your bank will notify you if someone tried to use your information. This is a cheap approach for performing your own vetting on an entity for which you have no control. Note, however, that this approach is not foolproof: had you stored photos in the cloud, it might be harder to discover whether they were leaked.

Decision trigger

A decision trigger represents the logic used to transform all data flowing throughout a network of things into the final output of that network. A CA would need to ensure that decisions are made correctly so that all actions taken as a result are appropriate. This would be similar to showing that the network of things satisfies its specification. While decision triggers can be hardwired, they will likely be implemented in software. For decision triggers (as with any software-based component in a network of things), a CA might be asked to argue that the decision trigger has no embedded malicious behaviors such as Trojan horses, Easter eggs, logic bombs, viruses, trap doors, and so on.[3]

The whole system

It is not enough to certify the individual elements of a network of things; the whole system needs to be certified to some level of trust. Certifying things as standalone entities does not solve the fundamental problem of trusting the general promise of moving toward the IoT.[6,7] But, unfortunately, systems of many things are generally untestable due to the significant presence of third-party black-box functionality, scalability, and heterogeneity.

Even if all things are certified, it does not mean they will interoperate well (or correctly) in a fixed environment. Networks of things must be certified with respect to an environment/context, but environments/contexts are hard to define. Even if they could be defined comprehensively, how could a CA certify a distinct network of things for multiple distinct environments? Consider a toaster oven: it is certified for your kitchen counter, not a bathtub full of water.

There is an approach that might help: allowing a network of things to self-diagnose problems in real time by logging and auditing the network's internal data. Although this is not a priori certification of a network before deployment, it could give consumers more confidence over time, provided the log reports are understandable. If they are written in "geek speak," they will likely be useless for consumers.

CERTIFYING PEOPLE

Licensing is a mandatory requirement in US states and other jurisdictions that ensures that individuals offering professional services to the public have some minimal competency to practice. Government authorities (states in the US) license electrical, control, and software engineers. But none of these licenses focus specifically on the IoT—only on basic concepts of reliability, security, and privacy that would be needed in IoT devices. Licensing is also a controversial issue—even so, very few electrical and even fewer software engineers are licensed in the US.

Certification is a voluntary process in which individuals obtain certificates from private entities, vendors, and universities, each acting as its own CA. Microsoft, Cisco, Dell, AT&T, IBM, and many others have certification or partner programs that require certification training. There are also massively open online courses (MOOCs) than can serve the role of promoting minimal competency. But the quality of any individual personnel certification will vary widely, and therefore the quagmire arises from trying to sort out the meaning and value of these differing certifications from differing sources. Hopefully, people who are building things or architecting purposed networks of things have professional credentials. However, to our knowledge, there aren't yet any IoT personnel certifications.

CERTIFYING PROCESSES

Finally, there remains the question of who can certify the processes that were used in building things, testing them, and integrating them. We have no answer. Capability Maturity Model (CMM)-style certification of hardware and software creation processes could be created for the IoT, but the confusion of process standards and the value-add has been its own quagmire for decades.

The original argument from decades ago was that certifying software was too difficult, and the best that could be achieved was certification of the process that built the software. This argument has all but been debunked over the past 20 years. Finally, a vendor's self-certification of processes will have limited and possibly questionable relevance due to conflicts of interest.

Certification of anything—hardware or software—is a challenge, and it is especially challenging for the IoT. Security and privacy might be uncertifiable properties to any useful specificity, even when environments are respected, because environments are difficult to properly define. The inability to certify software and other things to any "believable" level of confidence without enormous expense is a problem for the IoT—"believable" certification takes time and money. This only adds to the morass.

Let's take a hypothetical example of where IoT certification might be immediately useful: healthcare. Why? Cost savings via increased efficiency.[8] In the US, healthcare is regulated via legislation such as HIPPA. As mentioned before, the IoT is not currently regulated. Going forward, the IoT will likely depend on, or control, differing levels of automation, and automation creates the potential for litigation when it fails. The IoT is not well defined, and healthcare has a huge human element (doctors, nurses, first responders, and so on). Poorly defined automation and imperfect humans often do not interact well. The avionics industry and its passengers have learned this the hard way over the past 30 years, when pilots did not understand the automation and vice versa. IoT healthcare will likely suffer similar mishaps until IoT certification comes of age.

But should we let these challenges stop us from certifying IoT technologies? No. We cannot let the perfect be the enemy of the good. When it comes to dealing with the IoT certification quagmire, "good enough" is better than nothing, although "good enough" remains in the eye of the beholder.

REFERENCES

1. H. Chung et al., "Alexa, Can I Trust You?," *Computer*, vol. 50, no. 9, 2017, pp. 100–104.
2. T. Hunt, "What Would It Look Like If We Put Warnings on IoT Devices Like We Do Cigarette Packets?," blog, 13 Oct. 2017; www.troyhunt.com/what-would-it-look-like-if-we-put-warnings-on-iot-devices-like-we-do-cigarette-packets.
3. J. Voas, "Certifying Off-the-Shelf Software Components," *Computer*, vol. 31, no. 6, 1998, pp. 53–59.
4. D. Miorandi et al., "Internet of Things: Vision, Applications and Research Challenges," *Ad Hoc Networks*, vol. 10, no. 7, 2012, pp. 1497–1516.
5. J. Voas, "Networks of 'Things,'" NIST Special Publication SP 800-183, 2016; https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-183.pdf.
6. J. Voas and P. Laplante, "The IoT Blame Game," *Computer*, vol. 50, no. 6, 2017, pp. 69–73.
7. I. Bojanova and J. Voas, "Trusting the Internet of Things," *IT Professional*, vol. 19, no. 5, 2017, pp. 16–19.
8. N.L. Laplante, P.A. Laplante, and J.M. Voas, "Stakeholder Identification and Use Case Representation for Internet-of-Things Applications in Healthcare," *IEEE Systems J.*, 2016; doi: 10.1109/JSYST.2016.2558449.

JEFFREY VOAS is an IEEE Fellow and the 2017–2018 President of the IEEE Reliability Society. Contact him at j.voas@ieee.org.

PHILLIP A. LAPLANTE is a professor of software and systems engineering at Penn State. He is a Fellow of IEEE and SPIE. Contact him at plaplante@psu.edu.

Computer Society Standards Drive Industry

Jon Rosdahl, Qualcomm Technologies, Inc.

The IEEE Computer Society provides core standards for the benefit of industry including for networking, integrated circuit design and test, cloud computing and software and system engineering. These standards enable industry to move technology forward at a rapid pace in order to deliver amazing products to consumers.

FROM THE EDITOR

As I mentioned in the last column, in the next few months, this column will describe the work of the various technical sponsors of standards in the IEEE Computer Society (CS). This column starts that overview by covering five of those technical sponsors. Jon Rosdahl, the 2018 CS Standards Activities Board vice president worked with the chairs to provide the first of these overviews of the Society's technical sponsors. *–F.D. Wright*

The IEEE Computer Society (CS) has three main program areas: publications, conferences, and standards. Each area of activity provides opportunities for contributions from CS members and nonmembers alike. My goal this year as vice president of the Standards Activities Board is to encourage active participation in the many standards areas, and with that in mind, I'd like to share this overview of what standards are currently being developed in hopes that you will want to join in and participate.

What follows is a brief summary of the work program of five of the CS's standards sponsoring committees.

IEEE CLOUD COMPUTING STANDARDS COMMITTEE

The IEEE Cloud Computing Standards Committee (CCSC; www. computer

0018-9162/18/$33.00 © 2018 IEEE

.org/web/standards/cloud) is chartered by the CS's Standards Activities Board to promote the development of standards in all aspects of the cloud computing ecosystem. It facilitates the development and use of standards-based choices by cloud computing ecosystem participants (cloud vendors, service providers, and users) in areas such as cloud application interfaces, cloud portability interfaces, cloud management interfaces, cloud interoperability interfaces, cloud file formats, and cloud operation conventions. CCSC has three active projects, the Guide for Cloud Portability and Interoperability Profiles, the Standard for Intercloud Interoperability and Federation, and the Standard for Adaptive Management of Cloud Computing Environments, which are described below.

Guide for Cloud Portability and Interoperability Profiles (IEEE P2301)

The purpose of this guide, also referred to as CPIP (https://standards.ieee.org/develop/project/2301.html), is to assist cloud computing vendors and users in developing, building, and using standards-based cloud computing products and services, which should lead to increased portability, commonality, and interoperability. Cloud computing systems contain many disparate elements, and each element can offer multiple options that would result in different externally visible interfaces, file formats, and operational conventions. In many cases, these visible interfaces, formats, and conventions have different semantics, and so to define these for a variety of sources, this guide enumerates options and groups them logically into "profiles." In this way, cloud ecosystem participants can benefit from improved portability, commonality, and interoperability, growing the cloud computing adoption rate overall.

IEEE COMPUTER SOCIETY STANDARDS COMMITTEES

The CS Standards Activities Board consists of the chairs of each of the CS's standards sponsoring committees plus additional appointed officers. There are 11 sponsoring committees covering a wide spectrum of topics lead by the noted sponsor chair:

- Cloud Computing Standards Committee—chair, Steve Diamond
- Design Automation Standards Committee—chair, Stan Krolikoski
- Cybersecurity & Privacy Standards Committee—chair, Eric Hibbard
- Learning Technology Standards Committee—chair, Avron Barr
- LAN/MAN (IEEE 802) Standards Committee—chair, Paul Nikolich
- Microprocessor Standards Committee—chair, Baker Kearfott
- Portable Applications Standards Committee—chair, Joseph Gwinn
- Software & Systems Engineering Standards Committee—chair, Paul Croll
- Simulation Interoperability Standards Organization Standards Committee—chair, Katherine Morse
- Test Technology Standards Committee—chair, Adam Cron
- SAB Sponsor Committee—chair, Paul Eastman

Standard for Intercloud Interoperability and Federation (IEEE P2302)

This standard, also referred to as IIF (https://standards.ieee.org/develop/project/2302.html), creates an economy among cloud providers that's transparent to users and applications, which provides for a dynamic infrastructure that can support evolving business models. In addition to the technical issues, appropriate infrastructure for economic audit and settlement must exist.

Standard for Adaptive Management of Cloud Computing Environments (IEEE P2303)

The purpose of this standard (https://standards.ieee.org/develop/project/2303.html) is to provide entities designing, developing, and providing new cloud applications and adaptive management environments with an underlying reference for the adaptive management systems supporting dynamic cloud computing ecosystems. In particular, P2303 focuses on providing a foundational reference for the core structure, information, and components needed to support and maintain cloud computing ecosystems' characteristic highly dynamic environments.

IEEE DESIGN AUTOMATION STANDARDS COMMITTEE

The Design Automation Standards Committee (DASC) is focused on language-based design, modeling, integration, and verification standards for use in creating integrated circuits (ICs). This includes standards for checking timing, design correctness, power usage, intellectual property (IP) reuse, IP encryption, and test of ICs.

DASC standards are used worldwide by developers of consumer,

TABLE 1: IEEE 802 Standard-related activities for the past eight years.

Activity	Year							
	2017	2016	2015	2014	2013	2012	2011	2010
Average attendees per plenary	655	692	754	724	725	730	~800	~900
Projects started	16	18	24	19	17	20	16	19
Projects in process	43	44	41	29	25	23	25	38
Projects completed (total projects ratified)	22	15	11	11	12	21	20	19
Ratified total page count	6373	2178	7302	4171				
Projects withdrawn	01	01	0	0	0	0	03	0
Withdrawn standards	0	01	0	0	0	01	02	01
Total No. of 802 activities/year	80	79	76	59	54	65	66	77
Total No. of active 802 standards and amendments	81	75	75	79	68			
Total No. of 802 standards pages	28,124							

military, and industrial electronics. Indeed, it would be difficult to find an electronic product (besides simple components) that wasn't developed using software that implements at least one DASC standard.

Some of the best-known DASC standards include:

- IEEE Std. 1076—Standard VHDL Language Reference Manual
- IEEE Std. 1800—Standard for SystemVerilog—Unified Hardware Design, Specification, and Verification Language
- IEEE Std. 1801—Standard for Design and Verification of Low Power, Energy Aware Electronic Systems
- IEEE Std. 1850—Standard for Property Specification Language (PSL)

LAN/MAN STANDARDS COMMITTEE (IEEE 802)

The IEEE 802 Local and Metropolitan Standards Committee (LMSC; www.ieee802.org) develops and maintains global standards for local, metropolitan, and other area wireline and wireless networks, primarily within layers 1 and 2 of the Open System Interconnection Reference Model. "Local" means building/campus, and "metropolitan" means intracity. IEEE 802 Standards are being used for WAN (intercity) and personal area network (PAN; in-room) applications as well.

IEEE 802 LMSC has had a significant impact on the networking market. For example, devices with interfaces based on 802 standards are put in service at a rate of more than 100 per second, which is roughly over 3 billion per year. To put the dedicated 802 volunteer participants' contributions in perspective, the total dollar value of their time and expertise are estimated to be in excess of $250 million per year. Participation is completely open, with volunteers from dozens of countries around the globe. All work is documented in a fully open and transparent manner—the complete library of ratified 802 standards has grown to over 28,000 pages.

The committee meets three times per year—in March, July, and November—in plenary session, as it has since it was formed in February 1980. The working groups and technical advisory groups can also hold additional interim sessions as needed—these are typically also held three times per year in January, May, and September.

IEEE 802 has multiple working groups (WGs) and technical advisory groups (TAGs) as described below.

- 802.1 Higher Layer LAN Protocols (working group)
- 802.3 Ethernet (working group)
- 802.11 Wireless LAN (working group)
- 802.15 Wireless Personal Area Network (WPAN) (working group)
- 802.18 Radio Regulatory (technical advisory group)
- 802.19 Coexistence (working group)
- 802.21 Media Independent Handoff (working group)
- 802.22 Wireless Regional Area Networks (WRAN) (working group)

There are also two hibernating (inactive) working groups for 802.16 Broadband Wireless Access and 802.20 Mobile Broadband Wireless Access.

The level of activity within IEEE 802 is illustrated in Table 1.

IEEE SOFTWARE AND SYSTEMS ENGINEERING STANDARDS COMMITTEE

The IEEE Software and Systems Engineering Standards Committee (S2ESC) has been setting standards in this field for four decades, managing the scope and direction of IEEE Software and Systems Engineering and Standards. The committee was initially chartered by the CS in 1976 to develop the first standards for software engineering that

would support a family of products and services based on software engineering standards for use by practitioners, organizations, and educators to

- improve the effectiveness and efficiency of their software engineering processes,
- improve communications between acquirers and suppliers, and
- to improve the quality of delivered software and systems containing software.

As more and more system functions became dependent upon software for realization, S2ESC expanded its scope to include standards for systems engineering.

S2ESC is represented on the IEEE-CS Standards Activity Board, with longtime ties to the IEEE-CS Technical Council on Software Engineering. Under the oversight of the IEEE Standards Association, S2ESC has worked to provide a standards collection that provides a consistent view of the state of the practice, is aligned with the Software Engineering and Systems Engineering Bodies of Knowledge (SWEBOK and SEBOK), addresses practitioner concerns, and is affordable.

From a broader perspective, in addition to the development of standards, S2ESC develops supporting knowledge products and sponsors or cooperates in annual conferences and workshops in its subject area. S2ESC also helps coordinate international standards-making between IEEE and ISO/IEC JTC 1 through the IEEE-CS membership in the U.S. Technical Advisory Group (TAG) to ISO/IEC JTC1/SC7 and through a direct liaison between JTC 1/SC7 and IEEE-CS.

The mission of S2ESC

1. Develop and maintain a family of software and systems engineering standards that is relevant, coherent, comprehensive, and effective in use. These standards are for use by practitioners, organizations, and educators to improve the effectiveness and efficiency of their systems and software engineering processes, to improve communications between acquirers and suppliers, and to improve the quality of delivered software and systems.
2. Develop supporting knowledge products that aid practitioners, organizations, and educators in understanding and applying our standards.
3. Support and promote a Software Engineering Body of Knowledge, certification mechanisms for software engineering professionals, and other products contributing to the profession of software engineering.

As the marketplace became less prescriptive, S2ESC refocused its standards as "best practice" information aids. Our standards can be viewed as consensus documents that codify accepted common practice.

To ensure that the needs of the standards user community are being met, S2ESC actively tracks the needs for the standardization of software and systems engineering practices and sponsors working groups that perform the core standards implementation activities.

A key tenet of S2ESC's portfolio is the harmonization of system and software life cycle processes and their supporting standards in order to provide a consistent approach to the engineering of software-intensive systems. This includes ensuring that our IEEE standards are compatible with those of other relevant international standards bodies. S2ESC has made great strides, through our liaison with ISO/IEC JTC1/SC7, Software and Systems Engineering, in making IEEE and SC7 standards consistent and complementary through mutual adoptions, joint development, and coordinated revisions. As such, our joint umbrella standards for systems and software life cycle processes, ISO/IEC/IEEE Std. 15288 (System Life Cycle Processes) and ISO/IEC/IEEE Std. 12207 (Software Life Cycle Processes), and their supporting standards drive much of S2ESC's current portfolio of over fifty standards supporting the entire life cycle.

S2ESC continues to explore new areas of interest to the community. Recent areas of interest include systems and software assurance, DevOps, architectural evaluation, and ethical concerns in system design.

As more and more system functions became dependent upon software for realization, S2ESC expanded its scope to include standards for systems engineering.

TEST TECHNOLOGY STANDARDS COMMITTEE

The Test Technology Standards Committee (TTSC) has been busy creating and refining standards addressing electronics testing, usually focusing on integrated circuit testing. This technical sponsor has completed 19 standards through the years, two new standards in the development stages, and two new working groups in the early definition stages.

The two new working groups will be focusing on analog test access and fault coverage. The two standards in development include a layer of IEEE Std. 1687 called P1687.1, which seeks to define an API and perhaps some hardware to enable access to IEEE Std. 1687 networks via chip-level functional interfaces. The second is P1838 which is in the final stages of defining design for test (DFT) access for chip stacks, or 3D-ICs. This standard is also leveraging other tried-and-true standards

such as IEEE Std. 1149.1, IEEE Std. 1500, and IEEE Std. 1687.

Other test technology standards include

- IEEE Std. 1149 - Series on Test Access Ports and Boundary Scan Architecture
- IEEE Std. 1450 - IEEE Standard Test Interface Language (STIL) for Digital Test Vector Data
- IEEE Std. 1500 - Standard Testability Method for Embedded Core-based Integrated Circuits

IEEE Computer Society standardization activities cover a broad range of technology areas that are core to the converging world of computers, communications, and consumer electronics. Without IEEE Std. 802.11 and other LAN/MAN standards, system and software engineering development process standards, and cloud computing standards, today's ever more mobile users would find doing their jobs and staying in touch with friends and family extremely difficult. Without design automation and test technology standards, the design, production, and test of integrated circuits with ever increasing gate counts would grind to a halt.

JON ROSDAHL is a Senior Staff Engineer with Qualcomm Technologies, Inc. and the 2017-2018 Computer Society Vice President of Standards. He also serves as a member of the 802 Executive Committee.

MOTI YUNG TO RECEIVE THE 2018 IEEE COMPUTER SOCIETY W. WALLACE MCDOWELL AWARD

Moti Yung, Fellow of IEEE and ACM, has been selected to receive the 2018 IEEE Computer Society W. Wallace McDowell Award. He is recognized "for innovative contributions to computer and network security, predicting both attack scenarios and design needs in this important evolving era."

Yung is an adjunct senior research faculty member at Columbia University where he has worked with and co-advised numerous PhD students over the years. In parallel, he has had a career as a research scientist in industry, including for IBM Research, Certco, RSA Laboratories, Google, and Snap.

Yung's main professional interests are in security, privacy, and cryptography. His contributions to research and development treat science and technology holistically—from theoretical mathematical foundations, via conceptual mechanisms which typify computer science, to participation in industrial product design and development.

His work predicts the needs secure systems and analyzes potential threats, which has led to the development of basic theoretical and applied notions, including ransomware attacks, cryptosystems subversion attacks, concurrent sessions in authentication protocols, strong secure encryption, and digital signatures from simplified cryptography.

His industrial work led to new diversified mechanisms, some of which are in extensive use, including public-key-based second-factor authentication devices; new factors for user identification; distributed signing methods; numerous very large scale (web and mobile) encryption schemes; anonymization of historical user data; transparency and control for web users; secure data collection; secure, large-scale distributed computation protocols for privacy-preserving data analytics; and secure cloud storage.

Also a Fellow of the International Association for Cryptologic Research (IACR), and the European Association for Theoretical Computer Science (EATCS), Yung gave the IACR Distinguished Lecture in 2010. He also received the 2014 ACM SIGSAC Outstanding Innovation award, the 2014 ESORICS (European Symposium on Research in Computer Security) Outstanding Research award, an IBM Outstanding Innovation award, a Google OC award, and a Google founders' award.

The McDowell Award recognizes individuals for outstanding recent theoretical, design, educational, practical, or other innovative contributions in the field of computing. The award is given for a single contribution of great merit or a series of lesser contributions that have had an important influence on the computing field. The award consists of a bronze medal and a $2,000 honorarium, and it will be presented at the IEEE CS annual awards ceremony on Wednesday, 6 June 2018, in Phoenix.

One of computing's most prestigious individual honors, the W. Wallace McDowell Award has a list of past winners that reads like a who's who of industry leaders. They include FORTRAN creator John W. Backus (1967); supercomputer pioneers Seymour Cray (1968), Gene Amdahl (1976), and Ken Kennedy (1995); the architect of IBM's mainframe computer Frederick Brooks (1970); Intel Corp. co-founder Gordon Moore (1978); COBOL creator Grace Murray Hopper (1979); Donald Knuth, the father of algorithm analysis (1980); microprocessor inventor Federico Faggin (1994); World Wide Web inventor Tim Berners-Lee (1996); Lotus Notes creator and Microsoft Chief Software Architect Ray Ozzie (2000); and IBM Fellow Ronald Fagin (2012).

For more information on the award, including a complete list of past recipients, visit www.computer.org/portal/web/awards/wallace.

CHANGES TO SOCIETY BYLAWS AVAILABLE ONLINE

The IEEE Computer Society Board of Governors recently approved the first reading of amendments to the Society's bylaws.

Changes to Article II, Section 2 and Section 5, which cover the nominations and elections of IEEE Computer Society Board of Governors members, will reduce the number of annually elected Board member positions from 7 to 6, and as a result the required nominees will change from 11 to 9.

The proposed bylaws section is available for review at https://goo.gl/EgNGwJ. Changes are marked in the text. Only the relevant segments of the bylaws in question are reproduced.

Changes to existing Society bylaws that receive first- and second-reading approval by the Board of Governors are listed by title in *Computer*, with links to a website location hosting the actual documents. The documents remain accessible at this location until such time as the changes receive final approval.

Members can send comments to Anne Marie Kelly (amkelly@computer.org) by close of business 31 May 2018.

0018-9162/18/$33.00 © 2018 IEEE

IEEE COMPUTER SOCIETY: Be at the Center of It All

IEEE Computer Society membership puts you at the heart of the technology profession—and helps you grow with it.

Here are 10 reasons why you need to belong.

IEEE Computer Society—keeping you ahead of the game. Get involved today.

www.computer.org/membership

IEEE computer society

CONNECT ON INTERFACE

Explore **INTERFACE**, a communication resource to help members engage, collaborate and stay current on Computer Society activities. Use **INTERFACE** to learn about member accomplishments and find out how your peers are changing the world with technology.

We spotlight our professional sections and student branch chapters, sharing their recent activities and giving leaders a window into how chapters around the globe grow, thrive and meet member expectations. Plus, **INTERFACE** will keep you informed on Computer Society–related activities so you never miss a meeting, career development opportunity or important industry update.

Connect today at interface.computer.org

IEEE COMPUTER SOCIETY
INTERFACE